March 14th, 2023

Dr. Michael Hegedus
School of Engineering Science
Simon Fraser University
Re: Design specifications.

Dear Dr. Hegedus,

Presented below is the required design specifications document for ENSC 405W which has been prepared by Company 04 – MECALAUN. The purpose of this document is to outline the design specifications for project Healthbot and provide guidance for the development team.

The following sections will be discussed:

1. Introduction: this introduces the purpose of project Healthbot and highlights the overview and objectives.
2. System overview: this section provides a detailed description of the system architecture and the different components that make up the Healthbot system.
3. Design requirements: this section outlines the design requirements for project Healthbot, including functional, mechanical, software and safety requirements.
4. Design considerations: this section covers various factors that were considered during the design phase, including safety factors, power requirements, and maintenance considerations.
5. Testing requirements: this section outlines the testing procedures that will be used to ensure that the Healthbot system meets the design requirements and operates as expected.

We believe that this document provides a comprehensive overview for the design specifications for project Healthbot, and that this document will aid the development team during the initial phases of product development.

For any questions or concerns, please contact the COO of MECALAUN Steven Borkowski at sborkows@sfu.ca.

Kind regards,

Flynn Dowey

Flynn Dowey
CEO
MECALAUN

MECALAUN

Design Specifications

Healthbot

Submitted to Dr. Michael Hegedus

March 12, 2023

Company Members

Flynn Dowey – CEO
Ngoc Quynh Anh Vo – CFO
Sammy Kaspar – CTO
Steven Borkowski – COO
Bao Nguyen – CMO
Gary Ho – CIO

Abstract

This design specification document outlines various aspects of Healthbot, a laundry delivery robot, including its mechanical, electronic, and software features. The goal of the robot is to facilitate the delivery of laundry from one location to another, thereby reducing the need for staff to handle the laundry and minimizing their contact with potentially contaminated materials. The robot's navigation system will use sensors and mapping technology to navigate through different environments and obstacles to reach the designated delivery location. The robot will also be equipped with a control system that will manage the robot's movement, communication, and safety features. The system will include sensors to detect obstacles, control the robot's speed, and avoid collisions. The robot will also be designed with a secure storage compartment to ensure the safety of customers' laundry. Additionally, the robot will be designed to have a modular structure, allowing for easy maintenance and repair of different components. All these subcomponents will go through thorough testing throughout development.

Table of Contents

Glossary of Terms 1

1 Introduction 2

1.1 Scope 2

1.2 Challenges 2

1.3 Updates 3

2 System Overview 3

3 Design Specifications 4

3.1 System Requirements 4

3.2 Software Requirements 5

3.3 Mechanical Requirements 9

3.4 Electrical Requirements 11

3.5 Sustainability Requirements 12

4 Conclusion 13

5 References 14

6 Appendix A – Alternative Design 15

6.1 Driving System 15

6.2 Microcontroller 15

7 Appendix B – Test Plan 16

List of Figures

Figure 1. Flowchart of Healthbot's operation 2
Figure 2. System diagram of Healthbot 3
Figure 3. CAD model of Healthbot 4
Figure 4. PID controller with DC motors [4]. 5
Figure 5. Schematic of electrical and control connections 7
Figure 6. ROS architecture like that of Healthbot [5] 8
Figure 7. Obstacle avoidance and navigation with ROS [6] 9
Figure 8. Dimensioned drawing of Healthbot 10
Figure 9. DC motor fixed to a motor mount [7] 10

List of Tables

Table 1. Encoding of the three development stages 4
Table 2. System requirements 4
Table 3. Software requirements 6
Table 4. Arduino pin connections 6
Table 5. Mechanical requirements 10
Table 6. Electrical requirements 11
Table 7. Mass of Healthbot components 11
Table 8. Sustainability requirements 12

Glossary of Terms

Term	Definition
PID	Proportional integral derivative control
PWM	Pulse width modulation
ROS	Robot operating system
Arduino	A microcontroller
Nvidia Jetson Nano	A microprocessor
Lidar	Light detection and ranging
Magnetic encoder	A device used to count the displacement of an actuator
Mecanum wheel	A wheel composed of angled rollers

1 Introduction

The province of BC has been undergoing a deepening nursing shortage crisis [1]. As a result, hospital staff find themselves working overtime and patient care may decline despite best efforts [1]. The field of robotics can provide a creative solution to this problem, as service robots have been working in various roles in hospitals across the globe. Their functions can range from taking vital signs, to transporting patient samples, to working reception [2]. Therefore, MECALAUN will be developing Healthbot: a laundry transport robot that will facilitate the hospital workflow by assisting nurses in patient cleaning. By working alongside the hospital staff and automating parts of the laundry process, Healthbot can help the staff focus on more vital tasks and reduce their exposure to pathogens [3]. The below flowcharts depict the process Healthbot will go through in its operation:

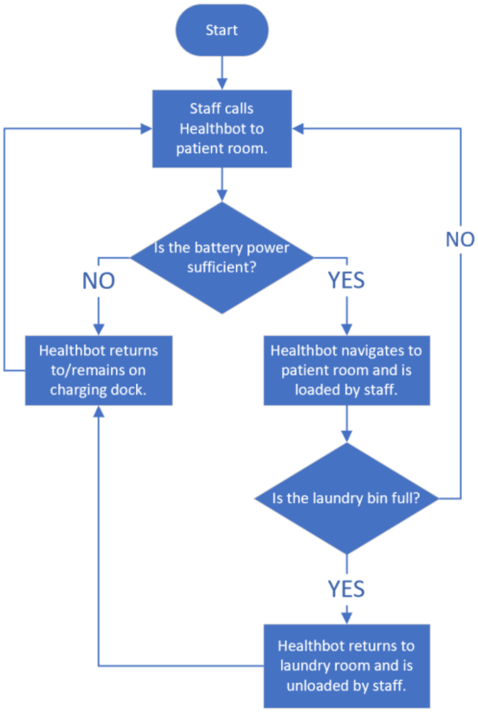

Figure 1. Flowchart of Healthbot's operation

1.1 Scope

In this document, the design choices MECALAUN have made in the development of Healthbot will be enumerated and justified. High level details will be provided as well as diagrams and schematics that will be implemented. Additionally, alternative designs will be explored.

1.2 Challenges

There are several challenges to be overcome in the development of this product, as it will require integration of software with electronic and mechanical parts. The greatest hurdles presented by the following design requirements include:

- Latency in communication between the main terminal, Nvidia Jetson Nano and Arduino boards.
- Runtime of the battery powering the motors.
- Structural integrity of load bearing parts of the frame.
- Reliability of obstacle detection to prevent collisions.

1.3 Updates

Throughout the process of designing Healthbot, it has been integral addressing feedback from end users and experts. The following considerations have been made according to these comments:

- Healthbot should assist staff during patient cleaning times rather than replace a function in the hospital to improve healthcare workflow.
- MECALAUN must compare Healthbot with similar solutions in the market to have a competitive price and match the hospital budget.
- A single Arduino board can be used as opposed to two Arduinos, as more components can result in a greater risk of failure and latency between the devices.
- The Nvidia Jetson Nano can be used as opposed to a Raspberry Pi for more processing power.

2 System Overview

Healthbot consists of a platform with mecanum wheels, a removable laundry bin with a lid, and an onboard Arduino microcontroller for autonomous and manual control. The drivetrain consists of four 96 mm mecanum wheels, each of which is controlled by a separate DC motor. These wheels allow the robot to move in any direction, making it highly maneuverable. The DC motors in the system are controlled by two motor drivers, each of which can control two DC motors simultaneously. The motor drivers receive signals from the microcontroller and use these signals to determine the speed and direction of the motors. The microcontroller communicates with a host computer in the proof-of-concept stage to allow for manual control. The robot is powered by a rechargeable battery, and the status of the battery is indicated by LED indicators on the robot. The robot is also equipped with safety features such as emergency stop buttons, Lidar, and ultrasonic sensors to prevent collisions. The robot's base consists of two wooden platforms that are parallel to each other. The bottom plate is used to house the electronic components, while the top plate supports the laundry bin. These plates are separated by four aluminum rails spaced evenly between them. This design ensures that the robot is stable and that the weight of the laundry is evenly distributed across the base. A Nvidia Jetson Nano will be used to run ROS in the prototyping stage, which allows different components of the robot to communicate and work with each other in a coordinated way. Manual control will be implemented in the proof-of-concept stage so that the robot can follow user’s instructions through a controller and navigate through doors and hallway. For the prototyping stage, autonomous control will be added so that the robot can navigate on its own. Below is a system diagram illustrating the functionality of Healthbot.

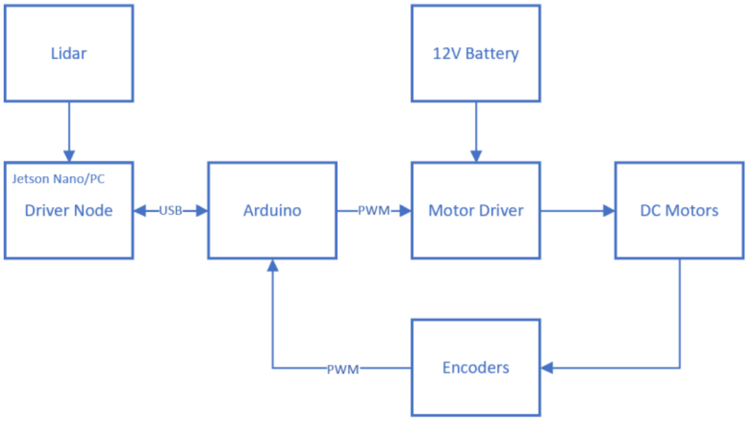

Figure 2. System diagram of Healthbot

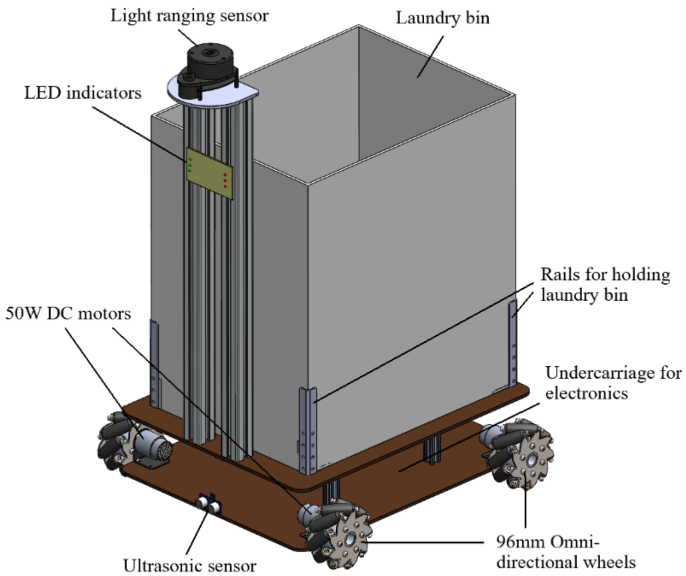

Figure 3. CAD model of Healthbot

3 Design Specifications

The design specifications are categorized into the three development stages below.

Encoding	Product Development Stage
A	Proof-of-concept
B	Engineering Prototype
C	Production Version

Table 1. Encoding of the three development stages

3.1 System Requirements

The system specifications here describe how the integral functions of Healthbot will be implemented.

ID	Encoding	Description	Corresponding Req. Spec.
Des. 1.1	A	The robot will be dimensioned such that it can navigate through doors, between rooms and halls.	Req. 1.1
Des. 1.2	A	The robot has a covered laundry bin mounted on it to prevent the spread of pathogens.	Req. 1.2
Des. 1.3	A	The robot uses PID speed control to navigate safely depending on the load it is carrying.	Req. 1.3
Des. 1.4	B	The robot uses a Lidar and ultrasonic sensor to map out surroundings and detect obstacles.	Req 1.4

Table 2. System requirements

For the robot to be able to scan and detect obstacles in its surroundings, a Lidar will be installed at the highest point of the robot. This will allow it to scan the surrounding area from every angle, providing a 360-degree view. To support the Lidar, it will be placed on a small base that is supported by two aluminum rails. This ensures the Lidar has a stable base and won't move around while scanning the area.

In addition to the Lidar, an ultrasonic sensor will also be installed on the Healthbot. The sensor will be placed at the bottom of the robot, as this location allows it to detect obstacles on the ground and assist the Lidar in object detection. The placement of the ultrasonic sensor at the bottom also helps to avoid interference from objects or people that may outside the sensor's range of detection.

A PID controller will be used to control the motors speeds given its adaptable nature and control properties. The controller can comphesate for undesired changes in the speed of the robot by adjusting the speed and direction based on real-time feedback from the magnetic encoders mounted on inside the dc motors. PID controllers also have the ability to prevent overshoot and instability. Consider the following block diagram for the architecture of a PID controller.

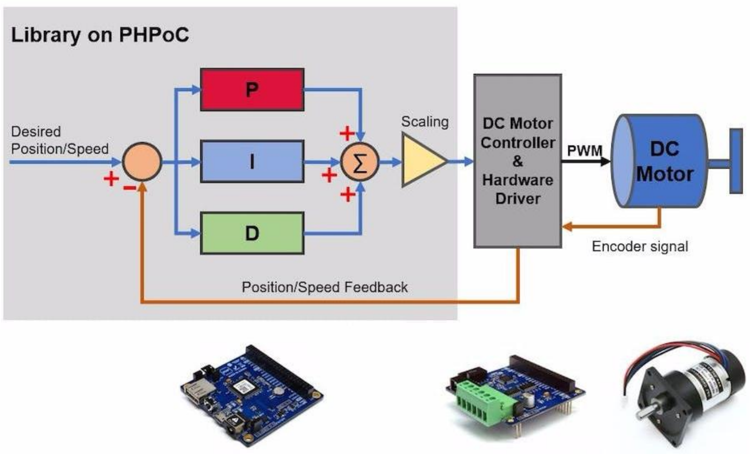

Figure 4. PID controller with DC motors [4].

3.2 Software Requirements

The following table lists the software requirements that will be used in the implementation of Healthbot.

ID	Encoding	Description	Corresponding Requirement Spec.
Des. 2.1	A	The software should be compatible with ROS and should be able to communicate between the host machine and Arduino.	Req. 2.1, Req. 2.2
Des. 2.2	A	The software should enable real-time control of the robot's movements and actions.	Req. 2.1, Req. 2.2
Des. 2.3	A	The software should be able to collect and interpret data from the robot's sensors and use it in a meaningful way.	Req. 2.2
Des. 2.4	A	The software should control the speeds of the motors that enable the robot to move.	Req. 2.2

Des. 2.5	B	The software should provide navigation functionality such as path planning and obstacle avoidance.	Req. 2.2, Req. 2.3
Des. 2.6	B	The software should allow for communication between external devices such as a remote controller.	Req. 2.2
Des. 2.7	A	The software should prioritize safety, ensuring that the robot operates within safe parameters and does not pose risk to the public, the user, or objects within the robot’s environment.	Req. 2.3

Table 3. Software requirements

Regarding Des. 2.1, there are several reasons why software should be compatible with the robot’s operating system and be able to communicate with a microcontroller. ROS is a widely used interface within the robotics community and has become the standard for developing robotic software. ROS also provides a framework for developing robot applications including communication between nodes, sensor integration and control. Making software compatible with ROS will allow developers to use the existing tools and libraries ROS has to offer.

Arduino is a popular microcontroller which is widely used for building robots and DIY projects. Allowing the software to be compatible with Arduino will make it easier to interface with sensors, actuators, and control the robot’s behavior. The following table shows the pin connections of the Arduino Leonardo on board Healthbot:

Pin	Explanation
0, 1, 2, 3	Interrupt pins to read encoders position, connected to encoders outputs
5, 6, 9, 10	PWM pins to drive the motors, connected to motor driver inputs
Gnd	Connected to motor driver Gnd and encoder Gnd
5V	Connected to Motor driver Enable pin and encoder V_{cc}

Table 4. Arduino pin connections

The schematic below depicts an overview of the connections of the system in its proof-of-concept stage. It should be noted that Healthbot will be using an additional Arduino Leonardo and motor controller, and two additional motors.

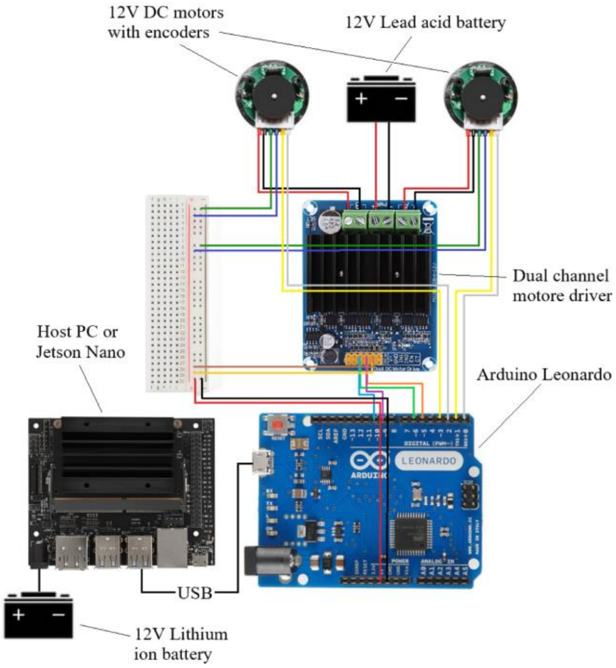

Figure 5. Schematic of electrical and control connections

By enabling communication between the host machine and Arduino, developers have the flexibility to develop software on a more powerful computer which they can then deploy on the Arduino for real time control. This enables developers to take advantage of the processing power of the host machine while still maintaining the low latency and real-time capabilities of the Arduino. Moreover, by proving this interface between the Arduino and host machine, developers can focus on designing the core functionality and logic without needing to worry about the low-level details of communication protocols. In turn, this decreases development time and increases visibility of bugs and errors.

Regarding Des. 2.2, real-time control of a robot’s movements and actions is essential for successful robotic applications. Real-time control is crucial for ensuring the safety of the robot and its surroundings. In a hospital environment, the robots’ movements need to be precisely controlled to avoid collisions with people or other objects in the robot’s environment. Real-time control allows for immediate reactions to unpredictable situations which minimizes the risks of accidents.

Regarding Des. 2.3, the ability for software to collect and interpret data from the robot’s sensors is required to enable the robot to perform autonomous tasks accurately. The robot’s sensors are designed to capture data from the environment which allows for informed decisions about the robot’s actions. Without a software system to interpret this data, the robot would not be able to make use of these sensors, making them useless. Refer to the following illustration which depicts a high-level representation of the ROS architecture.

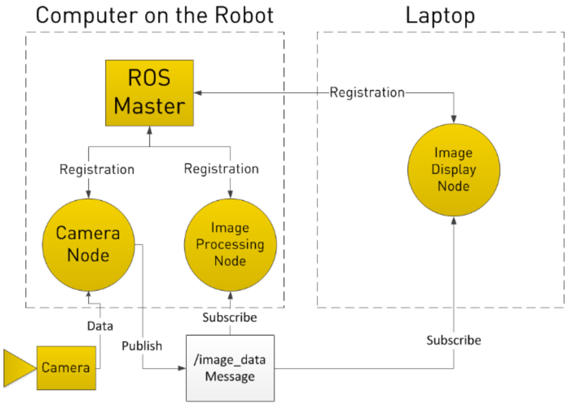

Figure 6. ROS architecture like that of Healthbot [5]

The software which renders the collected data should be designed to handle the specific data generated by each specific sensor, such as Lidar and ultrasonic sensors. This information would then be used to guide the robots' movements, such as avoiding obstacles or navigating to a specific location.

Regarding Des. 2.4, controlling the speeds of the robot's motors is yet another essential building block required in a robotic system. The speed of the motors determines the robot's movement. The software that controls the motors should be designed to regulate the speeds and directions of the wheels based on feedback from the sensors. If the robot's motors are not controlled then its motion maybe be erratic, causing the robot to collide with objects, fail to complete its programmed tasks or potentially cause harm to nearby objects or humans. Finally, the software should control the motors within a safe operational limit to prevent overheating, burnout, or any other potential mechanical failures. The following is a sample pseudocode used by the Arduino to drive the motors:

```
loop(){

      curr_time = GET_TIME_ms();
      if(curr_time - prev_time >= 10){
            goal_vel = GET_GOAL_SPEED();
            while(!interrupt){}
            goal_displacement = goal_vel * (curr_time - prev_time);
            real_displacement = curr_pos - prev_pos;

            prev_time = curr_time;
            prev_pos = curr_pos;

            error = real_disp - goal_disp;
            pwm_sig = PID_CONTROLLER(error, goal_vel);
            WRITE_TO_OUTPUT(pwm_sig);
      }
}
```

Regarding Des. 2.5, the ability for a robot to navigate and avoid obstacles is required to demonstrate autonomous behavior where human intervention is limited. In such an environment, the robot should be able to operate independently and maneuver safely without human intervention. Therefore, the software should be designed to schedule a path that the robot must follow within certain tolerances based on the current environment. The software should also be designed to detect objects within the robot's path of motion to prevent collisions and send the necessary instructions to maneuver around them. Refer to the following illustration for the ROS architecture being used.

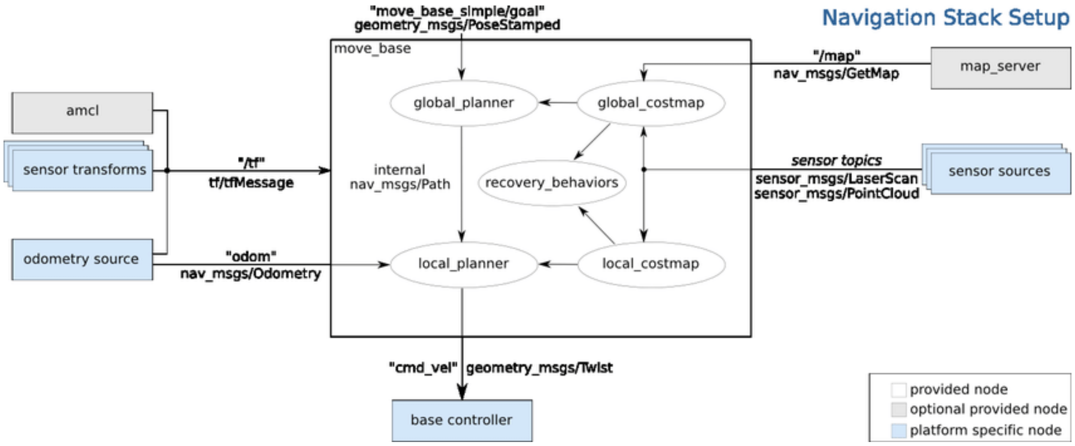

Figure 7. Obstacle avoidance and navigation with ROS [6]

Regarding Des. 2.6, allowing the robot's operating system to communicate with external devices is to allow for remote control of the robot. Robots can malfunction, and if not designed for safety the result could be hazardous to the robot, surrounding environment or humans. Thus, the software should be designed to detect abnormalities and perform the correct steps to minimize or prevent the hazard.

Regarding Des. 2.7, the software that controls the robot should be designed to prioritize safety, ensuring that the robot operates within safe limits and does not pose risk to the public, the user, or its surrounding environment.

3.3 Mechanical Requirements

To ensure the safety of patients and healthcare professionals, a Healthbot designed for hospital use should have a meticulously crafted mechanical design. The placement of sensors should be well-thought-out, enabling them to detect obstacles and avoid causing any damage to medical equipment. Additionally, the robot's design should prioritize user experience, being friendly, easy to use, operate and receive message or signal sent from the robot.

The following table lists the mechanical requirements that will be used in the implementation of Healthbot.

ID	Encoding	Description	**Corresponding Req. Spec.**
Des. 3.1		The chassis consists of a wooden platform with 37mm DC motor mounts at each corner. The dimensions of the platforming composing the chassis will be 40x50x0.6 cm^3.	Req. 3.1
Des. 3.2		A wooden platform is mounted above the chassis on four aluminum rails at each corner. A cloth laundry bin with a lid will rest on this platform.	Req. 3.2

Des. 3.3		The laundry bin should be compatible with leak-proof laundry bags.	Req. 3.3
Des. 3.4		The robot will be driven by a 4 omnidirectional wheel system.	Req. 3.4
Des. 3.5		4 L-shaped shaped wooden slots will protrude from the upper platform to serve as a guide for removing and inserting the laundry bin.	Req. 3.5
Des. 3.6		The chassis of the robot is designed to have rounded edges and without any protruding sharp edges.	Req. 3.6

Table 5. Mechanical requirements

Des. 3.1-3.2 seek to ensure the Healthbot's safety while in operation, and describe the mechanical design of the base consisting of two wooden platforms. The lower platform will hold all the electrical components of the robot, while the upper platform will serve as a holder for the laundry basket. The edges and corners of the robot will be smoothed and rounded to prevent any injury or damage to the surroundings. Four 95.25mm aluminum rails will be used to connect the two platforms.

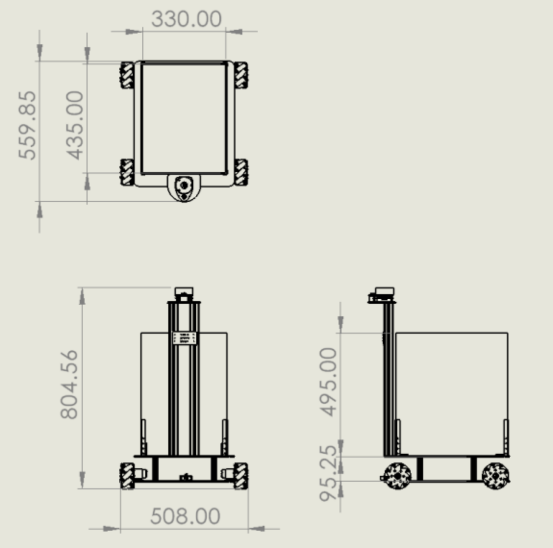

Figure 8. Dimensioned drawing of Healthbot

As can be seen from the figure above, the Lidar will be elevated 80.4cm above the ground to achieve a comprehensive view of the surroundings while minimizing interference from lower objects. The ultrasound sensor will be placed on the bottom platform just above the ground to facilitate the detection of ground-level obstacles and supplement the Lidar’s detection abilities.

Figure 9. DC motor fixed to a motor mount [7]

According to the figure above, four motors will be fixed onto the lower wooden board using mounting bracket holders with a 6mm hex coupling. Following this, the selected mecanum wheels can be connected directly to their respective motor shafts since they also have a coupling diameter of 6mm.

Regarding Des. 3.4, mecanum wheels have been selected due to the mobility they afford [8].

3.4 Electrical Requirements

The following table lists the electrical requirements that will be used in the implementation of Healthbot.

ID	Encoding	Description	Corresponding Req. Spec.
Des. 4.1	B	The Jetson Nano will be powered by a 12V Lithium-ion rechargeable battery.	Req. 4.1, Req. 4.2
Des. 4.2	B	The Arduino board(s) will be powered by a connection to the Jetson Nano.	Req. 4.1
Des. 4.3	A	The driving system will use four 12V motors operated by powered by 12V lead acid battery through a pair of motor controllers.	Req. 4.1, Req. 4.2
Des. 4.4	A	The electronics are secured in an undercarriage between the two wooden platforms of the robot.	Req. 4.4

Table 6. Electrical requirements

The selection of motors must be made based on the load (including laundry and laundry bin) that Healthbot must carry, which, in compliance with Req. 3.1, should be 17lbs at maximum.

The weight of the robot when carrying a maximum load will be approximated with following values:

Item	Mass (kg)	Quantity	Total Mass (kg)
12V DC motors	0.21	4	0.84
Wooden boards	1.10	2	2.20
Aluminum Rails	0.07	6	0.42
Laundry bin	1.24	1	1.24
Omnidirectional wheels	0.15	4	0.60
Motor controllers	0.04	2	0.08
Motor mounting bracket	0.04	4	0.16
Arduino & Nvidia Jetson	0.08	1	0.08
Lithium-ion battery	0.38	1	0.38
12V Lead acid battery	2.00	1	2.00
Lidar sensor	0.32	1	0.32
Maximum laundry load	-	-	6.47
TOTAL:			14.79

Table 7. Mass of Healthbot components

Given the values above, the calculations below were used to decide which motors would be suitable for Healthbot. The constants that will be used are:

Mass of the robot: $m = 14.79\ kg$

Coefficient of static friction: $\mu = 0.6$ [9]

Radius of wheel: $r_w = 0.048\ m$

Gravity constant: $g = 9.81\ N/kg$

Firstly, the force of friction between the floor and wheel that must be overcome by the motor to start moving is calculated by:

$$F_{fr} = \frac{mg}{4}\mu = 21.76\ N$$

The factor of ¼ in the above equation comes from the fact that the weight of the robot will be divided among 4 wheels. The torque required by the motor can be calculated by:

$$\tau_m = F_{fr} r_w = 1.04\ Nm = 10.4\ kgcm$$

Therefore, the 12V motors that were selected have a rated current of 1.5A with a load of 9.5 kgcm and a maximum torque of 38 kgcm [10].

3.5 Sustainability Requirements

The following table lists the sustainability requirements that will be used in the implementation of Healthbot.

ID	Encoding	Description	Corresponding Requirement Spec.
Des. 5.1	B	The robot implements ROS for intelligent path planning	Req. 5.1
Des. 5.2	A	The frame of the robot is made of wood and aluminum.	Req. 5.2
Des. 5.3	A	The robot is designed in a modular fashion, allowing for specific parts to be replaced or upgraded.	Req. 5.3
Des. 5.4	A	The laundry bin atop the robot is made of cloth and bamboo.	Req. 5.4
Des. 5.5	A	The batteries used for the electronics and motors are rechargeable.	Req. 5.5
Des. 5.6	A	The robot uses reprogrammable microcontrollers and microprocessors for control.	Req. 5.6

Table 8. Sustainability requirements

To align with the cradle-to-cradle (C2C) concept of sustainability, the Healthbot has been carefully designed with several sustainable materials. The robot frame is mainly constructed of wood and aluminum, both of which are renewable and recyclable materials. The use of wood as a biodegradable material has a minimal environmental impact, and aluminum’s ability to be reused or recycled at the end of its life cycle would help decrease waste production. These materials are chosen over steel due to their light weight, which helps reduce energy consumption and increase the robot’s efficiency, but still guarantee a stable and durable frame. The laundry bin of the robot is also made of cloth and bamboo, which are also environmentally friendly materials. To further expand its lifespan, the robot is designed in a modular manner, allowing for the replacement or upgrade of specific components. The use of rechargeable batteries for electronics and motors also increases sustainability and minimizes waste. In addition to the sustainable materials and components used in the robot’s design, Healthbot also incorporates reprogrammable microcontrollers and microprocessors for control. This allows for software updates and improvements to be made without replacing any hardware, which helps reduce electronic waste and promotes sustainability. By taking sustainability into consideration, the Healthbot is expected to not only fulfill its intended purpose but also promote responsible use of resources and minimize harm to the environment.

4 Conclusion

The design specifications for Healthbot outlined in this document entail the choices that have been made to make Healthbot a safe, effective, and sustainable solution to facilitating the hospital workflow and minimize the contact with potentially contaminated materials. This document has covered various aspects of the robot, including its structural, mechanical, electronic, and software features. By considering the feedback from healthcare professionals and experts in the field, MECALAUN has created a comprehensive set of design considerations and testing requirements to ensure the success of the project. Throughout the design process, several logistical and technical difficulties presented themselves. Consulting with potential end users and experts was essential in honing the requirements enumerated here to address these challenges to make Healthbot a viable product.

5 References

[1] "Nurse staffing crisis," BC Nurses' Union, 24 November 2022. [Online]. Available: https://www.bcnu.org/news-and-events/update-magazine/2022/spring-2022/nurse-staffing-crisis. [Accessed 13 March 2023].

[2] H. Ahn, M. Lee and B. MacDonald, "Healthcare robot systems for a hospital environment: CareBot and ReceptionBot," in *2015 24th IEEE International Symposium on Robot and Human Interactive Communication*, Kobe, Japan, 2015.

[3] L. Sehulster and R. Chinn, "Guidelines for Environmental Infection Control in Health-Care Facilities," *Infection Control*, 2003.

[4] "PID controller, auto-tuning library and example for DC Motor," Hackster.io, 12 July 2018. [Online]. Available: https://www.hackster.io/phpoc_man/pid-controller-auto-tuning-library-and-example-for-dc-motor-bc028f. [Accessed 3 March 2023].

[5] "Intro to ROS," Clearpath Robotics, 2015. [Online]. Available: https://www.clearpathrobotics.com/assets/guides/noetic/ros/Intro%20to%20the%20Robot%20Operating%20System.html. [Accessed 13 March 2023].

[6] G. Granosik, Grzegorz, Andrzejczak, Kacper, Kujawinski, Mateusz, Bonecki, Rafal, Chlebowicz, Łukasz, Krysztofiak, Blazej, Mirecki, Konrad, Gawryszewski and Marek, "USING ROBOT OPERATING SYSTEM FOR AUTONOMOUS CONTROL OF ROBOTS IN EUROBOT, ERC AND ROBOTOUR COMPETITIONS," in *Acta Polytechnia CTU*, 2016.

[7] "DC Gear Motor With Magnetic Linear Encoder," Lampatronics, [Online]. Available: https://lampatronics.com/product/dc-gear-motor-with-magnetic-linear-encoder-jga25-371-210rpm-6v/. [Accessed 14 March 2023].

[8] E. Malayjerdi, H. Kalani and M. Malayjerdi, "Self-Tuning Fuzzy PID Control of a Four-Mecanum Wheel Omni-directional Mobile Platform," in *Iranian Conference on Electrical Engineering*, Mashhad, Iran, 2018.

[9] E. Matsinos, "Modelling of the motion of a Mecanum-wheeled vehicle," 2012.

[10] DFRobot, "12V DC_Motor 122 rpm w Encoder," [Online]. Available: https://wiki.dfrobot.com/12V_DC_Motor_122_rpm_w_Encoder_SKU__FIT0403#target_0. [Accessed 13 March 2023].

6 Appendix A – Alternative Design

The following appendix details other designs MECALAUN may use that would also meet user needs.

6.1 Driving System

When selecting the drive system for Healthbot, MECALAUN mainly considered two designs, mecanum wheels and differential drive. Mecanum wheels facilitate omnidirectional movement with no need for turning, enabling the robot to move forward, backward, diagonally, and sideways. In contrast, differential drive utilizes two independent wheels to control the robot's direction and speed. By adjusting the speed and direction of each motor separately, the vehicle can move and turn in any direction. For example, to turn right, the motor on the right side of the vehicle would be slowed down while the motor on the left side moves faster. To turn in place, the two motors would turn in opposite directions. Castor wheels would need to be equipped to stabilize the robot.

Each of these designs has its own set of benefits and drawbacks. Mecanum wheels are ideal for environments where the robot needs to move around quickly and without the need for turning. These wheels perform best on smooth surfaces such as hospital floors. However, their more complex design and higher cost may require more maintenance and greater expenses. In addition, requiring each of the 4 motors to turn independently means we'll need to read from each of the 4 motor encoders to accurately adjust each wheel as needed.

In contrast, differential drive is a cost-effective and more straightforward option and requires less complex motor control algorithms. However, it may not be as efficient in navigating uneven surfaces or moving around confined spaces.

6.2 Microcontroller

When selecting the control system for Healthbot, two Arduino Leonardo's were selected. This was done to accommodate the encoders affixed to the selected motors, as they each require two interrupt pins, whereas the Leonardo has four. However, more research will be done into whether a single Arduino Mega board can be used by converting its digital pins to interrupt pins. Using a single Arduino board as opposed to multiple could reduce latency, lower the risk of system failure, and simplify the design.

7 Appendix B – Test Plan

This appendix details the testing procedure followed in the proof-of-concept presentation to provide to the user assurance of an effective product.

MECALAUN - Healthbot Verification Test Sheet

Tester: ___________________ Tester Signature: ____________________ Date: ______________________

Mechanical Parts

Test Plan

1. Perform this procedure close to a doorway that serves as the goal in this test.
2. Tester will start a ROS controller node on the host machine.
3. Tester will input commands to the robot to put it through the following routine:
   - Drive forward towards the doorway.
   - Adjust by turning towards the doorway.
   - Enter through the doorway.
4. Tester will insert items into the laundry bin up to 6.47kg.
5. Tester will demonstrate driving the robot under a load.
6. Tester will stop the robot and remove the laundry bin from the robot.
7. Tester will insert the laundry bin on top of the robot.

1. Driving system Robot drives forward. ☐ Yes ☐ No	Comments:
Robot can turn using omnidirectional wheels. ☐ Yes ☐ No	Comments:
Robot can drive through a doorway. ☐ Yes ☐ No	Comments:
2. Upper Platform Upper platform can support 17lb load, including the laundry bin. ☐ Yes ☐ No	Comments:
3. Laundry Bin and guiding rail User can insert items into laundry bin, remove the bin from the platform, and reinsert the bin on the platform. ☐ Yes ☐ No	Comments:
Users can remove the bin from the platform and reinsert the bin on the platform. ☐ Yes ☐ No	Comments:

Electrical Parts

Test Plan

1. Perform this procedure with a DMM.

2. Ensure switch affixed to terminal of lead acid battery is in the ON position. **3.** Measure the voltage of the fully charged lead acid battery. **4.** Flick the switch on the terminal of the lead acid battery and measure its voltage. **5.** Measure the voltage of the fully charged Lithium-Ion battery.	
1. Lead Acid Battery Battery outputs > 12V. ☐ Yes ☐ No	Comments:
Switch affixed to battery terminal enables and disables power to the motor drivers. ☐ Yes ☐ No	Comments:
2. Lithium Ion Battery Battery outputs > 12V. ☐ Yes ☐ No	Comments:
Software Parts	
Test Plan **1.** Perform this procedure in the terminal of the host machine and in the Arduino IDE. **2.** Start a ROS controller node on the host computer and input sample commands. **3.** Switch to the Arduino IDE and see the data being printed that is collected by the motor encoders.	
1. ROS Robot responds to commands such as driving forward and turning. ☐ Yes ☐ No	Comments:
2. Arduino Arduino reads data from the motor encoders. ☐ Yes ☐ No	Comments: