# NSWS Report (SESTO)

*Lim Wei Jian & Song Cheng Yan*

**Abstract**

This comprehensive guide aims to help future users understand the features, functionalities, and operation of the SESTO IMagnus. It includes detailed instructions for setting up the SESTO IMagnus and its custom-made remote control. By providing clear and concise information, we strive to make controlling the SESTO IMagnus intuitive and accessible for all users, ensuring they can maximise its capabilities and enhance their overall experience.

**Acknowledgments**

We would like to express our sincere gratitude to Nigel for his invaluable troubleshooting of the Sesto Robot, as well as to Nicholas Chew and Eugene for their expert technical support in payload and PCB design. We also extend our thanks to Yee Teng, Patrick, Ms. Annie, and Alvin for their invaluable advice and logistical support. Our appreciation goes to the Lift Door Startup team for setting up a new door communication system, which significantly enhanced our project. Lastly, we are deeply grateful to Royston and Dr. Yen for giving us the opportunity to bring our projects to life. Their contributions have been essential to the success of this endeavour, and we truly appreciate all the support.

# Table of Contents

## 1.1 Introduction (SESTO IMagnus)

The SESTO IMagnus is a platform robot that can carry up to 300 kgs, it can be programmed using Fleet UI or manually controlled with the controller. There's two missions for it, one is to transport the blood packs in the hospital, while another is to transport the toolbox in the Innovation hub. Therefore, the current cabinet on it has two slots, one of which is to put the cooler boxes and another is to put the toolbox. For simplicity, we shall now call the SESTO IMagnus, Sesto, from here on.

***PLEASE READ BEFORE PROCEEDING:***

> The follow steps is a summary to control and program the Sesto, to obtain more details of the Sesto software or troubleshoot any errors that are not stated here, please read the software manual of the Sesto.

### *1.1.1 Login to UI*

There are two UI for the Sesto, Fleet and AMR. Fleet is used to control and program the Sesto, while AMR is to configure and troubleshoot the Sesto. To use the interface, you will need to connect your computer's WIFI to the local server of Innovation hub. After connecting the WIFI, make sure the Sesto is switched on.

*The details of the WIFI:*
*Name: TP-Link_DP83*
*Password: 46193346*

To login to the Fleet, open any browser and search for "http://192.168.0.100", where 192.168.0.100 is the local IP of the Sesto.

*The details of the account:*
*Name: partner*
*Password: P@ssword123*

To login to the AMR, open any browser and search for "http://192.168.0.100/AMR".

*The details of the account:*
*Name:*
*Password:*

### *1.1.2 Setup*

After switching on the Sesto for a while, it should start flashing blue which means its' recovering the location. Go to Fleet, under the monitoring and control page select the recovery point that the Sesto is located.

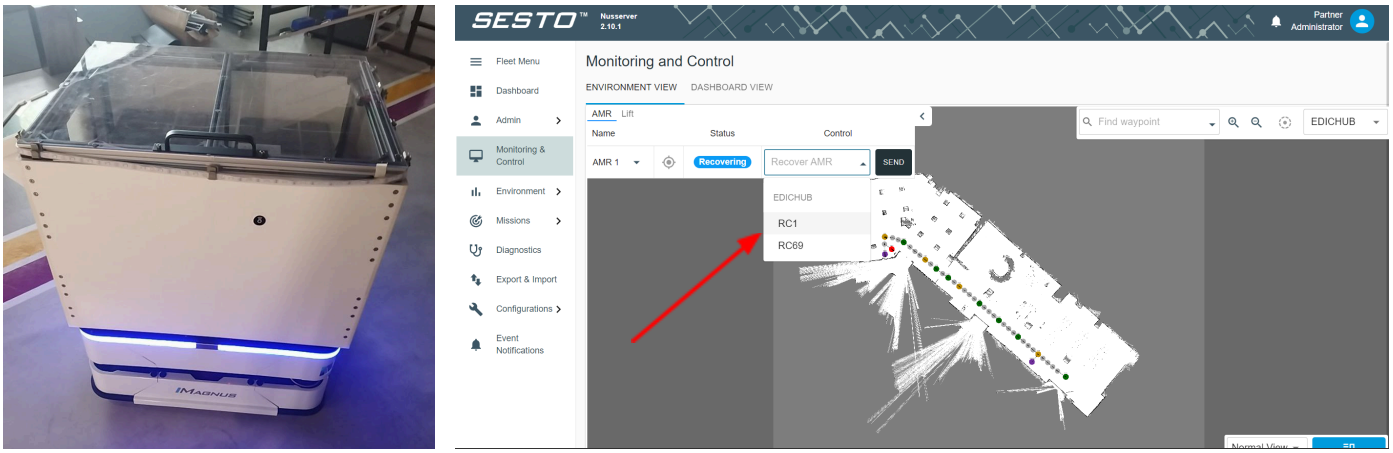


*Figure 1.1.2.1: Sesto Flashing Blue & Recovery Point Selecting*

If the Sesto is not in a recovery point, press on recovery and locate the location of the Sesto in the map.

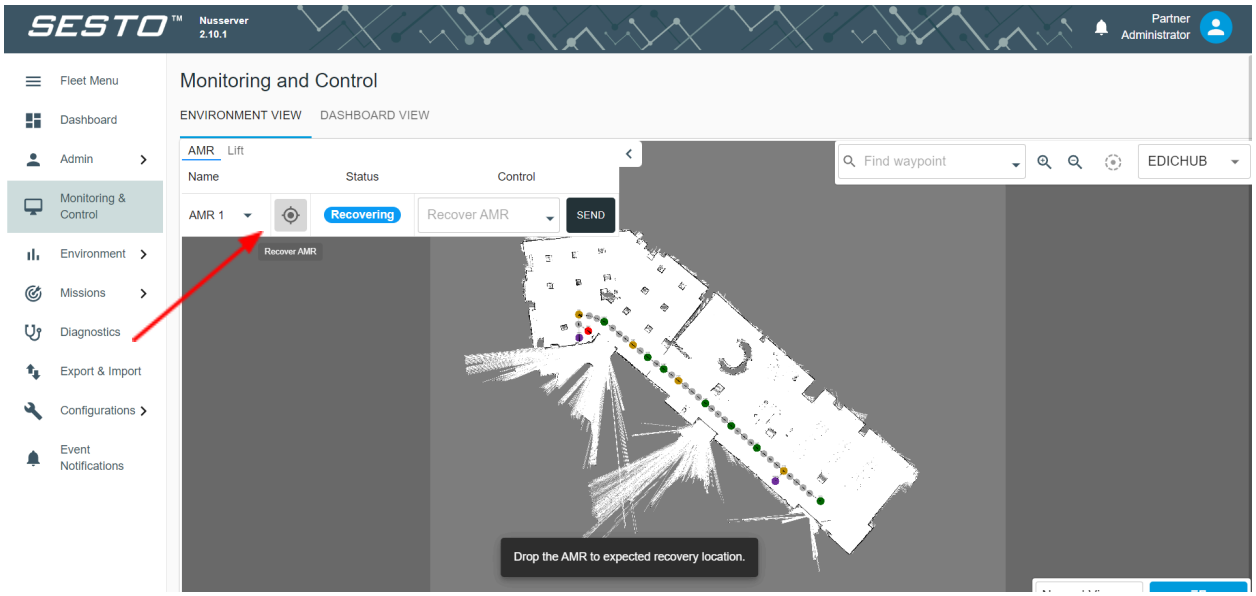


*Figure 1.1.2.2: Selecting Recovery Location*

Ensure the E stopped button is not pressed, the Sesto should light up light blue and you will see the monitoring page showing Sesto status as “Idle”. With that you have done setting up.

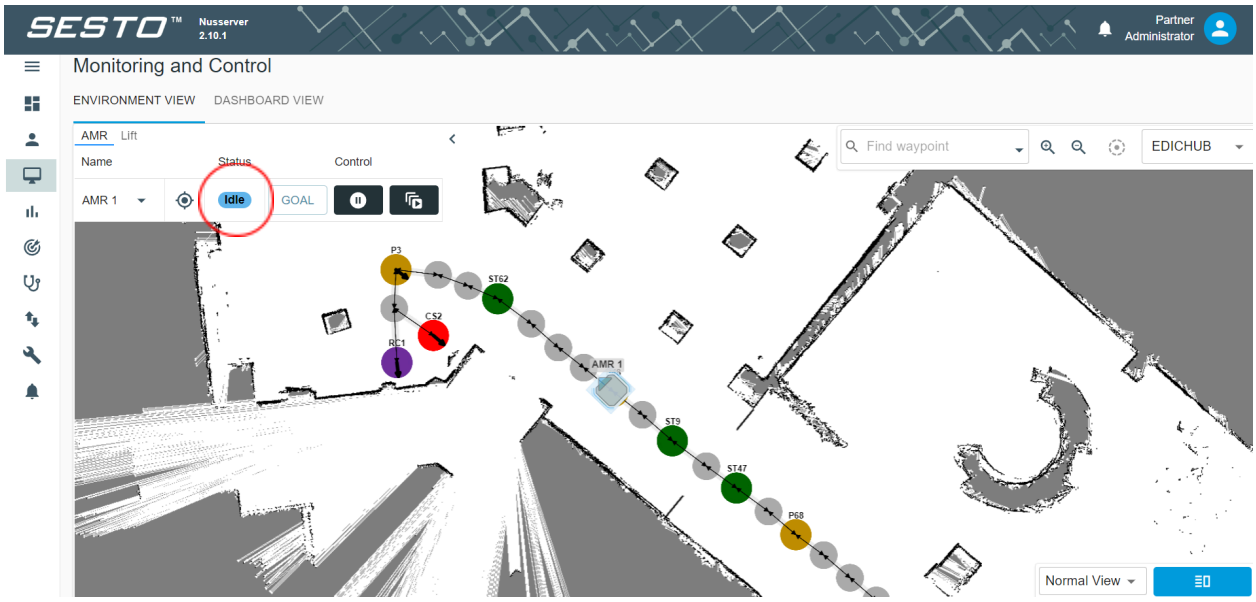


*Figure 1.1.2.3: IDLE State*

## *1.1.3 Control*

You can control the Sesto to any station at any waypoint on the map.
Some important waypoint colours:

| Colour | Meaning |
|---|---|
|  | Recovery Point |
|  | Charging Station |
|  | Parking Station |
|  | Station |

*Table 1.1.3.1: Waypoint Colours*

The Sesto will only park at a parking station, recovery point and charging station. Therefore, if the Sesto is called to any station, it will pause on the spot and go to the nearest parking station once you release it.


*Figure 1.1.3.1: Release Sesto*

If the Sesto is blocked when travelling to a waypoint, and it could not find any path to avoid the obstacles, it will go to the nearest parking station before repeating the goal in 2 minutes.

### *1.1.4 Add Waypoints*

To add or edit waypoints of the map, under the environment page select path settings. After that, select "edit path", followed by "load path from server". The currently used path is "EDICHUB".

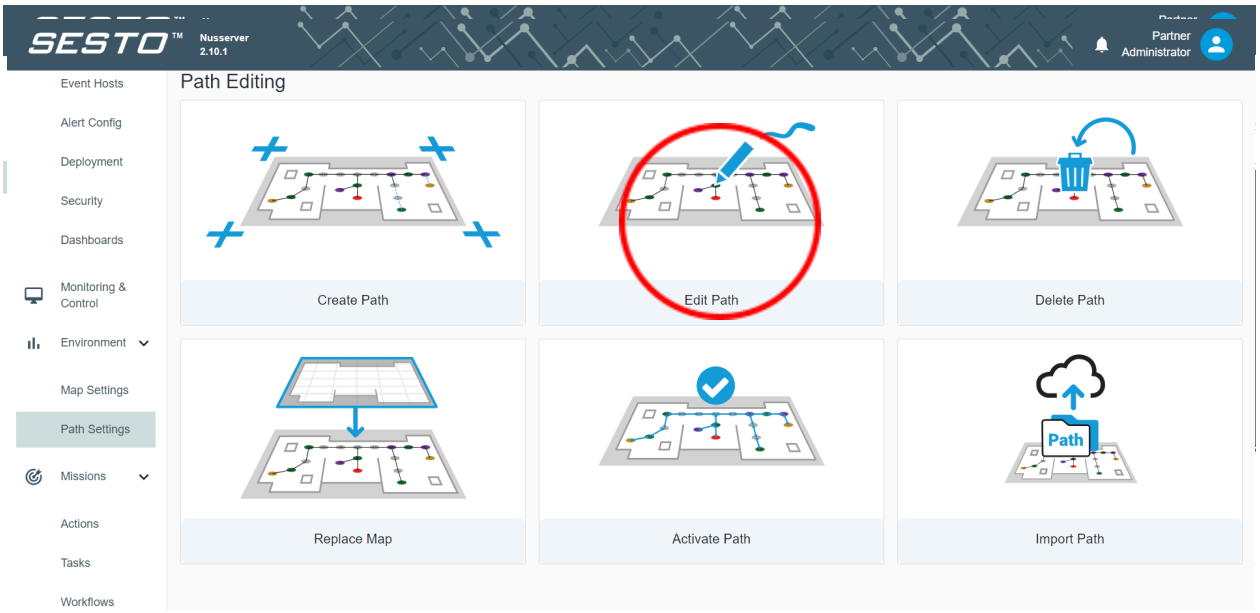


*Figure 1.1.4.1: Edit Path*

You will see different types of waypoints under the type list in “Plotting Configuration”.

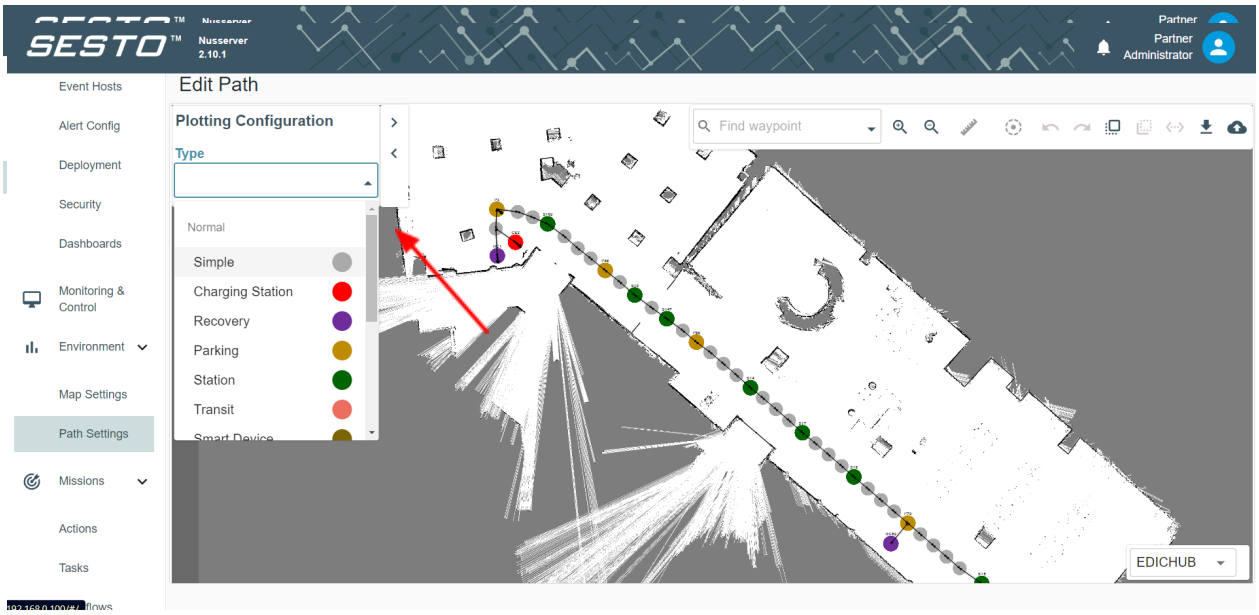


*Figure 1.1.4.2: Waypoints List*

To add a station in the map, select the station and click on the desired location in the map. Then click the arrow beside the waypoints list. Use a bidirectional arrow to join the new waypoint with other waypoints. Note that it will auto generate simple waypoints (grey colour node) for its route.

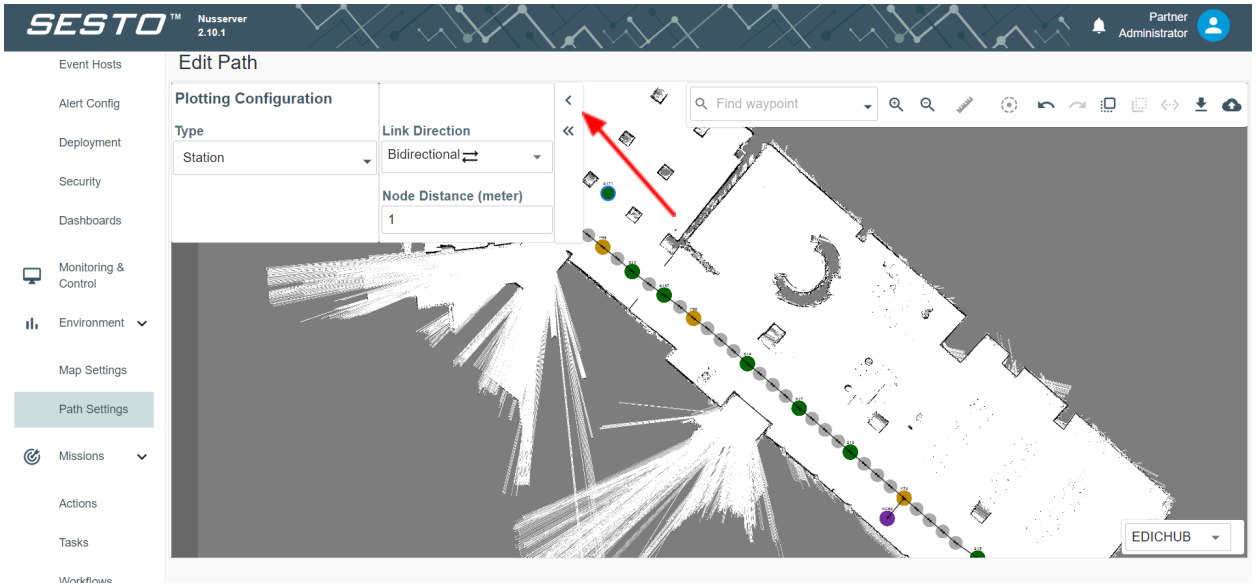


*Figure 1.1.4.3: Link Direction*

After finishing adding the nodes, click upload and save the new map.

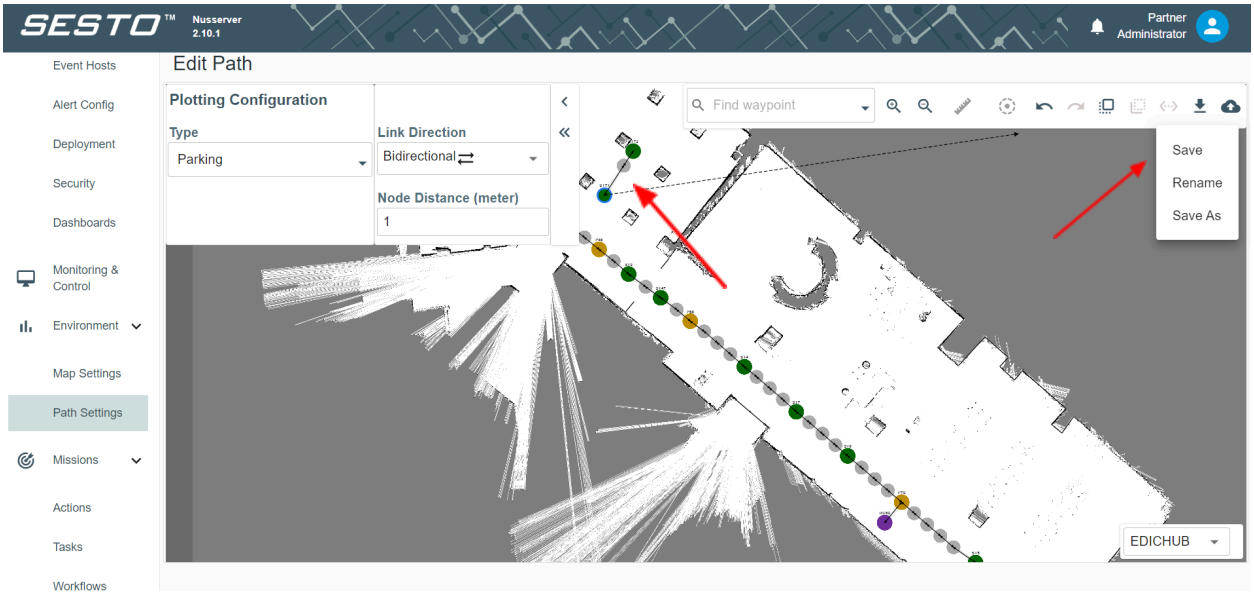


*Figure 1.1.4.4: Linking Nodes & Save*

*To edit the map, read the software manual :)

## *1.1.5 Program Workflow*

To add a workflow, under the missions page select workflow. Then click “new workflow”, followed by “fleet 1”.

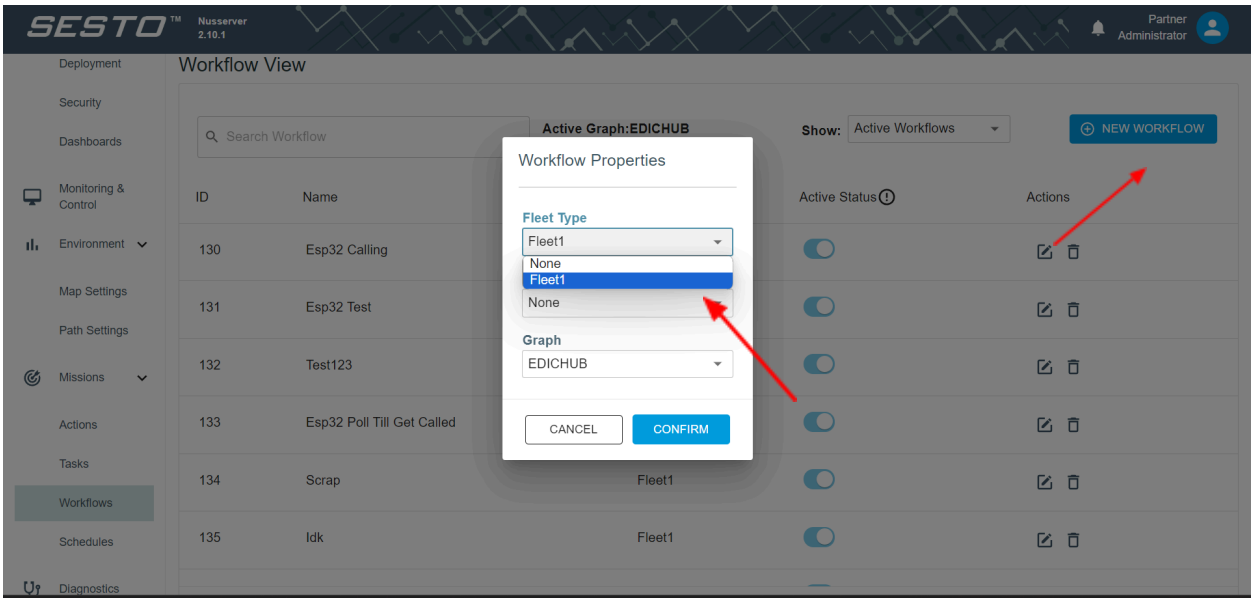


*Figure 1.1.5.1: Linking Nodes & Save*

Pull out “sequence” in “control flow”, and “station” in “general action”.

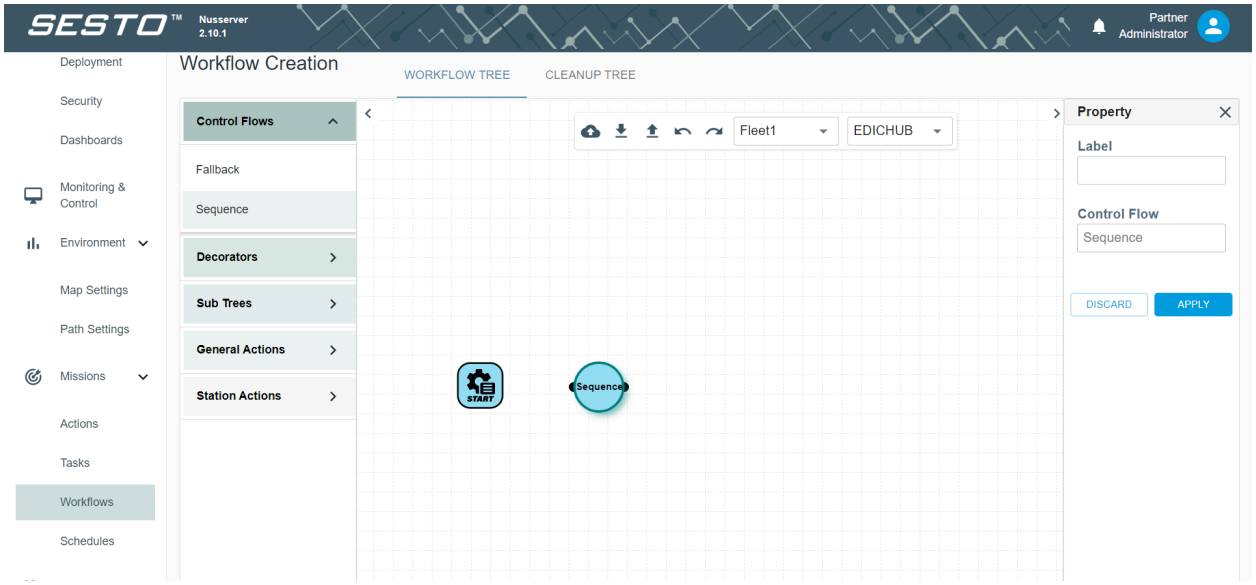


*Figure 1.1.5.2: Sequence*

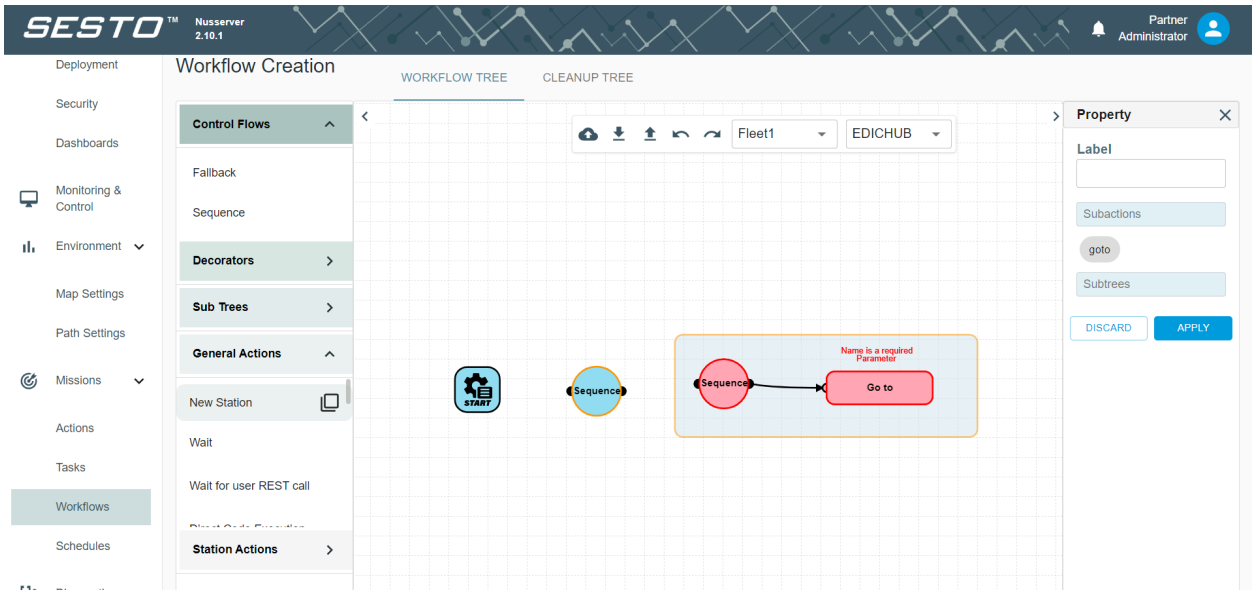


*Figure 1.1.5.2: Station*

Click on “go to” and select the desired station to go to. Right click on the boxes to join them, and save the workflow before leaving.

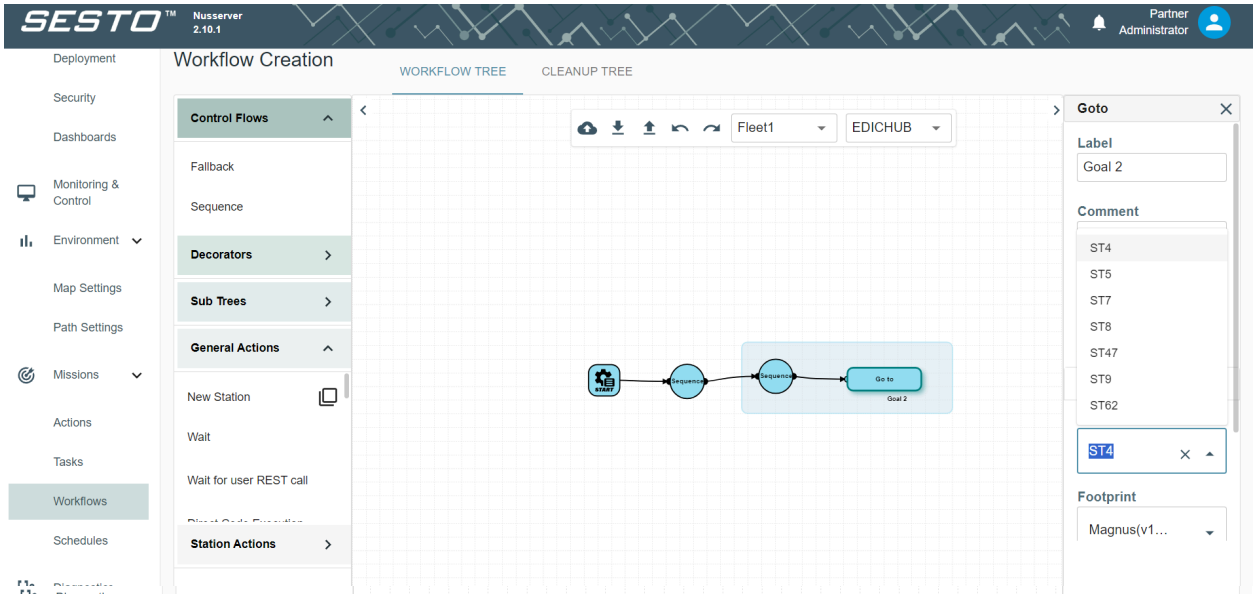


*Figure 1.1.5.3: Select Station*

## *1.1.6 Program Task*

Under the mission page, select the task and click on “create new task”. Next, select the desired workflow and click “create”, the Sesto will then carry out the task.

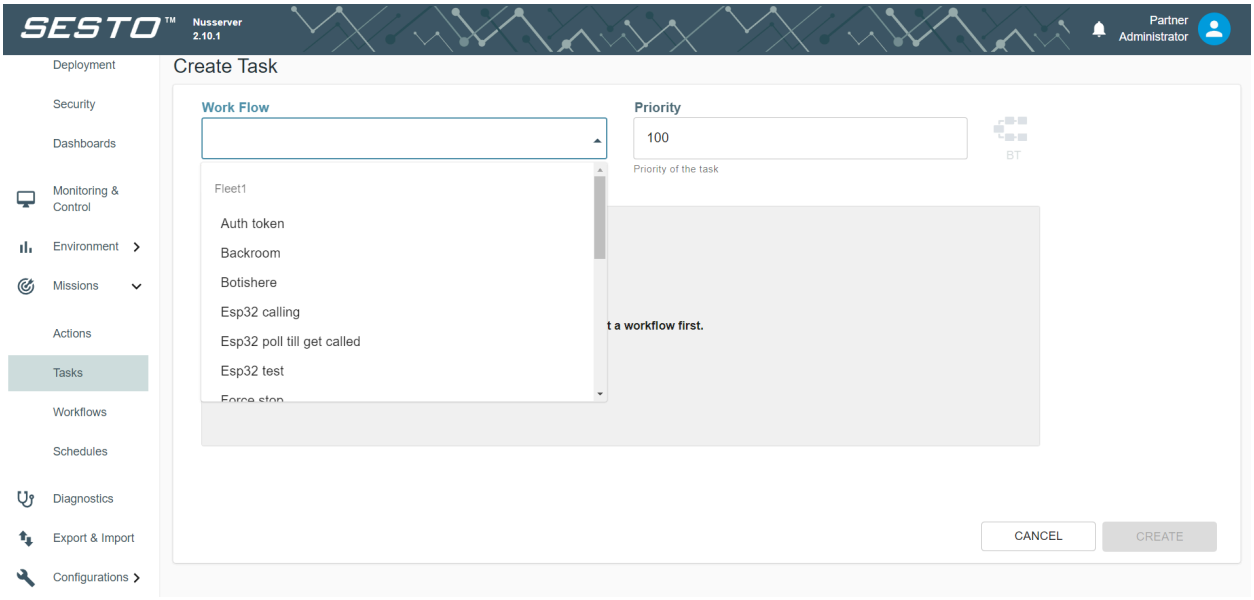


*Figure 1.1.6.1: Creating Task*

### *1.1.7 Change the Payload*

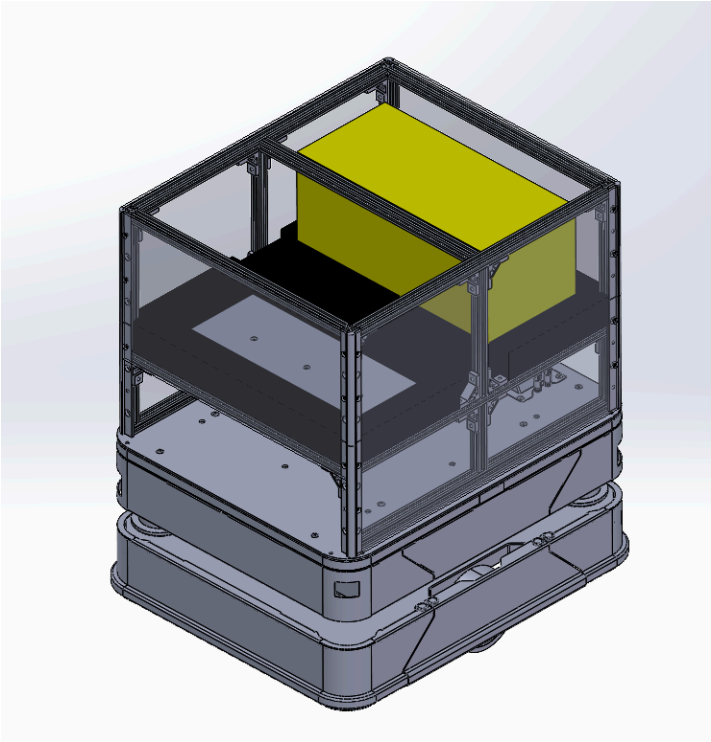

*Figure 1.1.7.1: Current Sesto and Payload*

To change the payload of the Sesto, remove the cabinet by unscrewing the screws on the profile bars that are on the Sesto. Before removing the cabinet from the Sesto, unplug the USB that connected to the Sesto interface.

When designing a new payload for the Sesto, can reference the footprint of the cabinet under section 3.1.6.

## 2.1 Introduction (SESTO Remote Control)

*Figure 2.1.1: Sesto Remote Controller*

The SESTO Remote Control's main function is to allow users to send various commands towards the SESTO IMagnus without using the provided application. It uses the ESP32-S3-DevKitC-1 as a microcontroller to send HTTP requests towards the robot, allowing it to execute the command. For simplicity, we shall now call the SESTO Remote Control, SRC, from here on.

The SRC contains multiple electrical and mechanical components, as stated below:

1. ESP32-S3-DevKitC-1
2. Customised PCB (refer to section 3.1 references for detailed description of PCB)
3. On / Off Switch
4. 3.7 V, 6000 mAh battery pack (1S2P)
5. 9 clicky Key Switches
6. Customised Casing for the PCB
7. SSD1306 OLED Display (0.96 inch)
8. AdaFruit Industry Lithium-ion Charger

## 2.2 SRC Power Management and Maintenance

### *2.2.1 Powering On the SRC*

To power on the SRC, flip the switch at the right side of the SRC upwards. Then, Press the button number 9 (referenced below under section 2.3). This will turn on the display, followed by the connections between the ESP32 and the WiFi. When the WiFi is connected, the screen will show the WiFi status and the local IP Address. You will finally be greeted with the Main Menu.

### *2.2.2 Sleep mode of the SRC*

To turn the SRC into sleep mode, ensure that your screen is inside the Main Menu, then press the button number 9. This will close the display and disconnect it from the WiFi.

### *2.2.3 Closing the SRC*

After turning the SRC into sleep mode, users can flip the switch downwards to turn off the SRC fully. This is to conserve the power of the battery.

### *2.2.4 Restarting the SRC*

To restart the SRC, simply follow the steps from section 2.2.3, then followed by section 2.2.2. In other words, turn off the SRC properly, and turn it on again.

### *2.2.5 SRC Power Enquiries*

***Charging the Battery***

Users can charge the battery with or without closing the SRC. User only needs to get a USB-MINI to charge the battery.

***Changing the Battery***

Changing the battery is as easy as just plucking out the JST wires connected to the AdaFruit Lithium-ion Battery Charger. Make sure to use the battery with the same specifications!

## 2.3 Controls and Navigation

*2.3.1 Functionalities of the SRC*

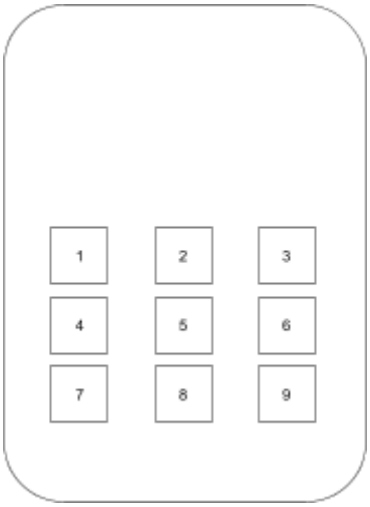


*Figure 2.3.1.1: Template of Remote Control*

As Figure 2.3.1.1 shows, there are a total of 9 buttons on the SESTO Remote Control (SRC). The table shown below will tell the functionalities of each of the buttons:

| Button No. | Functionality |
|---|---|
| 1 | No Functionality |
| 2 | No Functionality |
| 3 | No Functionality |
| 4 | *Left Button* |
| 5 | *Select Button* |
| 6 | *Right Button* |
| 7 | *Back Button* |
| 8 | No Functionality |
| 9 | *Power ON/OFF Button* |

Although there are a total of four buttons on the remote control that currently serve no specific purpose, users have the flexibility to implement custom functionalities for these buttons. This allows for a high degree of customization, enabling users to tailor the remote control to their specific needs and preferences. Whether it's assigning tasks, creating shortcuts, or integrating new features, these unused buttons provide a versatile platform for enhancing the SESTO robot's control interface.

*2.3.2 User Interface*

Inside the Main Menu, you will be greeted with 4 different Icons, each with its own respective functionalities. These are “Open Cabinet”, “Tinkering Corner”, “Battery Level”, and “Sandbox”. Below here shows the details of each of the functionalities:

| Icons | Descriptions |
|---|---|
|  | ***FUNCTION 1: OPEN CABINET***<br>Sends a HTTP request to another ESP32 inside the cooler box, which upon calling, will unlock the cooler box for approximately 5 seconds. A green light on the ESP32 of the cooler box indicates that it is open; red / no light indicates it is closed. |
|  | ***FUNCTION 2: TINKERING CORNER***<br>Sends a HTTP request to the SESTO IMagnus, which will either allow it to play, pause/resume, or cancel its journey to the Tinkering Corner Room. (refer to table 2.3.2.2 for the sub-functions)<br>(For now, this function is broken, since the door has been reconfigured and cannot be called until the router can connect to the Internet) |
|  | ***FUNCTION 3: BATTERY LEVEL***<br>Sends a HTTP request to the SESTO IMagnus, which will return its battery level. The battery level of the remote control will also be listed here. |
|  | ***FUNCTION 4: SANDBOX***<br>Sends a HTTP request to the SESTO IMagnus, which will either allow it to play, pause/resume, or cancel its journey to the Sandbox Room. (refer to table 2.3.2.2 for the sub-functions)<br>(For now, this function is broken, since the door has been reconfigured and cannot be called until the router can connect to the Internet) |

*Table 2.3.2.1: Functionalities inside the Main Menu*

Users are able to navigate around the Main Menu by pressing either the *Left or Right Buttons*. To select one of the functionalities, simply click the *Select Button*.

If users want to return to the Main Menu, simply select the “Back Button” on the SRC.

**Take note that if the button press is not responsive, just click and hold until a window pops up.**

There will be sub-functions upon selecting the room options (Tinkering Corner & Sandbox). These functions are: Play, Resume / Pause, and Cancel.

| Icons | Descriptions |
|---|---|
|  | ***SUB-FUNCTION 1: PLAY***<br>Sends a HTTP request to the SESTO IMagnus to play its journey towards the selected room. If SESTO IMagnus is already in the selected room, the journey is instantly cancelled. |
|  | ***SUB-FUNCTION 2: PAUSE/RESUME***<br>Sends a HTTP request to the SESTO IMagnus to either pause or resume its journey towards the selected room. |
|  | ***SUB-FUNCTION 3: CANCEL***<br>Sends a HTTP request to the SESTO IMagnus to cancel its journey towards the selected room. Upon selecting this sub-function, the SESTO IMagnus will return to its nearest parking spot. |

*Table 2.3.2.2: Sub-functions upon room selections*

If you are in the sub-menu and want to return to the Main Menu, simply press the “Back Button” on the SRC.

For all functions and sub-functions, the OLED display will show a successful window, if the HTTP request is, of course, successful. Likewise, the display will show an error window, if the HTTP request fails to be sent. (Refer to 2.5: Troubleshooting, for troubleshooting guidances)

## 2.4 Customisation

> *In this section, users can learn and customise the **unutilised buttons** for their own personal needs with the SESTO IMagnus (Unlikely though). Feel free to use the guide below as a beginning point (No block diagram since it requires long explanations.), or skip this section if your confident at reading the code line by line.*

The SRC code is programmed using the Arduino IDE and can be found on GitHub under the filename "master.ino" (refer to section 3.1: References for the GitHub link). This guide focuses on customising the remaining four unused buttons for personal use. Detailed explanations of the code itself are not included here; for that, please refer to the comments within the master.ino file. This guide serves as a beginning for users to learn and utilise the SRC, and will specifically cover setting up API endpoints and making HTTP request calls for each of the buttons.

### *Step 1: Setting up API Endpoints*

API endpoints are URLs that the ESP32 connects to in order to send HTTP requests. These requests are typically formatted using structured data formats like JSON or XML. In this guide, we will exclusively use the JSON format. Below is an example of how the API endpoints will be structured:

```
68    // This is the API endpoints used throughout the code
69    const char* cabinetserverName = "http://192.168.0.104/request";
70    const char* sestoinfoServerName = "http://192.168.0.100/public/amrs/statuses";
71    const char* requestServerName = "http://192.168.0.100/public/tasks";
72    String postServerName = "http://192.168.0.100/public/tasks/";
```

*Figure 2.4.1.: API endpoints*

To set up their own API endpoints, users should read the **SESTO API Manual**, given by SESTO themselves. The API endpoints should always start with this format: `http://192.168.0.100/public/…` This is followed by the directory shown inside the API Manual.

### *Step 2: Defining a Function*

Users should define a function for each separate HTTP request, to make the code readable. There are types of requests the user can do: POST, GET, PUT, DELETE. Typically, users would only either want to receive, or send a JSON data from the SESTO IMagnus. Here is a typical structure of the function:

```
224  int request_tinkering_corner() {
225      WiFiClient client;
226      HTTPClient http;
227      http.begin(client, requestServerName);
228      http.addHeader("Content-Type", "application/json");
229      http.addHeader("Authorization", AUTH_KEY_SESTO);
230      int httpResponseCode = http.POST("{\"amr_id\":15,\"workflow_id\":146}");
231      http.end();
232      return httpResponseCode;
233  }
```

*Figure 2.4.2.: Example Function*

**SENDING A COMMAND (POST)*

To send a command to the SESTO IMagnus, follow the format of `int request_tinkering_corner()` . Use `void` instead of `int` and do not `return` anything.

Next, under the line of code `http.begin(client, requestServerName);` , change "`requestServerName`" to the variable associated with the API endpoint the user has defined.

From the line `int httpResponseCode = http.POST("{\"amr_id\":15,\"workflow_id\":146}");` , change the parameters to the required parameters stated inside the SESTO API Manual.

**RECEIVING DATA (GET)*

To receive data from the SESTO IMagnus, follow the format of `int retrieve_workid()`. Since users only want to receive information from the SESTO IMagnus, there is only one thing that needs to be changed. The line `int id_data = object[0]["id"];` receives a value from a dictionary. Users can utilise this to receive data by changing the second parameter from `"id"` to other keys. Again, refer to the API Manual for the various keys available.

*NOTE: When returning the information, remember to change the function to the respective data structure of the returned value (String, int, float, etc)*

*Step 3: Implementing the Functions*

Scroll down to the `void loop()` section of the file. This is where all the logic of the SRC is located at. To implement your functions onto the unutilised buttons, add your functions onto the else if statements that contain nothing in it (These else if statements should contain button_reading values of 1, 2, 3 and 8). Refer to Section 2.3.1.1 for reference of each of the button numbers.

## 2.5 Troubleshooting

*2.5.1 What if the screen bugged out?*

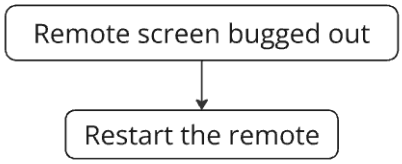


*Diagram 2.5.1.: Screen Bugged Out*

*2.5.2 What if the HTTP request keeps failing?*

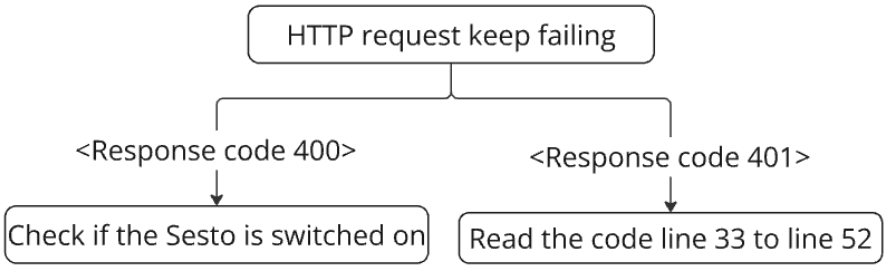


*Figure 2.5.2.: HTTP Request Failed*

*2.5.3 Can't open the cabinet?*

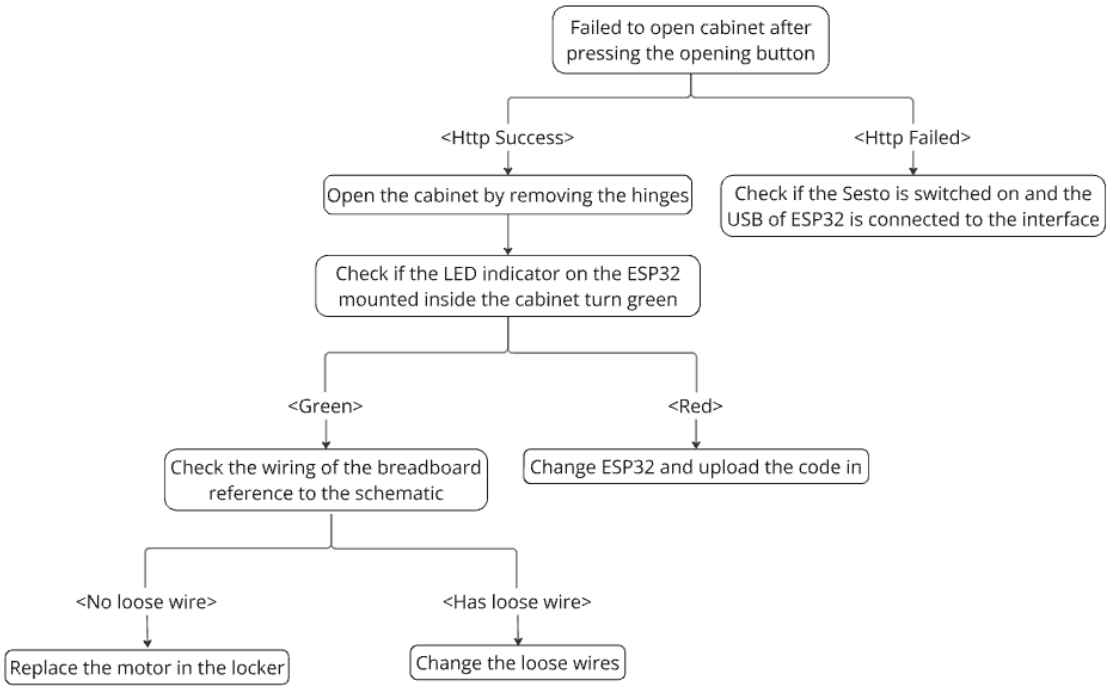


*Figure 2.5.3.: Cabinet Failed to Open*

### *2.5.4 Sesto occurring path block without any obstacles*

Possible Solution:

1. Switch off the environment lights
2. Recalibrate the sensors, reference to the manual
3. Reset the Sesto

### *2.5.5 Sesto flashing red*

Possible Issue:

1. E-Stop is pressed
2. Sesto's battery is low
3. Sesto ran out of its' path

## 3.1 References

### *3.1.1 PCB components*

| | |
|---|---|
| Display | : 0.96 inch I2C OLED Display |
| Diodes | : 1N4148WS |
| Resistors | : Vishay Intertech MRS16000C1002FRP00 |
| Header Pins | : CS-1165SDIG-1*22 |
| Display connector | : S4B-PH-SM4-TB(LF)(SN) |
| Power Supply connector | : S2B-PH-SM4-TB(LF)(SN) |
| Microcontroller | : ESP32-S3-devKitC-1 |

### *3.1.2 PCB schematics*

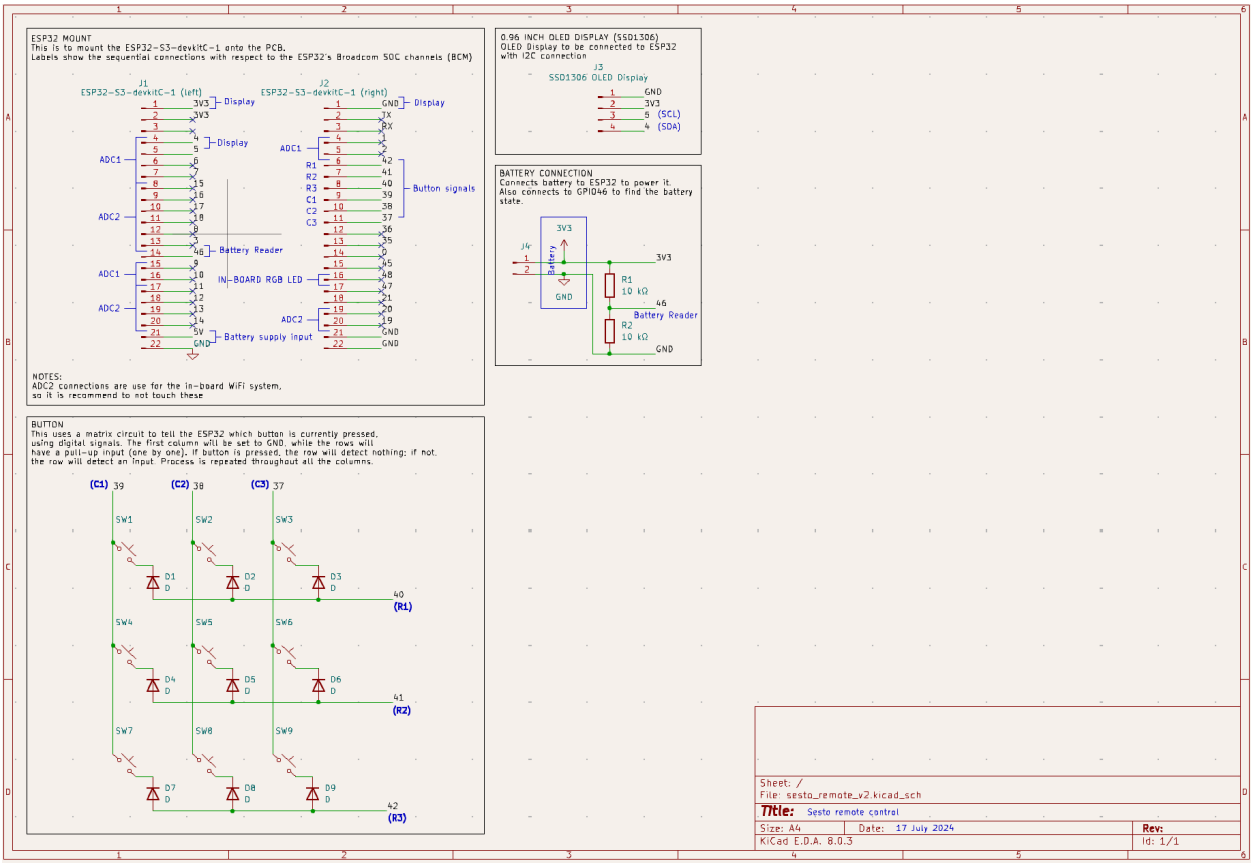


*Figure 3.1.2.1: PCB schematics*

### *3.1.3 SRC Battery Specifications*

**INR 18650 Lithium-ion Cell**

| | |
|---|---|
| Capacity | : 3000 mAh |
| Voltage Rating | : 3.7 V |
| Cell Arrangement | : 1S2P |
| Charging Method | : CC/CV Charger (Adafruit Battery Charger 259 MCP73833) |
| Total Capacity | : 6000 mAh |
| Total Voltage Rating | : 3.7 V |

### *3.1.4 SRC Drawings*

#### *3.1.4.1 Bottom Cover:*

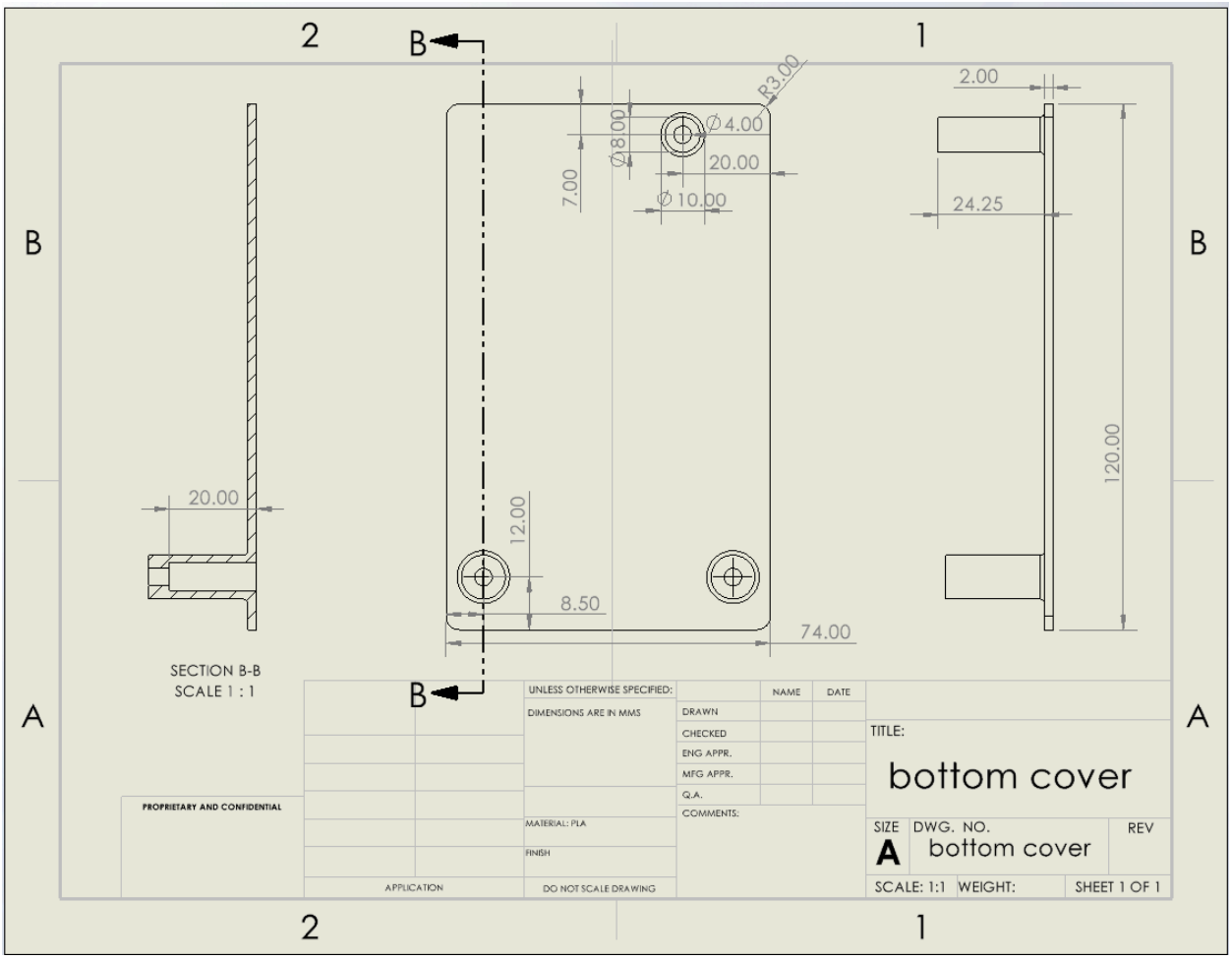


*Figure 3.1.4.1: Bottom Cover Drawing*

*3.1.4.2 Top Cover:*

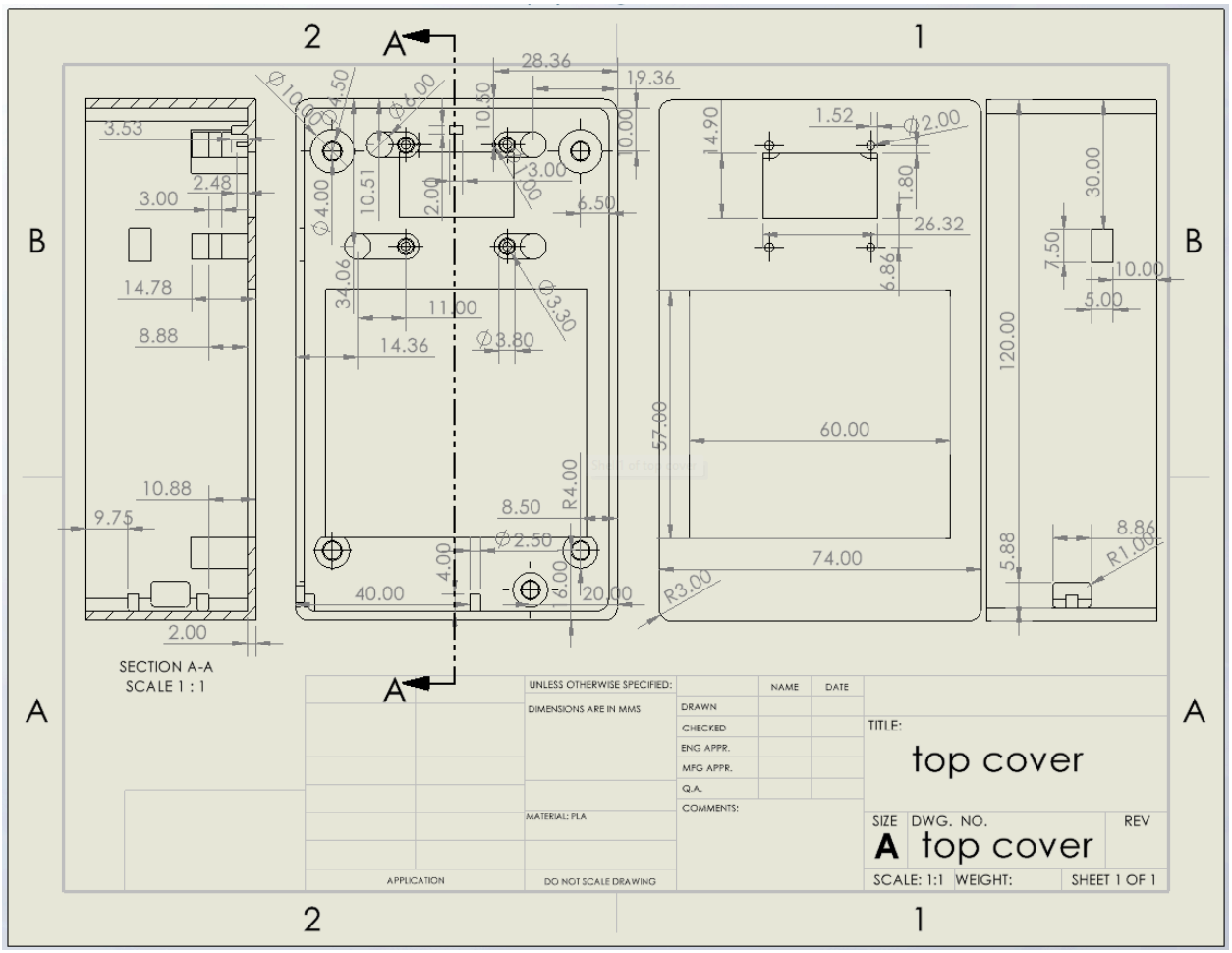


*Figure 3.1.4.2: Top Cover Drawing*

*3.1.4.3 SRC Components*

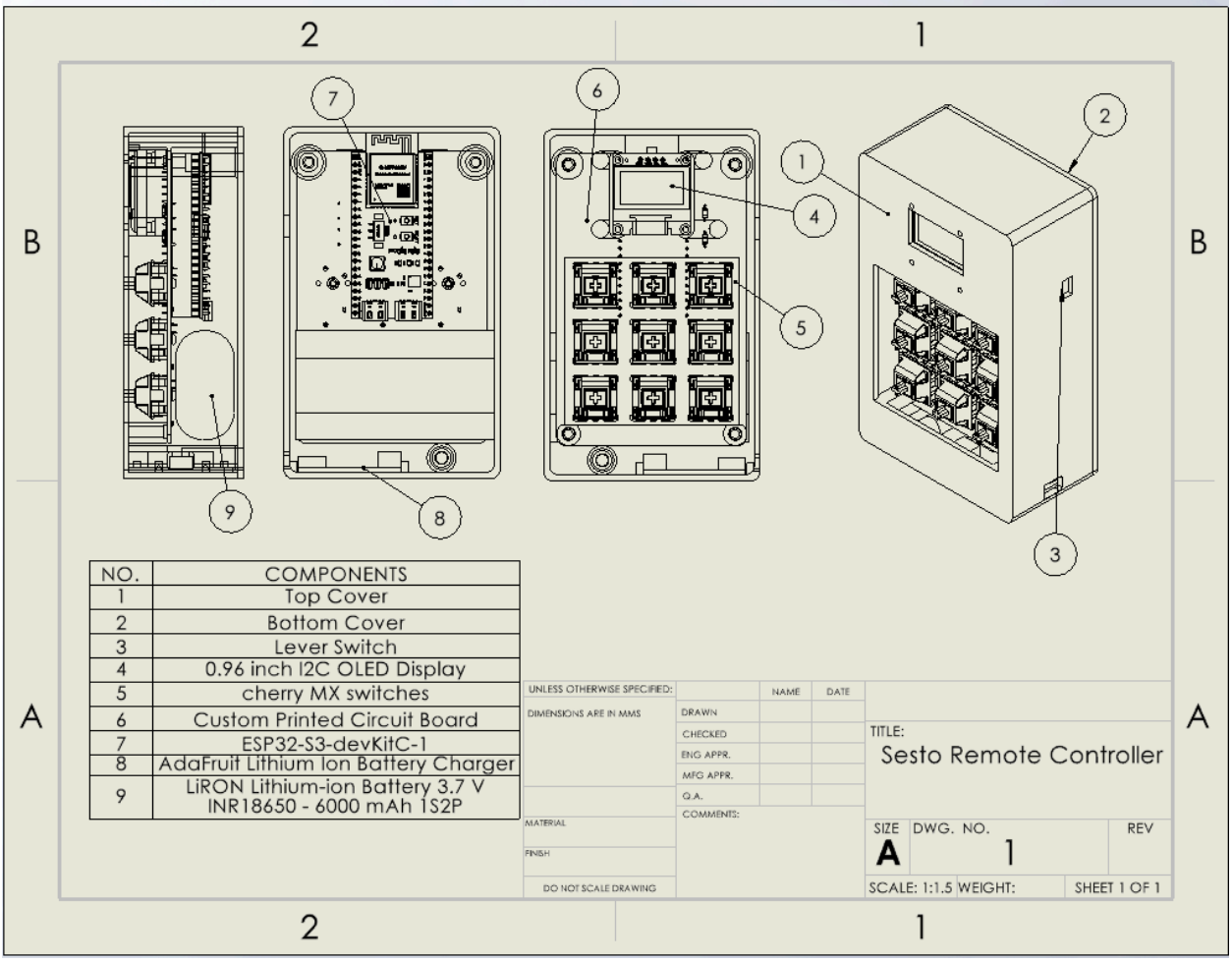


*Figure 3.1.4.3: Sesto Remote Controller Drawing*

### *3.1.5 Sesto Cabinet Opener footprint*

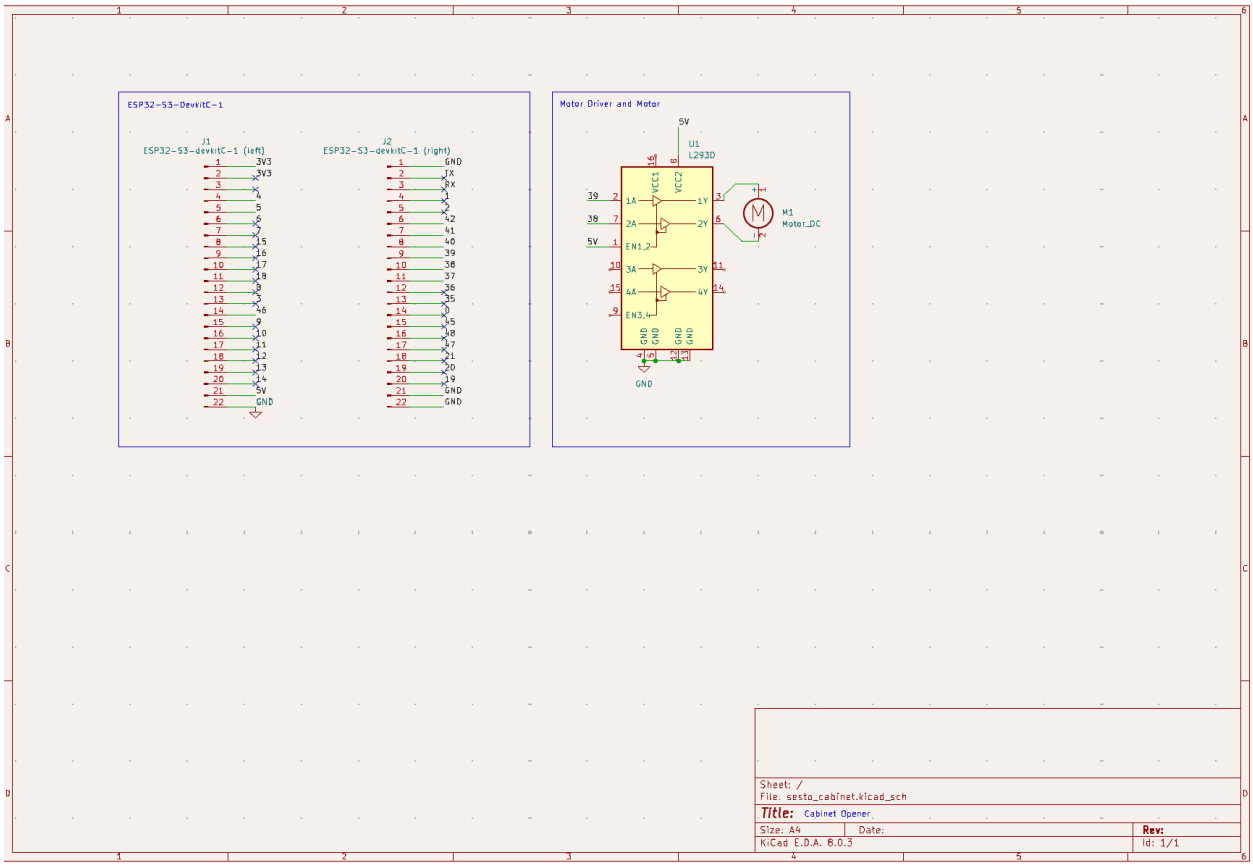


*Figure 3.1.5.1: Sesto Cabinet Opener Footprint*

*3.1.6. Sesto Cabinet Footprint*

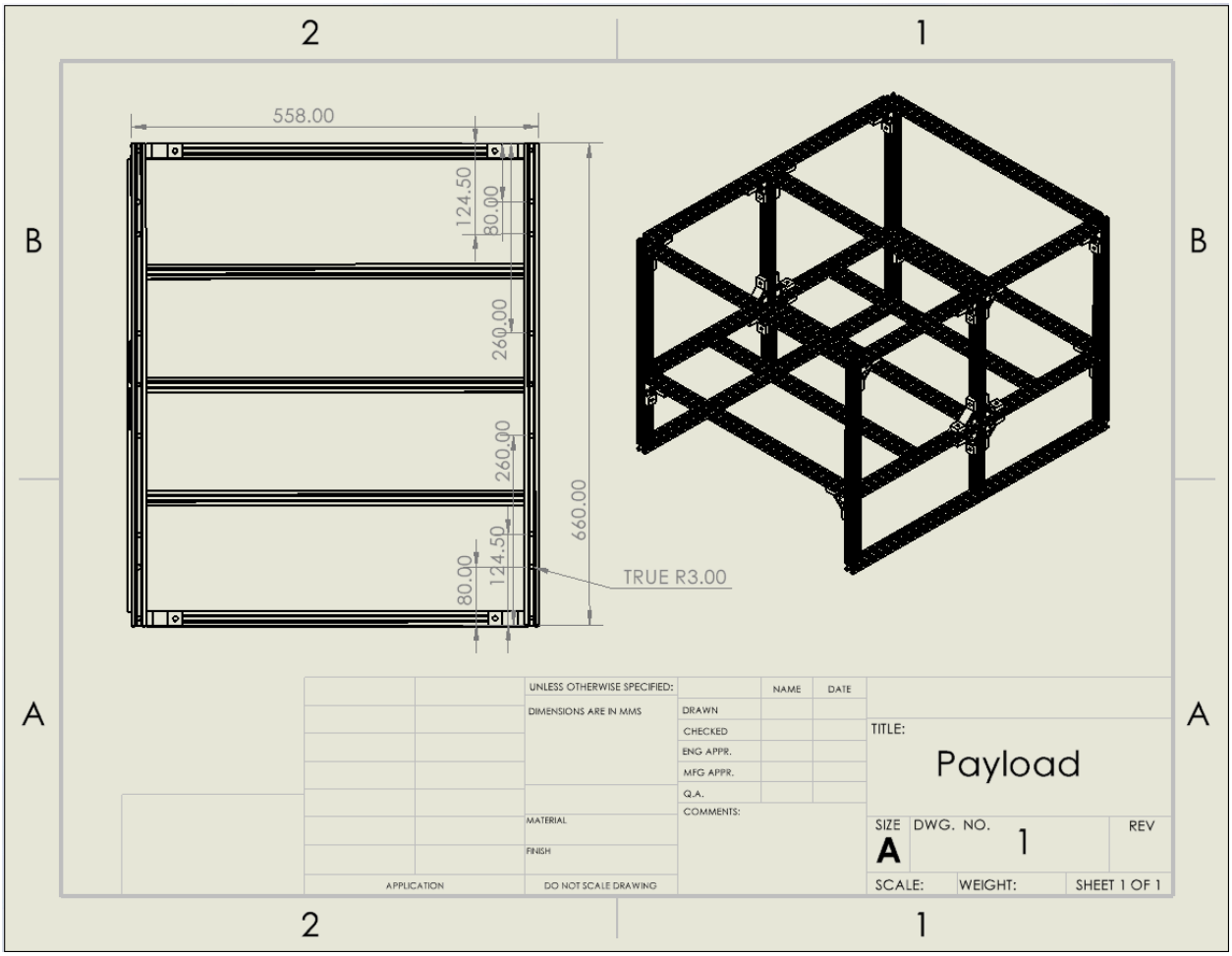


*Figure 3.1.6.1: Sesto Cabinet Footprint*

*3.1.6 GitHub link*

https://github.com/edic-nus/Sesto-Bot