LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title
CS5053NI/CC5068NI– Cloud Computing & IoT

"Controlling Wheelchair through Remote Access/Voice Command"

Assessment Type
50% Group Coursework

Semester
2023 Spring/Autumn

Group members

London Met ID	Student Name
22067501	Ankit Karna (Leader)
22068122	Bipul Shrestha
22068095	Nishreyta Lamsal
22068051	Shuvhanjal Adhikari

Assignment Due Date: 1/12/2024
Assignment Submission Date: 1/11/2024
Submitted to: Mr. Sugat Man Shakya
Word Count: 3026

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

ACKNOWLEDGEMENT

We would like to express our heartfelt gratitude to Islington College for offering an exceptional environment that has fostered our group's learning process over the course of our IoT project. The resources and support offered by the college have been instrumental in our successful journey.

We extend our heartfelt thanks to our module leader, Mr. Sugat Man Shakya. His leadership and steadfast assistance have been invaluable in helping our team navigate the complexities of our Internet of Things project. His dedication to creating a positive learning environment has motivated us to pursue excellence.

Additionally, we would like to thank the other module teachers for their invaluable contributions to our comprehension of a variety of project-related topics. Their enthusiasm and knowledge have expanded our horizons and enhanced our education. We would also like to thank the members of the group for their hard work and contribution in successfully completing the project.

We thank Mr. Sugat Man Shakya, Islington College, and all the module professors for their combined efforts in helping to shape our understanding of IoT and assisting us in successfully completing our project. We are appreciative of the chances given and the support we have received, as they have been essential for our success in completion of the project.

ABSTRACT

The goal of this Internet of Things-based mobility assistance project is to create a wheelchair prototype that can be operated by voice or remote controls, making it suitable for anyone with limited physical mobility. Using remote access algorithms and sensor technology, the intuitive interface decodes movements. Through the utilization of the Internet of Things, the wheelchair has the potential to transform into an intelligent and adaptable device that offers a smooth and user-friendly interaction. The initiative emphasizes the synergy between technology and accessibility, as well as how IoT can greatly improve the lives of people who have mobility issues. It investigates the revolutionary effect of technology on boosting user independence through empowering solutions and encouraging variety.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

1. Introduction

The internet of things, or IoT, is a network of interrelated devices that connect and exchange data with other IoT devices and the cloud. IoT devices are typically embedded with technology such as sensors and software and can include mechanical and digital machines and consumer objects. (Gillis, 2023)

Our project, "Smart Wheelchair is a modern study in the field of assistive technology that straddles the boundary between human-centered design and technological innovation. Our research, which aims to improve mobility for individuals with physical limitations, offers a novel solution that facilitates wheelchair control via voice control and remote access. It has different features like remote/voice access with obstacle detection. Also, it has feature of gesture control which helps the movement of wheelchair through the gyroscope of connected device.

1.1 Current Scenario

Individuals with lower limb impairments have several obstacles, such as limited accessibility in public areas and transit because necessary elements like ramps are not there. Discrimination and social isolation are made worse by social stigma and mental obstacles.

Unreachable recreational places are made more difficult by limited access to healthcare and expensive assistance technologies, which provide health and financial issues. Comprehensive initiatives are required to improve accessibility, increase consciousness, fight prejudice, and advance inclusivity in all societal contexts.

1.2 Problem Statement and Project as a Solution

Problem Statement:

The project provides a game-changing solution for people with disabilities in their legs. Using remote or voice control, the initiative solves accessibility issues with public transit and locations by providing wheelchair control.

Project as a Solution:

This device enhances independence and inclusivity, removing social obstacles and improving physical mobility. Its workplace navigation features could positively impact employment opportunities, marking a significant step toward enhancing the lives of individuals with leg-related disabilities.

1.3 Aims and Objectives

Aim:

The main aim of this project is to create an adaptable and useful wheelchair prototype that can be operated remotely via Bluetooth and Voice control, has an ultrasonic sensor for avoiding obstacles, and can be used as a thorough teaching tool in the field of IoT.

Objectives:

• Give a clear and understandable overview of the system, highlighting its main features and parts.

• Demonstrating the entire development process in detail, from planning to carrying out to testing iteratively.

• Talk about possible upgrades in the future, emphasizing scalability, sustainability, and extra features.

• Summarized the main conclusions, achievements, and thoughts regarding the development of the wheelchair prototype before closing.

2. Background

2.1 System Overview

The working prototype consists of a multipurpose robotic wheelchair with obstacle avoidance features that can be controlled remotely through Bluetooth and Voice commands. An Arduino microcontroller, which controls the integration of different hardware components, is the system's central component. Four DC motors make up the mechanism for propulsion, and each on is managed by an AFMotor module to provide exact movement control.

The wheelchair can detect obstacles and navigates its surroundings on its own thanks to the ultrasonic sensor mounted above a servo motor. With the help of a special mobile app, an HC-05 Bluetooth module creates a seamless connection that gives users-controlled capabilities. Another way to direct the wheelchair is by voice commands, which are recognized by a basic voice control system.

It also has an extra feature of Gesture Control which can be controlled by connecting the Bluetooth with the application. When the device is gestured-moved right and left, the wheelchair also moves respectively.

2.2 Design Diagram

- **System Architecture**

A system's architecture reflects its intended use as well as its interactions with external systems and other systems. It explains how every part of the system is connected to every other part of the system through a data link (InterviewBit, 2024).

Below is a diagram showing the system architecture of the prototype.

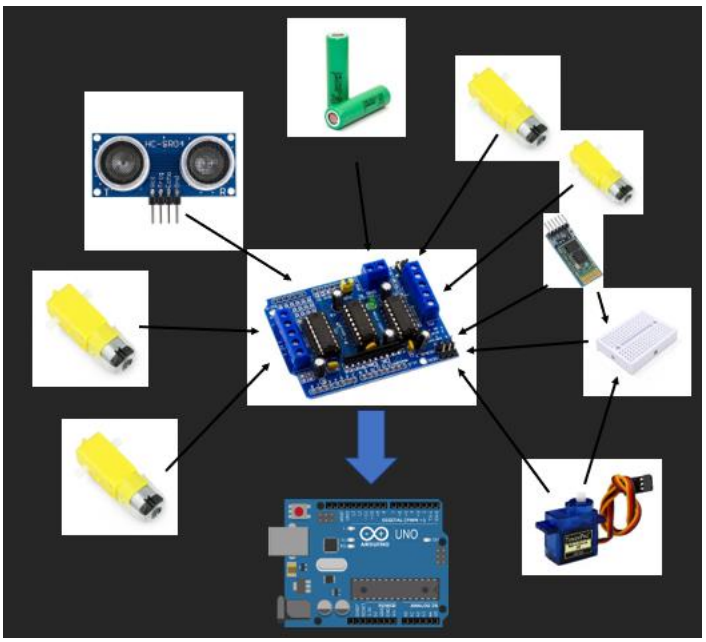

Figure 1: System Architecture of the Prototype.

- **Block Diagram**

A structure, project, or scenario is graphically represented by a block diagram. It offers a functional perspective of a system and shows the connections between its various components (Miro, 2024).

Below is a diagram showing the block diagram of the prototype.

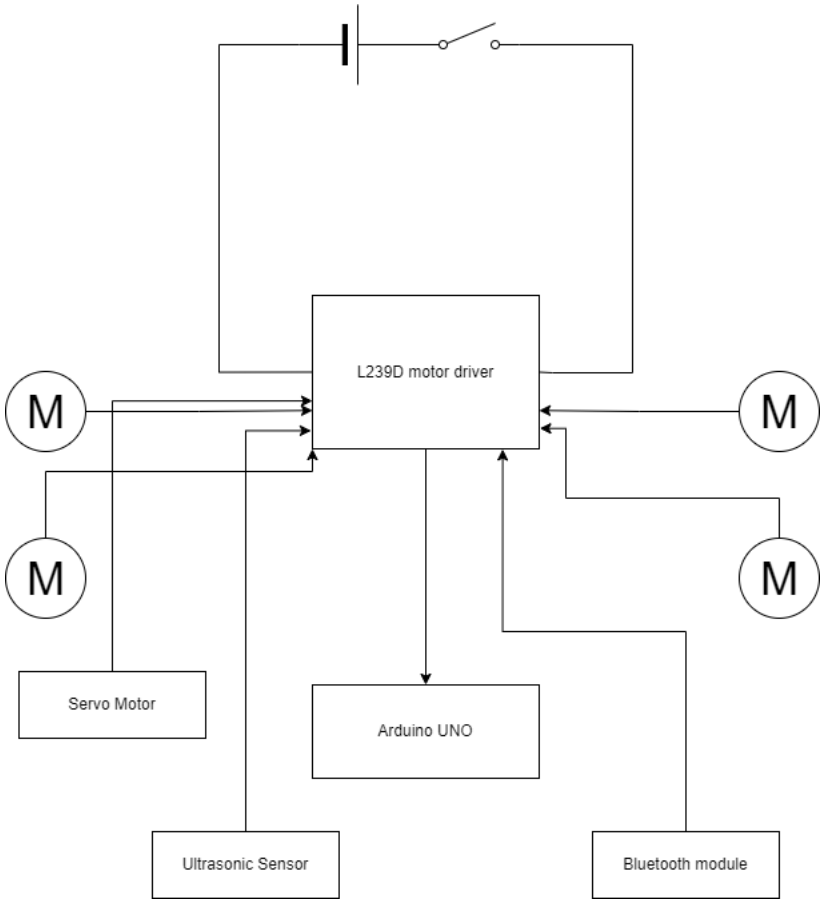

Figure 2: Block Diagram of Prototype.

- **Circuit Diagram**

A circuit diagram is a graphical depiction that makes an electrical circuit simpler. It is sometimes referred to as an electrical diagram, elementary diagram, or electronic schematic. It functions as a visual aid for electrical and electronic equipment design, building, and maintenance (BYJU'S, 2024).

Below is a diagram showing the circuit diagram of the prototype.

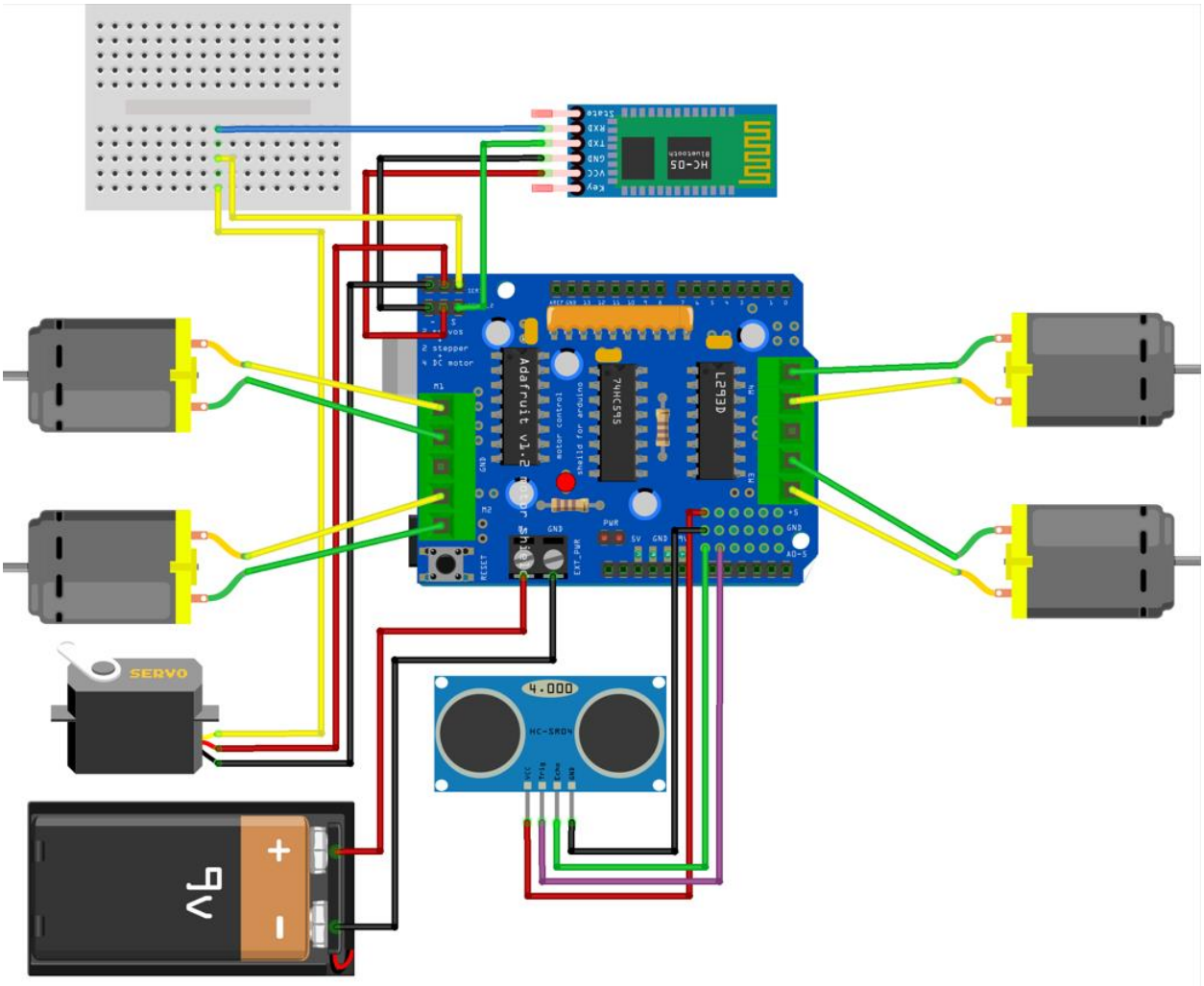

Figure 3: Circuit Diagram of Prototype.

- **Schematic Diagram**

A schematic diagram is a basic two-dimensional circuit illustration that demonstrates how various electrical components work together and are connected. Understanding the schematic symbols that indicate the components on a schematic diagram is essential for PCB designers (https://www.protoexpress.com/blog/whats-the-meaning-of-schematic-diagram/, 2024).

Below is a diagram showing the schematic diagram of the prototype.

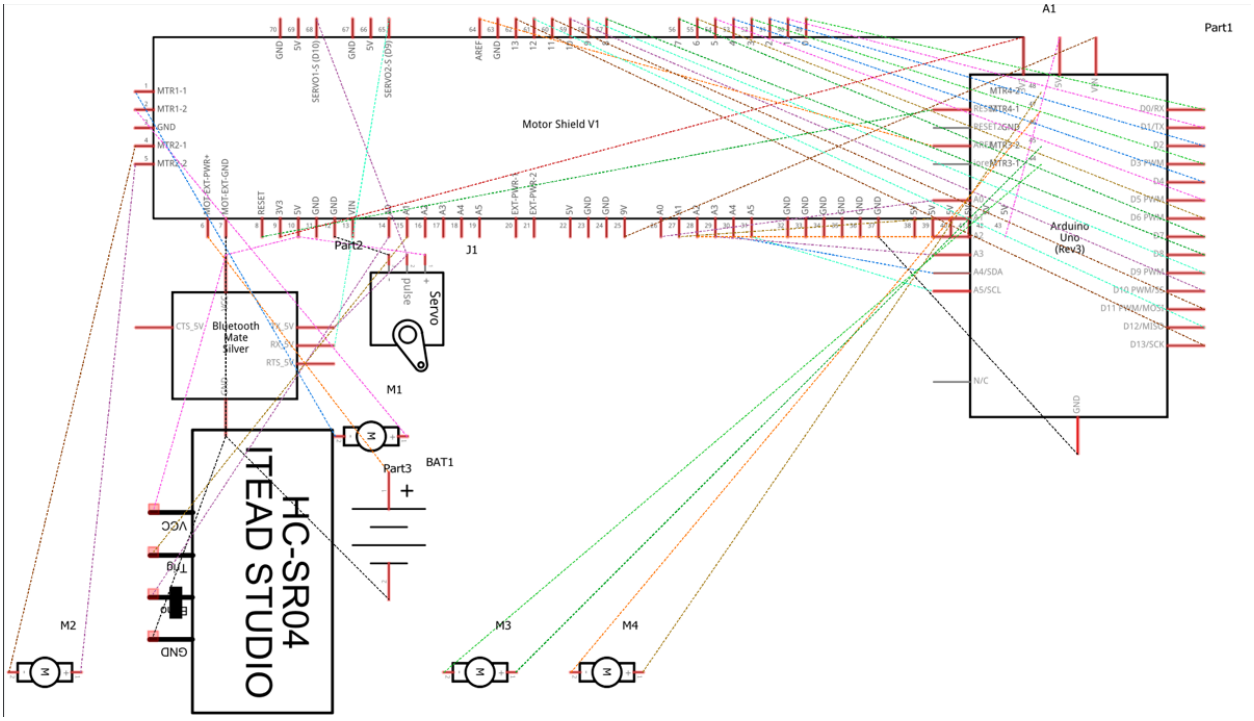

Figure 4: Schematic Diagram of Prototype.

- **Flowchart**

A flowchart is a graphic representation of a computer algorithm, system, or process. They are extensively used in many various domains to plan, analyze, document, and communicate in simple, understandable diagrams-often complex process (Lucid Software Inc, 2024).

Below is a diagram showing the flowchart of the prototype.

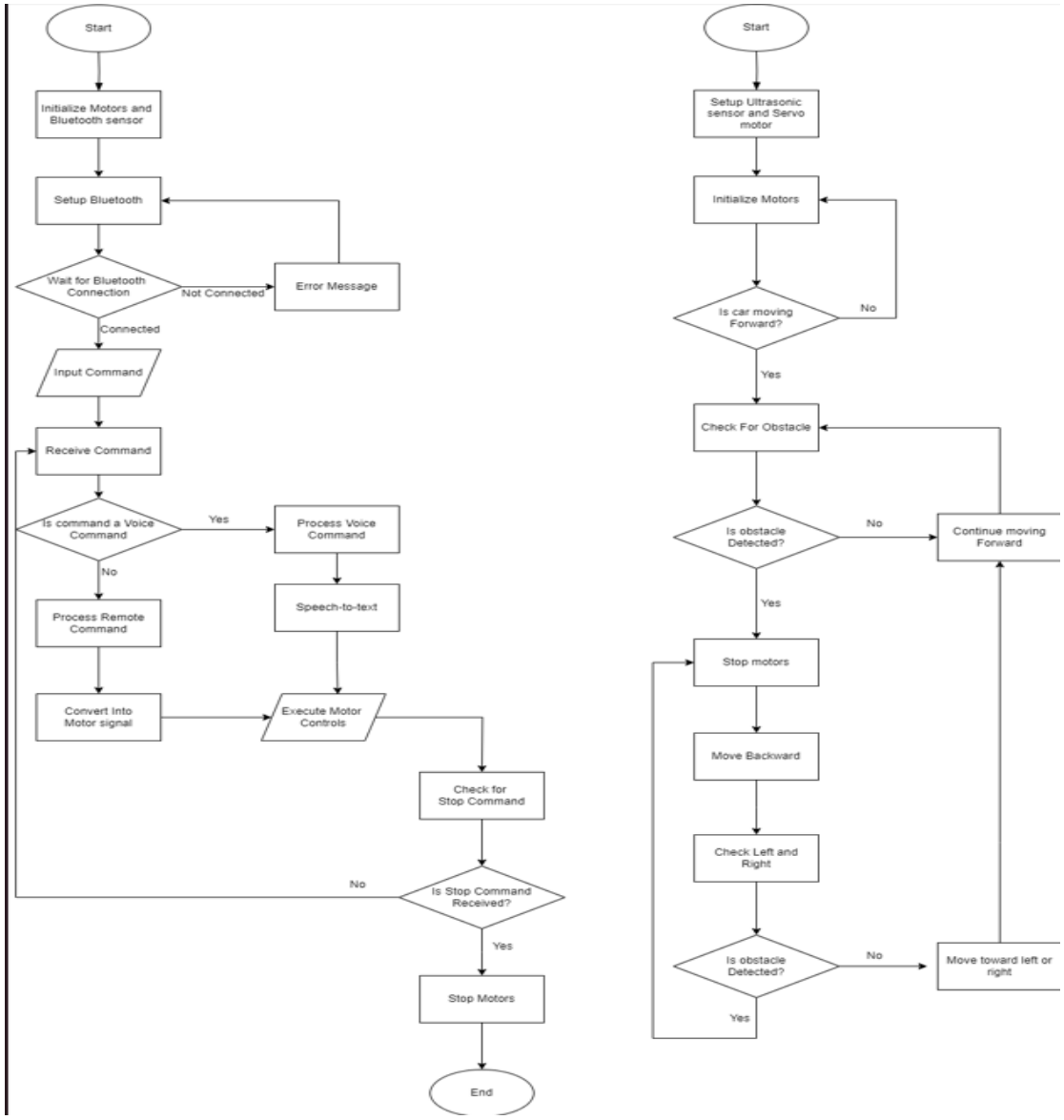

Figure 5: Flowchart of Prototype.

2.3 Requirement Analysis

- **Arduino UNO**

It is the main microcontroller which controls all the components of the prototype.

Read more…

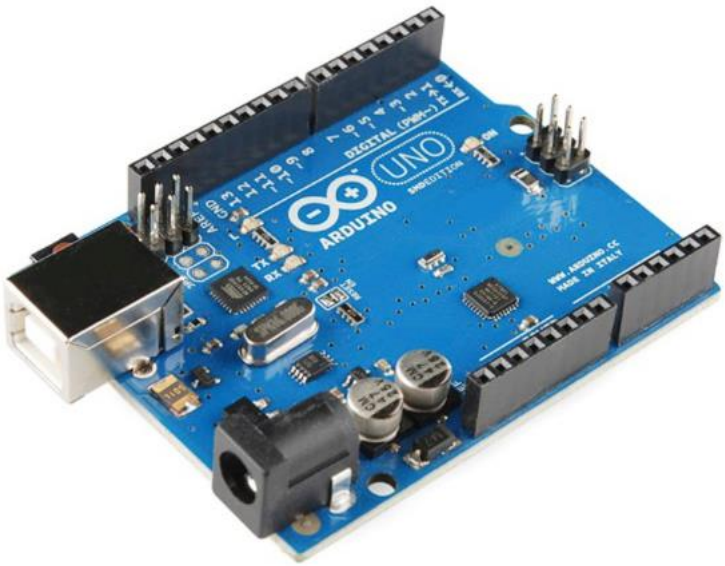

Figure 6: Arduino Uno (Evil Mad Scientist, 2024).

- **L293D Motor Driver Shield**

Its pin is connected to pins of Uno and all the motors, sensors are connected with motor driver shield. Read more…

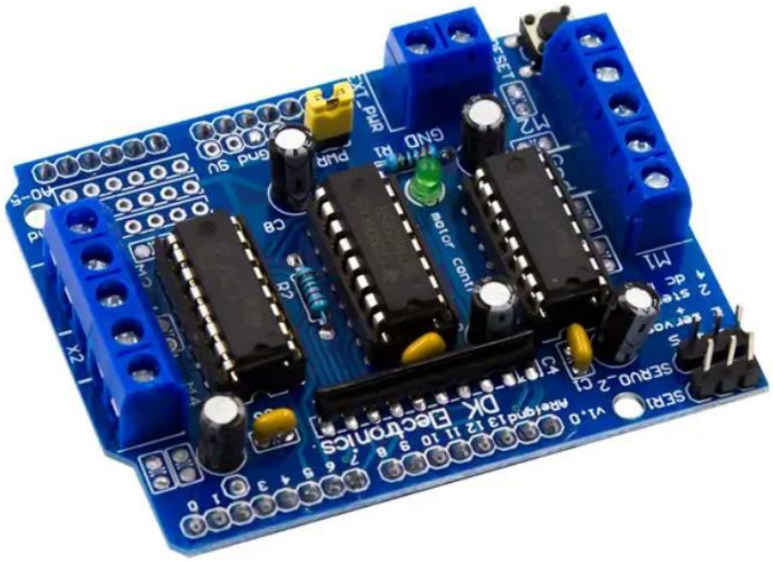

Figure 7: L293D Motor Driver Shield (Agnihotri, 2024).

- **HC-05 Bluetooth Module**

It is a Bluetooth module which is used for remote/voice access to the wheelchair.

Read more…

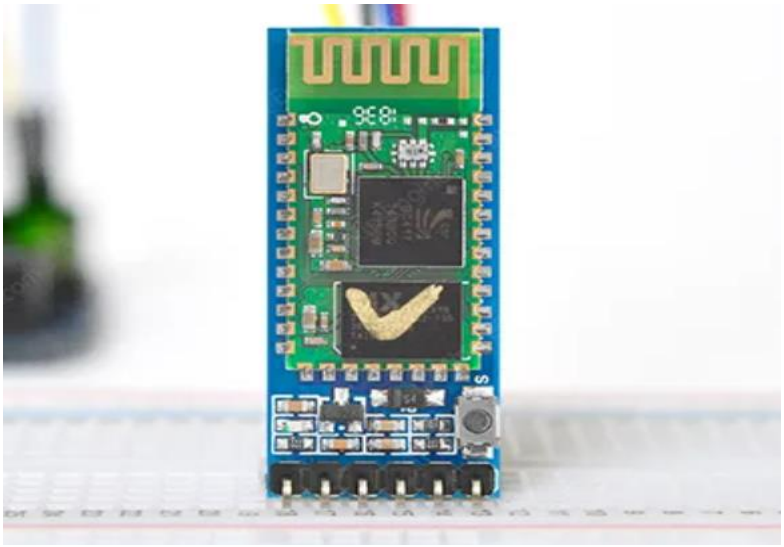

Figure 8: HC-05 Bluetooth Module (LastMinuteEngineers.com, 2024).

- **Ultrasonic Sensor**

It is an Active sensor that receives ultrasonic waves in Analog form and is used for obstacle detection. Read more…

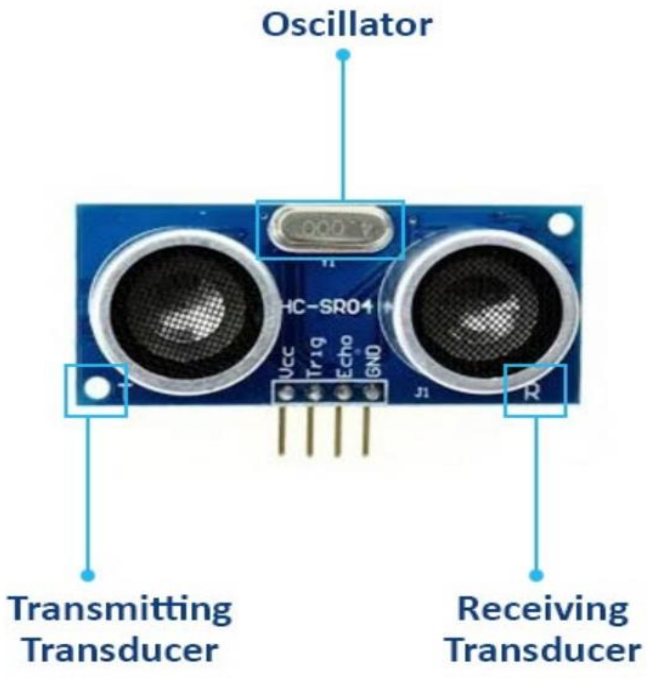

Figure 9: Ultrasonic Sensor (Barik, 2024).

- **Servo Motor**

It is an Electrical Actuator which has Rotary movement and is used for the rotation of ultrasonic sensor. Read more…

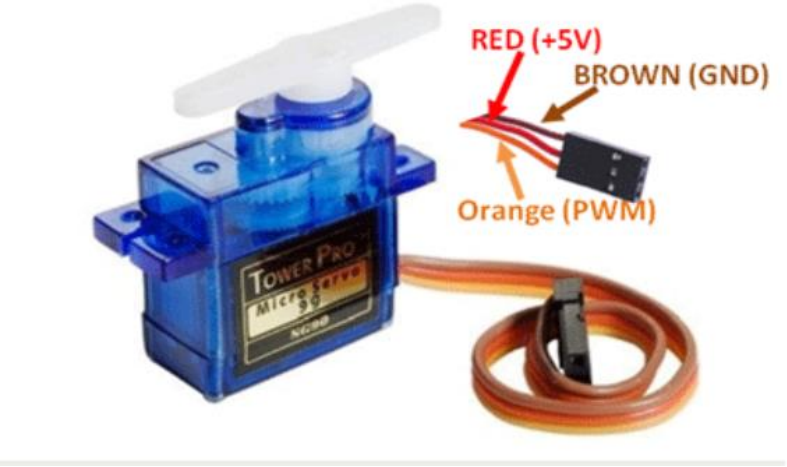

Figure 10: Servo Motor (Components101, 2024).

- **DC Geared Motor**

It is an Electrical Actuator which has Rotary movement and is used for the movement of wheelchair. Read more…

Figure 11: DC Geared Motor (GRobotronics, 2024).

- **Jumper wires**

It is used to connect sensors and actuators to the pins of motor driver shield. Read more…

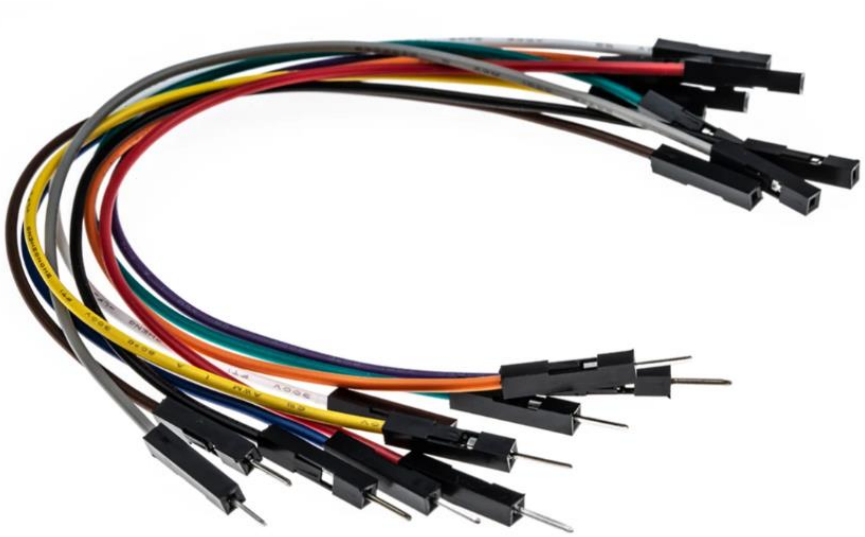

Figure 12: Jumper Wires (RS Components Ltd, 2024).

- **Battery and Battery Holder**

It is used to supply power to the wheelchair. Read more…

Figure 13: Battery and Battery Holder (OKW Gehäusesysteme, 2024).

- **Switch**

It is used to cut and give battery voltage supply to the wheelchair. Read more…

Figure 14: Switch (Elcom International, 2024).

- **Breadboard**

It is used to connect Bluetooth, Servo motor and L293D Motor Driver pins. Read more…

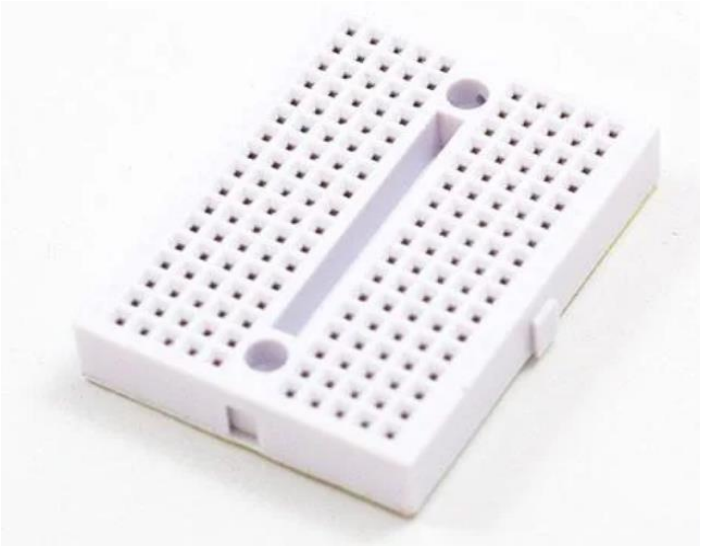

Figure 15: Breadboard (Embeded Studio, 2024).

3. Development

To complete this project, we went through various development phases such as the designing and planning, resource collection and system development which are as follows:

3.1 Designing and Planning

The initial thought was to make a wheelchair which could move using the hands gestures of the people, but it could not be done because of its complex algorithms and lack of knowledge so we thought instead of using the hand gestures, remote/voice-controlled wheelchair would also be a good option. So, we started to do our planning and designing for wheelchair which could be controlled by using remote/voice control.

Before designing the system architecture, we first outlined the tasks. Our attention was directed towards software features such as voice and Bluetooth control and motor control. Constant testing made sure all the parts functioned properly. A top focus was documentation, which included code comments and circuit diagrams. We set aside time slots for research, design, implementation, testing, and improvement according to a schedule. Safety was always the priority. In conclusion, careful planning led to a professional developed remote wheelchair.

3.2 Resource Collection

We used a combination of approaches for our project's resource collection phase, sourcing necessary materials from the College Resource Department and from the market. The Resource Department made basic resources such as Arduino Uno, Bluetooth Module, Ultrasonic Sensor, DC geared motors, Servo motor and some jumper wires available. Other materials such as motor driver shield, battery and battery holder, breadboard, etc., were obtained from the market.

3.3 System Development

The system development part was completed in six phases which are as follows:

Phase 1: In the first phase, the L293D motor driver shield and Arduino Uno was taken first. Then the L293D motor driver shield was connected to Arduino Uno by connecting the pins of driver shield to the pins of Arduino Uno.

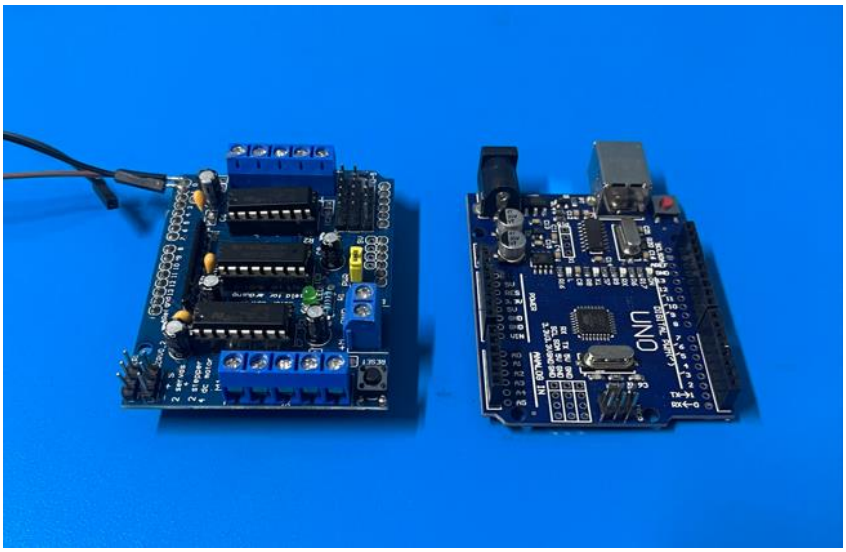

Figure 16: Connecting pins of Arduino Uno with L293D Motor Driver Shield.

The joint structure looks as shown in the figure below.

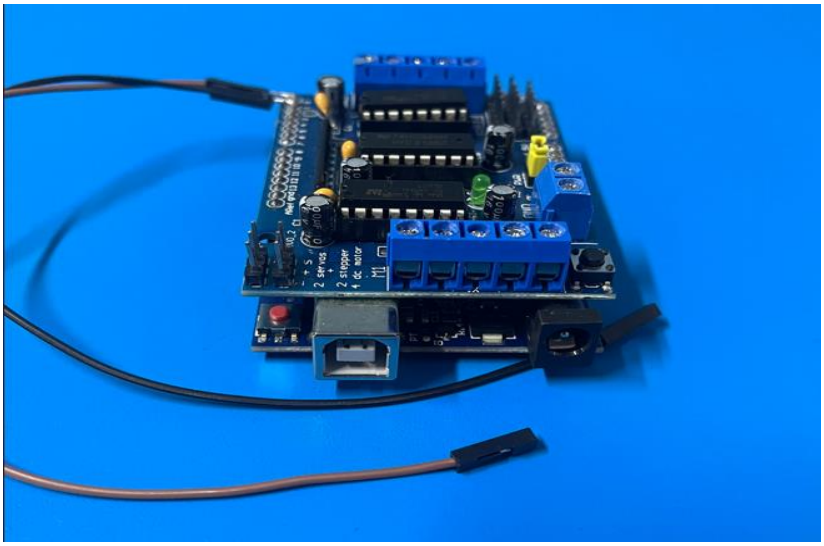

Figure 17: Joint structure of Arduino Uno and L293D Motor Driver Shield.

Phase 2: Then in the next phase, four DC geared motors were attached to the back side of the base and in each of the motors, a tire was attached. Then in each motor, two wires were connected as shown in the figure below.

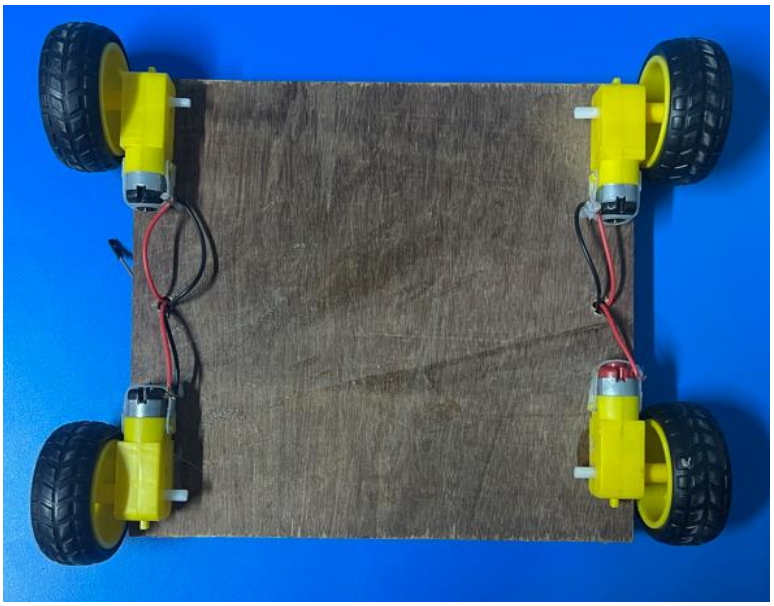

Figure 18: Attachment of DC Geared Motor and Wheels.

The other sides of the copper wire are connected to its respective sides in the motor driver shield as shown in the figure below.

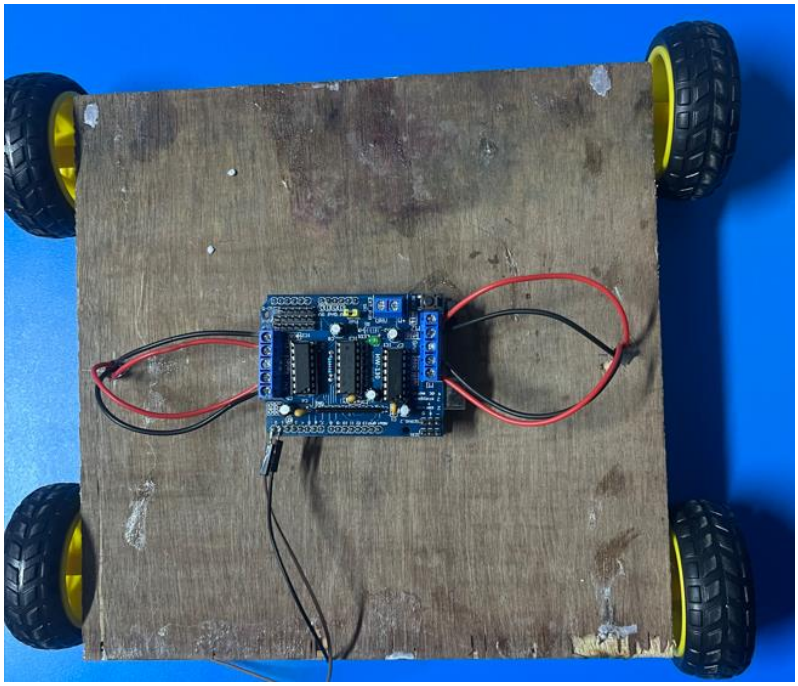

Figure 19: Wiring of DC Geared Motors with Arduino UNO.

Phase 3: Then, the battery holder was connected to the driver shield by distinguishing the positive and negative sides and two lithium-ion battery was inserted into the battery holder. A on/off switch was also attached to the battery holder to save the power of the battery as shown in the figure below.

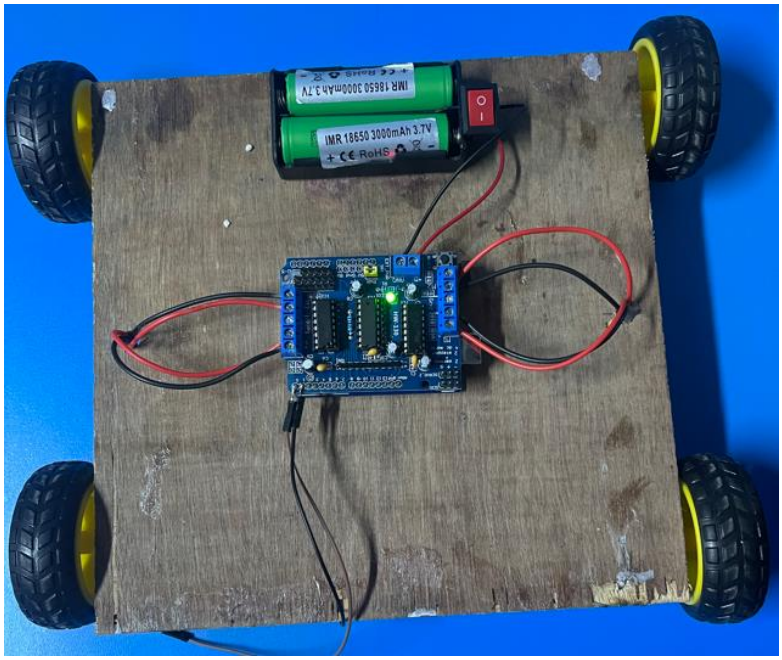

Figure 20: Attachment of Battery Holder and Battery.

The switch was turned on and off to check whether the batteries were supplying power or not.

Phase 4: In the fourth phase of the development, the Bluetooth module was connected to the motor driver shield using a breadboard. GND and VCC pins of Bluetooth module were connected to positive and negative pin of Motor Driver shield. TXD pin was connected to Servo 2 pin of Motor Driver Sheild while RXD pin was connected to breadboard. And then it was checked whether the Bluetooth module was turned on or not. The Bluetooth module blinked when the power was supplied to it as shown in the figure below.

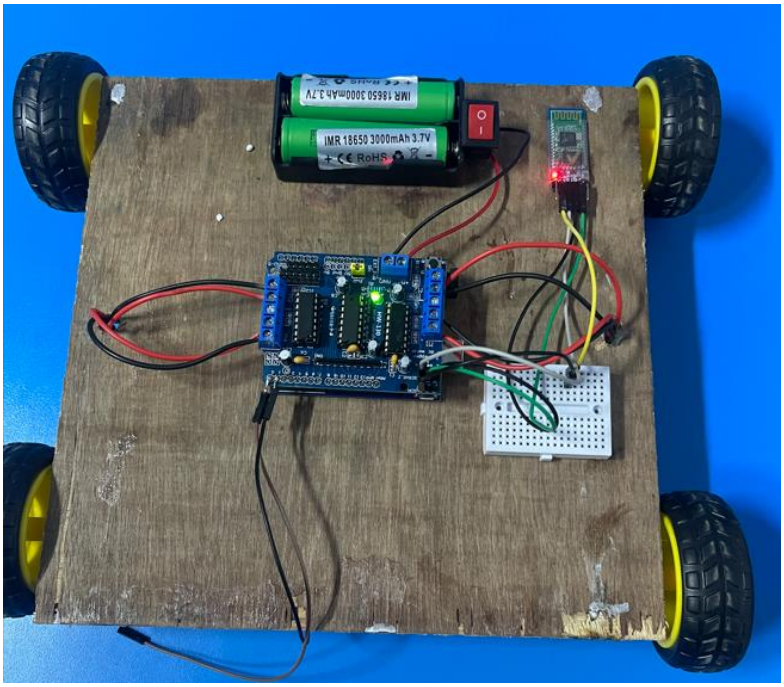

Figure 21: Connection of HC-05 Bluetooth Module.

Phase 5: In the fifth phase, the ultrasonic sensor was attached to the servo motor using hot glue gun as shown in the figure below.

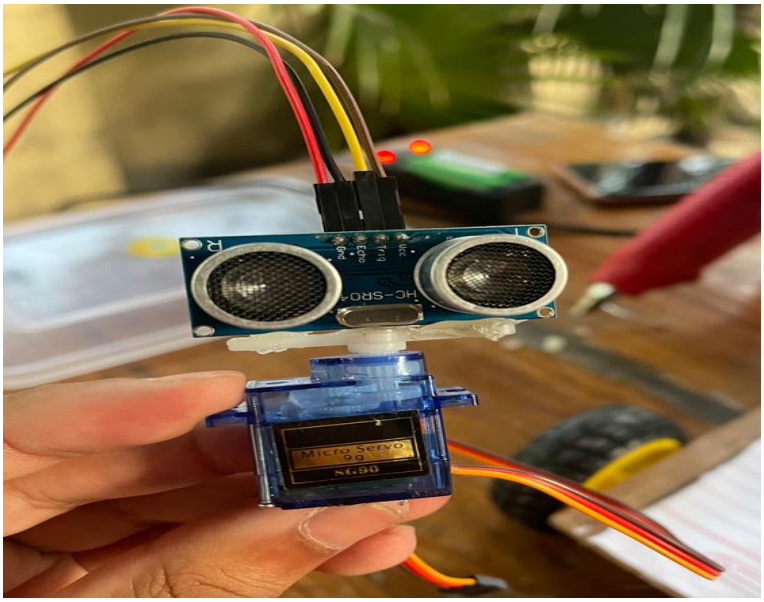

Figure 22: Attachment of Ultrasonic Sensor with Servo Motor.

Then the servo motor was attached to the front side of the base using hot glue gun and their wires were connected to the motor driver shield to their respective pins. Likewise, VCC and GND pins of ultrasonic were connected to 5V and GND pins of motor driver shield respectively. Similarly, echo and trig pin of ultrasonic sensor was connected to AO and A1 pins of Motor Driver shield as shown in the figure below.

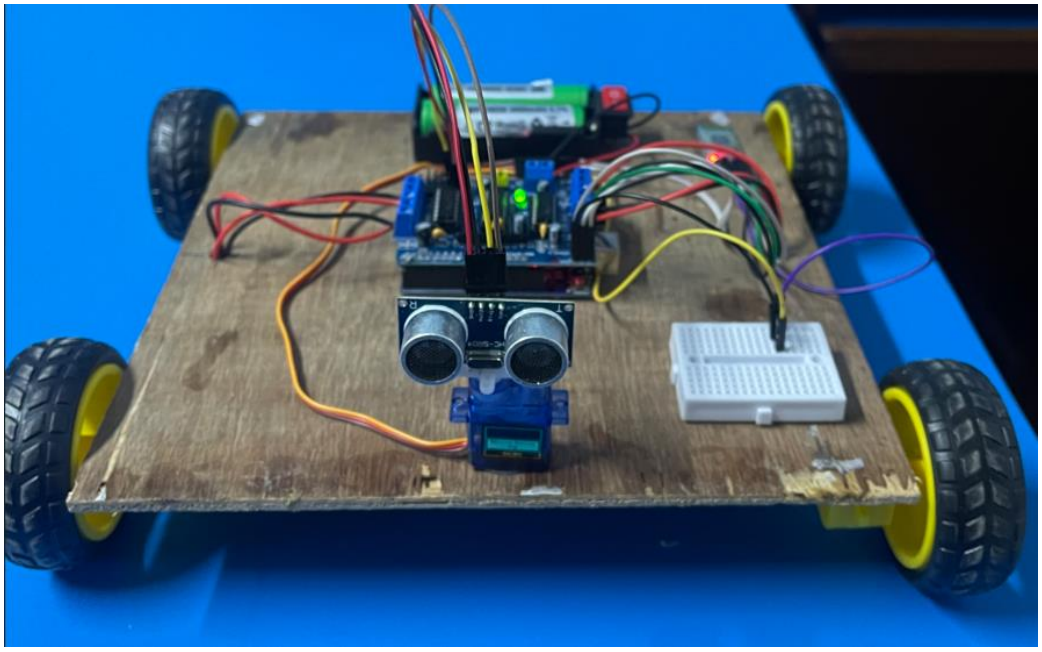

Figure 23: Attachment of Ultrasonic Sensor and Servo Motor with L293D Motor Driver Shield.

Phase 6: In the sixth phase, the code required for the prototype to function was uploaded to the Arduino Uno board after connecting the Arduino Uno to the laptop, compiling the code in the Arduino Uno IDE as shown in the figure below.

```
#include <AFMotor.h>
#include <SoftwareSerial.h>

SoftwareSerial bluetoothSerial(9, 10); // RX, TX

// Motors pin initial
AF_DCMotor dcmotor1(1, MOTOR12_1KHZ);
AF_DCMotor dcmotor2(2, MOTOR12_1KHZ);
AF_DCMotor dcmotor3(3, MOTOR34_1KHZ);
AF_DCMotor dcmotor4(4, MOTOR34_1KHZ);

char command;

void setup()
{
  bluetoothSerial.begin(9600);  // Setting the baud rate of Bluetooth module/ The amount of changes to the signal
}

void loop() {
  if (bluetoothSerial.available() > 0) {
    command = bluetoothSerial.read();

    Stop(); // Start with the motor stopped.
```

Figure 24: Code for Bluetooth/ Voice control.

```
    switch (command) {
      case 'U': // For forward
        forward();
        break;
      case 'D': // For backward
        back();
        break;
      case 'L': // For left
        left();
        break;
      case 'R': // For right
        right();
        break;
    }
  }
}

void forward() // For forward direction of the motor.
{
  dcmotor1.setSpeed(255); // Defining the max velocity
  dcmotor1.run(FORWARD);  // Clockwise rotation of the motor
  dcmotor2.setSpeed(255); // Defining the max velocity
  dcmotor2.run(FORWARD);  // Clockwise rotation of the motor
  dcmotor3.setSpeed(255); // Defining the max velocity
```

Figure 25: Code for Bluetooth/ Voice control continued.

```
  dcmotor4.run(FORWARD);  // Clockwise rotation of the motor
}

void back() // For backward direction of the motor.
{
  dcmotor1.setSpeed(255); // Defining the max velocity
  dcmotor1.run(BACKWARD); // Anti-clockwise rotation of the motor
  dcmotor2.setSpeed(255); // Defining the max velocity
  dcmotor2.run(BACKWARD); // Anti-clockwise rotation of the motor
  dcmotor3.setSpeed(255); // Defining the max velocity
  dcmotor3.run(FORWARD); // Clockwise rotation of the motor
  dcmotor4.setSpeed(255); //Defining the max velocity
  dcmotor4.run(BACKWARD); // Anti-clockwise rotation of the motor
}

void left() // For left direction of the motor.
{
  dcmotor1.setSpeed(255); // Defining the max velocity
  dcmotor1.run(BACKWARD); // Anti-clockwise rotation of the motor
  dcmotor2.setSpeed(255); // Defining the max velocity
  dcmotor2.run(BACKWARD); // Anti-clockwise rotation of the motor
  dcmotor3.setSpeed(255); // Defining the max velocity
  dcmotor3.run(BACKWARD);  // Anti-clockwise rotation of the motor
  dcmotor4.setSpeed(255); // Defining the max velocity
  dcmotor4.run(FORWARD);  // Clockwise rotation of the motor
```

Figure 26: Code for Bluetooth/ Voice control continued.

```
}

void right() // For right direction of the motor.
{
  dcmotor1.setSpeed(255); // Defining the max velocity
  dcmotor1.run(FORWARD);  // Clockwise rotation of the motor
  dcmotor2.setSpeed(255); // Defining the max velocity
  dcmotor2.run(FORWARD);  // Clockwise rotation of the motor
  dcmotor3.setSpeed(255); // Defining the max velocity
  dcmotor3.run(FORWARD); // Clockwise rotation of the motor
  dcmotor4.setSpeed(255); // Defining the max velocity
  dcmotor4.run(BACKWARD); // Anti-clockwise rotation of the motor
}

void Stop() // For stopping of the motor.
{
  dcmotor1.setSpeed(0);  // Defining the min velocity
  dcmotor1.run(RELEASE); // Release the button when stopping the motor
  dcmotor2.setSpeed(0);  // Defining the min velocity
  dcmotor2.run(RELEASE); // Release the button when stopping the motor
  dcmotor3.setSpeed(0);  // Defining the min velocity
  dcmotor3.run(RELEASE); // Release the button when stopping the motor
  dcmotor4.setSpeed(0);  // Defining the min velocity
  dcmotor4.run(RELEASE); // Release the button when stopping the motor
}
```

Figure 27:Code for Bluetooth/ Voice control continued.

```
#include <Servo.h> //Importing library
#include <AFMotor.h> // Importing library
#define Echo A0
#define Trig A1
#define motor 10
#define spoint 103
int length;
int Left;
int Right;
int Lt = 0;
int Rt = 0;
int L1 = 0; //Lt1
int R1 = 0; // Rt1
Servo servo;
//Motors pin initial
AF_DCMotor dcmotor1(1, MOTOR12_1KHZ);
AF_DCMotor dcmotor2(2, MOTOR12_1KHZ);
AF_DCMotor dcmotor3(3, MOTOR34_1KHZ);
AF_DCMotor dcmotor4(4, MOTOR34_1KHZ);
void setup() {
  pinMode(Trig, OUTPUT);
  pinMode(Echo, INPUT);
  servo.attach(motor);
}
```

Figure 28: Code for Obstacle detection.

```
void loop() {
  Obstacle();
}

void Obstacle() {
  length = ultrasonic();
  if (length <= 12) {
    Stop();
    back();
    delay(500);
    Stop();
    Lt = leftsee();
    servo.write(spoint);
    delay(800);
    Rt = rightsee();
    servo.write(spoint);
    if (Lt < Rt) {
      right();
      delay(500);
      Stop();
      delay(200);
    } else if (Lt > Rt) {
```

Figure 29: Code for Obstacle detection continued.

```
    } else if (Lt > Rt) {
      left();
      delay(500);
      Stop();
      delay(200);
    }
  } else {
    forward();
  }
}

// Function for distance reading of Ultrasonic Sensor
int ultrasonic() {
  digitalWrite(Trig, LOW);
  delayMicroseconds(4);
  digitalWrite(Trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(Trig, LOW);
  long t = pulseIn(Echo, HIGH);
  long cm = t / 29 / 2; // Distance converted time
  return cm;
}
```

Figure 30:Code for Obstacle detection continued.

```
void forward()
{
  dcmotor1.setSpeed(255); // Defining the max velocity
  dcmotor1.run(FORWARD);  // Clockwise rotation of the motor
  dcmotor2.setSpeed(255); // Defining the max velocity
  dcmotor2.run(FORWARD);  // Clockwise rotation of the motor
  dcmotor3.setSpeed(255); // Defining the max velocity
  dcmotor3.run(BACKWARD);  // Clockwise rotation of the motor
  dcmotor4.setSpeed(255); // Defining the max velocity
  dcmotor4.run(FORWARD);  // Clockwise rotation of the motor
}

void back()
{
  dcmotor1.setSpeed(255); // Defining the max velocity
  dcmotor1.run(BACKWARD); // Anti-clockwise rotation of the motor
  dcmotor2.setSpeed(255); // Defining the max velocity
  dcmotor2.run(BACKWARD); // Anti-clockwise rotation of the motor
  dcmotor3.setSpeed(255); // Defining the max velocity
  dcmotor3.run(FORWARD); // Anti-clockwise rotation of the motor
  dcmotor4.setSpeed(255); // Defining the max velocity
  dcmotor4.run(BACKWARD); // Anti-clockwise rotation of the motor
}
```

Figure 31: Code for Obstacle detection continued.

```
void left()
{
  dcmotor1.setSpeed(255); // Defining the max velocity
  dcmotor1.run(BACKWARD); // Anti-clockwise rotation of the motor
  dcmotor2.setSpeed(255); // Defining the max velocity
  dcmotor2.run(BACKWARD); // Anti-clockwise rotation of the motor
  dcmotor3.setSpeed(255); // Defining the max velocity
  dcmotor3.run(BACKWARD);  // Anti-clockwise rotation of the motor
  dcmotor4.setSpeed(255); // Defining the max velocity
  dcmotor4.run(FORWARD);  // Clockwise rotation of the motor
}

void right()
{
  dcmotor1.setSpeed(255); // Defining the max velocity
  dcmotor1.run(FORWARD);  // Clockwise rotation of the motor
  dcmotor2.setSpeed(255); // Defining the max velocity
  dcmotor2.run(FORWARD);  // Clockwise rotation of the motor
  dcmotor3.setSpeed(255); // Defining the max velocity
  dcmotor3.run(FORWARD); // Clockwise rotation of the motor
  dcmotor4.setSpeed(255); // Defining the max velocity
  dcmotor4.run(BACKWARD); // Anti-clockwise rotation of the motor
}
```

Figure 32: Code for Obstacle detection continued.

```
void Stop()
{
  dcmotor1.setSpeed(0);  // Defining the min velocity
  dcmotor1.run(RELEASE); // Release the button when stopping the motor
  dcmotor2.setSpeed(0);  // Defining the min velocity
  dcmotor2.run(RELEASE); // Release the button when stopping the motor
  dcmotor3.setSpeed(0);  // Defining the min velocity
  dcmotor3.run(RELEASE); // Release the button when stopping the motor
  dcmotor4.setSpeed(0);  // Defining the min velocity
  dcmotor4.run(RELEASE); // Release the button when stopping the motor
}
int rightsee() {
  servo.write(20);
  delay(800);
  Left = ultrasonic();
  return Left;
}
int leftsee() {
  servo.write(180);
  delay(800);
  Right = ultrasonic();
  return Right;
}
```

Figure 33: Code for Obstacle detection continued.

Phase 7: This is the final phase of system prototype. In this phase, the designing of the prototype is done. We used wooden base for the balancing of the whole prototype. We used Styrofoam for the whole structure of the wheelchair and then wrapped it with a black chart paper using hot glue gun.

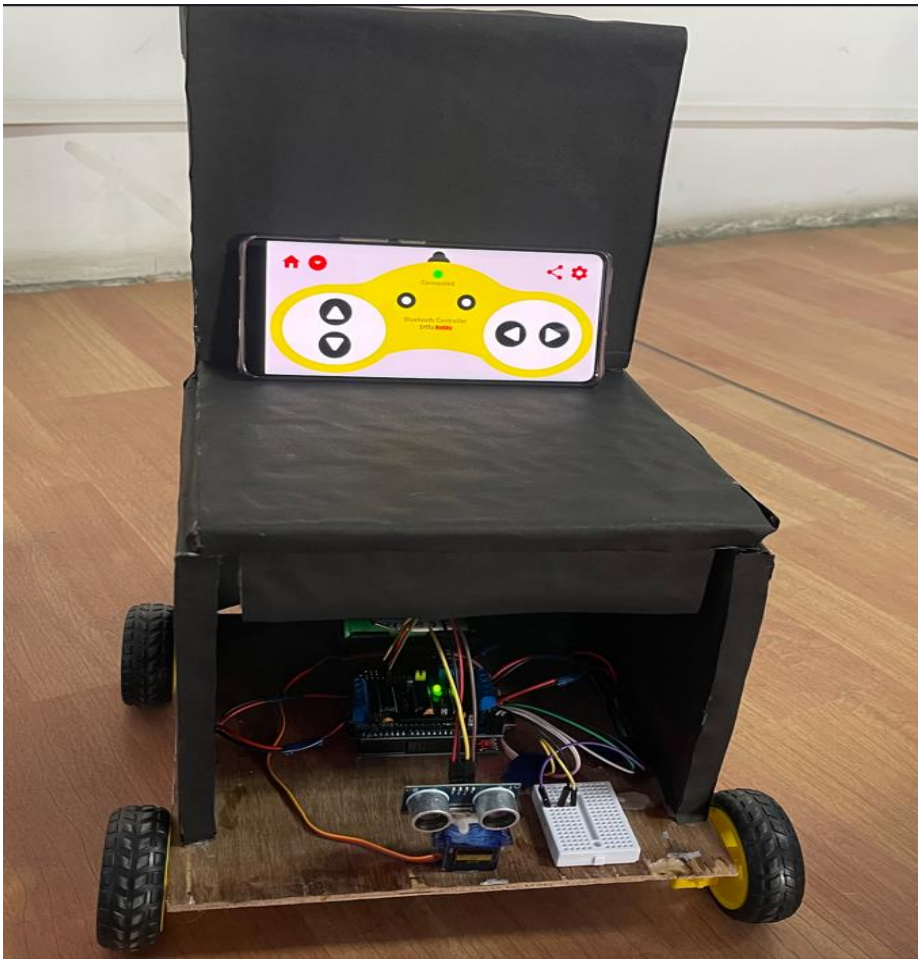

Figure 34: Final picture of Prototype.

Like this, the system development phase was completed in seven phases.

4. Results and Findings

4.1 Results

After the system development part was completed, the time was to check if the system performs the same as it was intended to perform at the beginning. According to the initial planning, the wheelchair was to be controlled with a controller or an android application by connecting the Bluetooth device and should work according to the provided command.

So, at first, we installed an android application and paired the Bluetooth module in the device. The Bluetooth module was connected to the wheelchair and using the commands in the application, the wheelchair was moving exactly as we wanted it to move. The code was working properly, and the motors were working according to the code provided to it. The ultrasonic sensor was also able to detect the obstacles and each time an obstacle appears, the motors stop for certain time and the sensor searches for another direction where there is no obstacle and tries to go in that direction.

4.2 Findings (Testing)

Certain tests were conducted to check whether the wheelchair was functioning right or not. The tests are shown in the table below.

Test 1

Objective	To check whether the Bluetooth module gets connected to the application or not.
Action	• The switch of the wheelchair was turned on. • The Bluetooth module was paired to the Bluetooth of the application device. • The Bluetooth was connected.
Expected outcome	The Bluetooth module of the wheelchair would connect to the Bluetooth of the application device.
Actual outcome	The Bluetooth module of the wheelchair was connected to the Bluetooth of the application device.
Result	The test was successful.

Table 1: To check whether the Bluetooth module gets connected to the application or not.

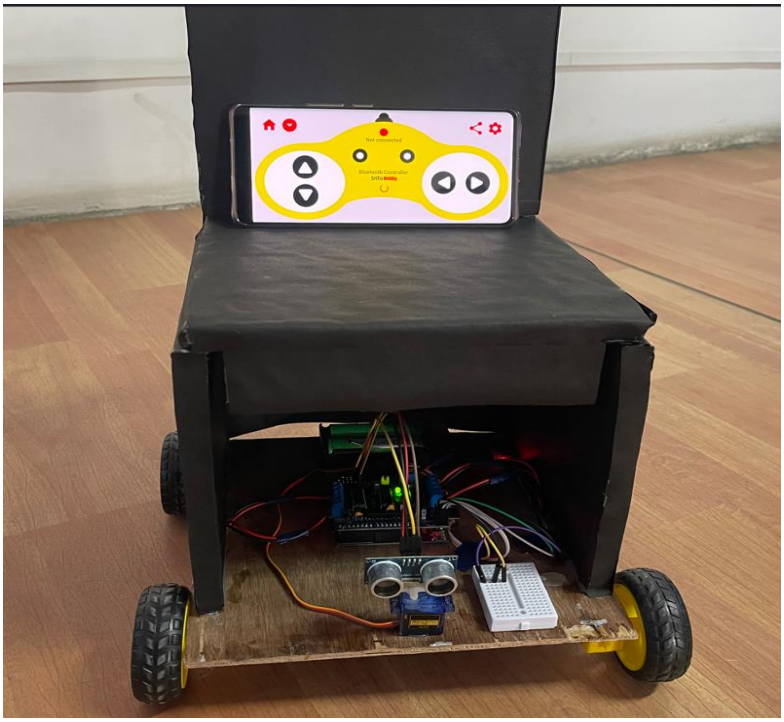

Figure 35: Showing disconnected Bluetooth connection.

Figure 36: Showing process of connecting Bluetooth.

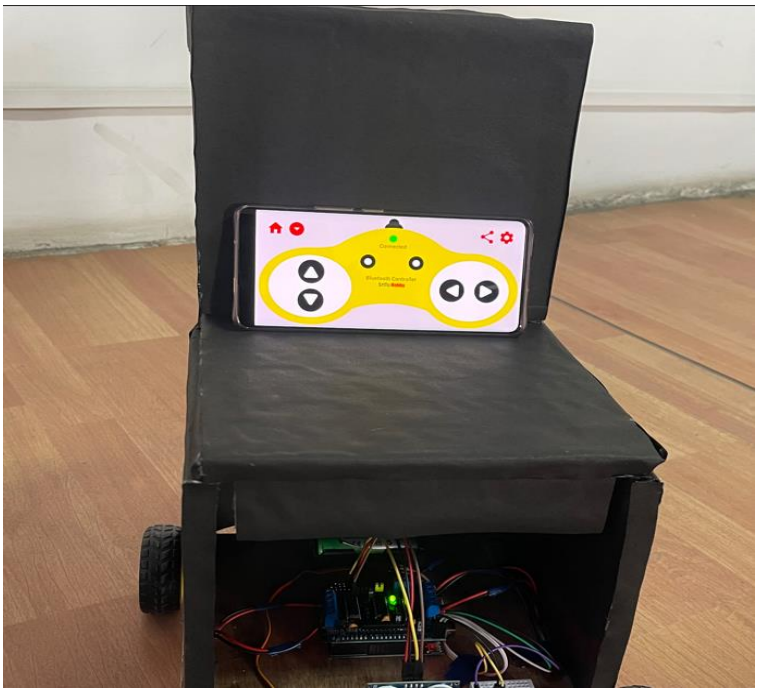

Figure 37: Showing Bluetooth connection successful.

Test 2

Objective	To check whether the wheelchair moves with the manual command on the application or not.
Action	The forward, backward, right, and left button was pressed one at a time on the command of application.
Expected outcome	The wheelchair would have moved forward, backward, right, and left respectively.
Actual outcome	The wheelchair moved forward, backward, right, and left respectively.
Result	The test was successful.

Table 2: To check whether the wheelchair moves with the manual command on the application or not.

Figure 38: Pressing the forward command.

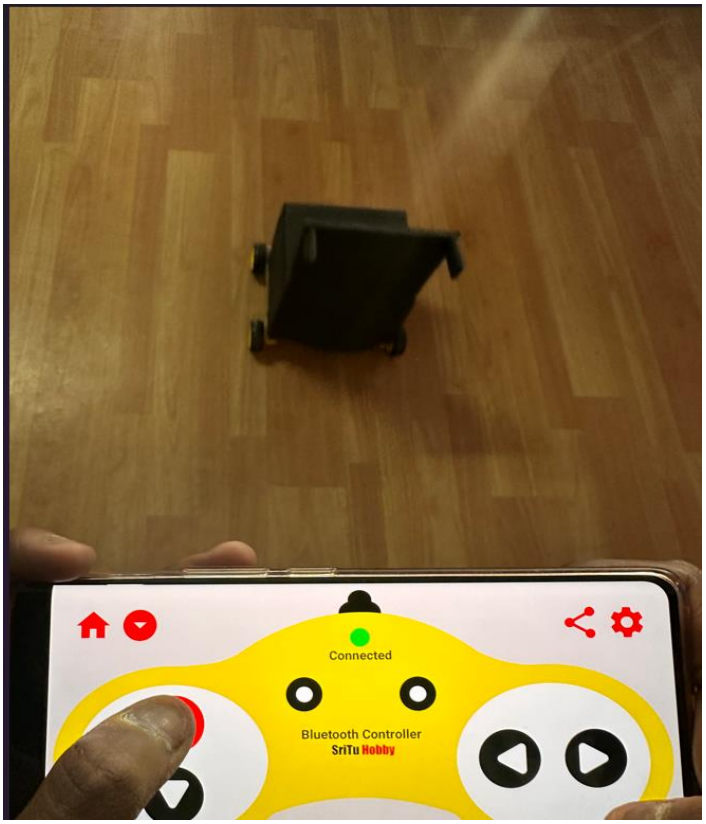

Figure 39: Wheelchair moved forward.

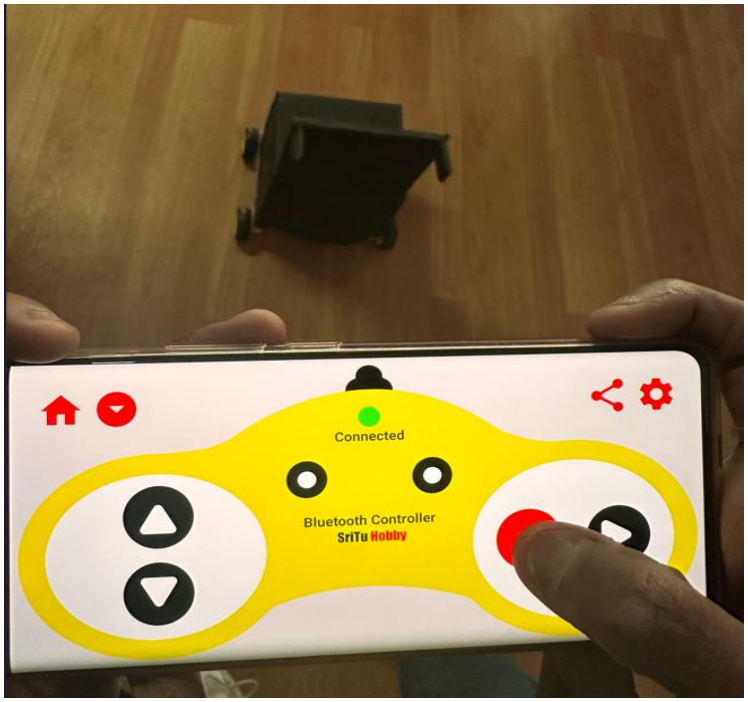

Figure 40: Pressing the Right command.

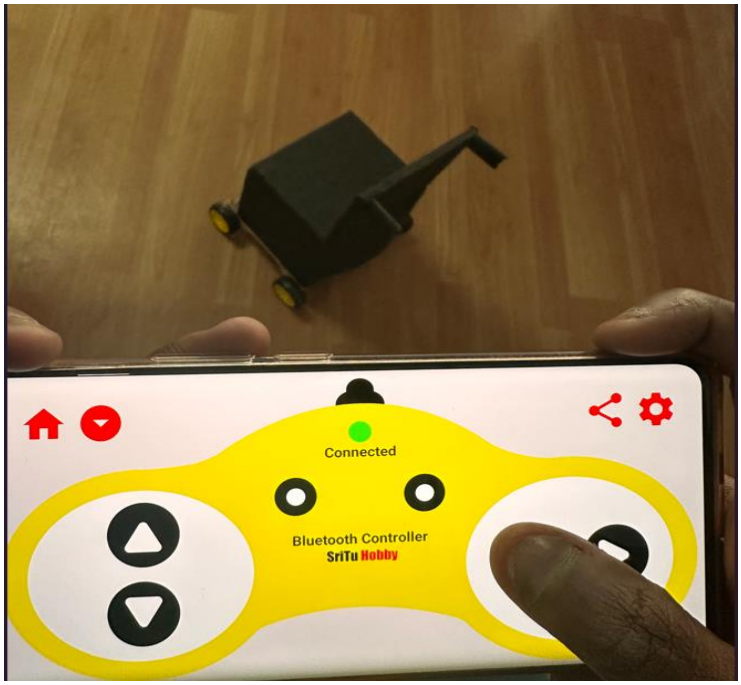

Figure 41: Wheelchair turning right.

Figure 42: Wheelchair turned right.

Test 3

Objective	To check whether the wheelchair moves with the voice command on the application or not.
Action	The forward, backward, right, and left command through voice was passed one at a time through the voice search of the application.
Expected outcome	The wheelchair would have moved forward, backward, right, and left respectively.
Actual outcome	The wheelchair moved forward, backward, right, and left respectively.
Result	The test was successful.

Table 3: To check whether the wheelchair moves with the voice command on the application or not.

Figure 43: Opening voice command.

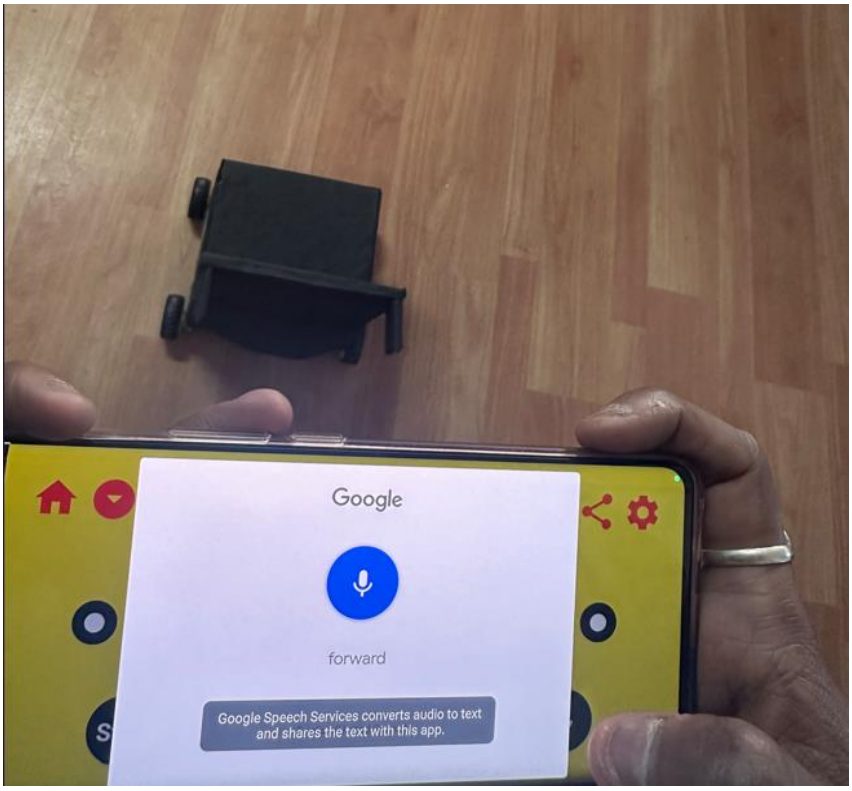

Figure 44: Giving voice command for Forward.

Figure 45: Wheelchair moving forward.

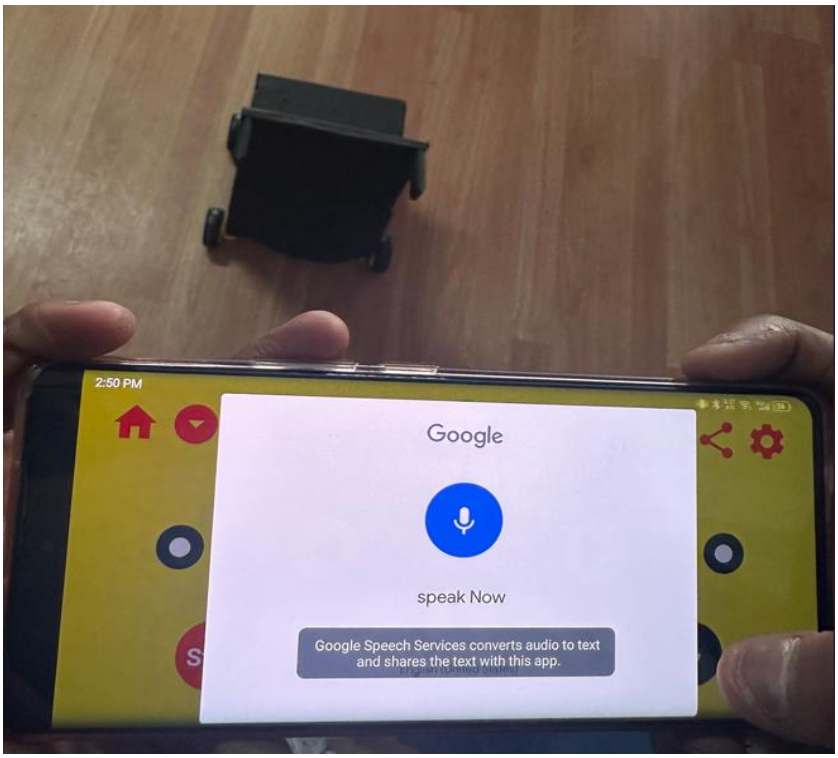

Figure 46: Opening voice command.

Figure 47: Giving voice command for Right.

Figure 48: Wheelchair moving Right.

Test 4

Objective	To check whether the ultrasonic sensor on the wheelchair detects the obstacle or not.
Action	The wheelchair was being moved and an obstacle was kept in front of the wheelchair.
Expected outcome	The ultrasonic sensor would have detected the obstacle, and the wheelchair would have stopped.
Actual outcome	The ultrasonic sensor detected the obstacle, and the wheelchair stopped.
Result	The test was successful.

Table 4: To check whether the ultrasonic sensor on the wheelchair detects the obstacle or not.

Figure 49: Turning on the Wheelchair.

Figure 50: Wheelchair moved near Obstacle.

Figure 51: Wheelchair moved backward after detecting the obstacle.

Test 5

Objective	To check whether the Bluetooth/ voice command and ultrasonic sensor works simultaneously or not.
Action	The Bluetooth was connected, and the wheelchair was moved forward continuously, and an obstacle was kept in front of the wheelchair.
Expected outcome	The wheelchair would have stopped after detecting the obstacle.
Actual outcome	The wheelchair did not detect the obstacle and it did not stop.
Result	The test was unsuccessful.

Table 5: To check whether the Bluetooth/ voice command and ultrasonic sensor works simultaneously or not.

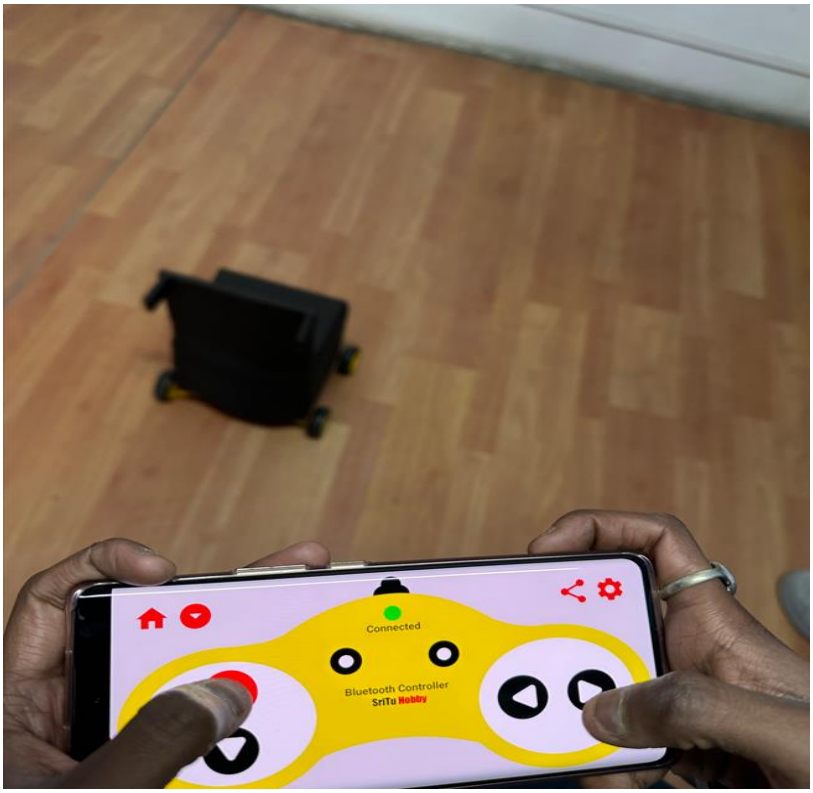

Figure 52: Giving the command for Forward in Bluetooth mode.

Figure 53: Wheelchair moving forward near obstacle.

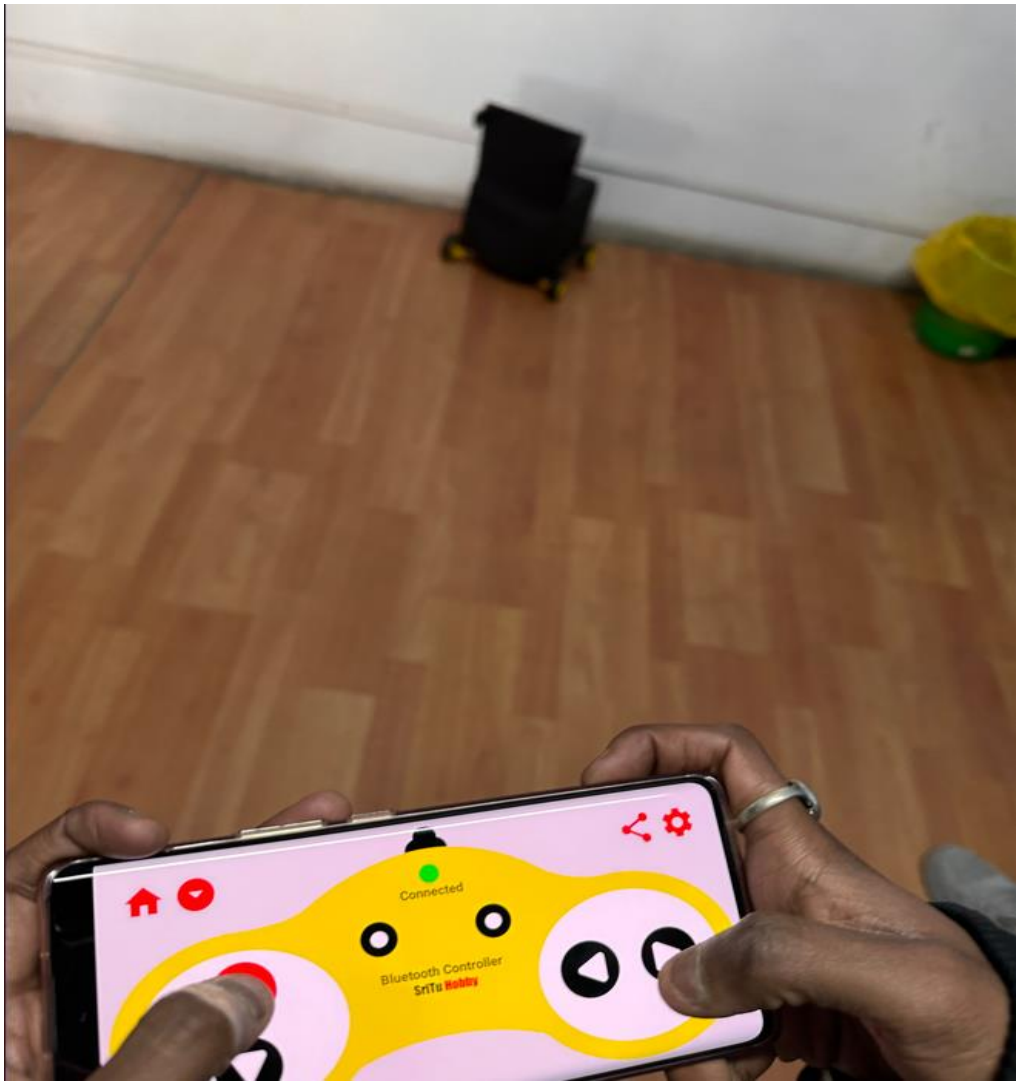

Figure 54: Wheelchair did not stop and slammed with the obstacle.

5. Future Works

As the wheelchair model continues to develop, various avenues for expansion and sustainability become obvious. Expanding the wheelchair's capabilities could involve investigating new features like improved obstacle detection methods, machine-learning based speech recognition, and the integration with external sensors. It is imperative to have a feasible deployment plan that considers eco-friendly materials, environmentally friendly components, and possibly backup power sources such as solar panels and advanced battery technologies.

Examining the incorporation of wireless charging technology may be a means of achieving sustained operational effectiveness. Moreover, tackling adaptability in diverse environments and weather scenarios continues to be a crucial element for additional advancement. Furthermore, investigating scalability and modular design principles will make it easier to incorporate updates and improvements in the future.

6. Conclusion

In conclusion, the creation of wheelchair prototype shows how diverse functionalities, such as voice recognition, obstacle avoidance, and remote-control features, have been successfully integrated. Iterative testing, systematic design, and careful planning are all responsible for the project's success. Using a dual-sourcing approach that combined market acquisitions with college resources worked well for securing the required hardware components.

In the future, there is a great deal of room for investigation and growth. Possibilities include enhancing adaptability, incorporating cutting-edge features, and implementing sustainable practices. This project establishes the groundwork for future innovation, teamwork, and useful applications in the larger IoT field in addition to providing the value of tactical planning and execution.

7. References

References

Components101, 2024. *Servo Motor SG-90.* [Online] Available at: https://components101.com/motors/servo-motor-basics-pinout-datasheet

GRobotronics, 2024. *DC Gear Motor TT - 130 RPM (With Wire).* [Online] Available at: https://grobotronics.com/dc-gear-motor-tt-130-rpm-with-wire.html?sl=en

Agnihotri, N., 2024. *Arduino's L293D motor driver shield guide.* [Online] Available at: https://www.engineersgarage.com/arduino-l293d-motor-driver-shield-tutorial/

Barik, P., 2024. *How to Interface Arduino with an Ultrasonic Sensor?.* [Online] Available at: https://circuitdigest.com/microcontroller-projects/interface-arduino-with-ultrasonic-sensor

BYJU'S, 2024. *What is a Circuit Diagram?.* [Online] Available at: https://byjus.com/physics/circuit-diagram/

Crowell, G., 2024. *What is a Breadboard?.* [Online] Available at: https://www.circuitbread.com/ee-faq/what-is-a-breadboard

Elcom International, 2024. *Electrical Switches | History, Types, & Uses.* [Online] Available at: https://elcom-in.com/electrical-switches-history-types-uses/

Electrical4U, 2023. *Servo Motor: Definition, Working Principle, and Applications.* [Online] Available at: https://www.electrical4u.com/what-is-servo-motor/#google_vignette

Embeded Studio, 2024. [Online] Available at: https://embededstudio.com/product/mini-breadboard-in-pakistan/

Evil Mad Scientist, 2024. *Arduino Uno Rev3 SMD.* [Online] Available at: https://shop.evilmadscientist.com/productsmenu/564

geeksforgeeks, 2023. *All about HC-05 Bluetooth Module | Connection with Android.* [Online]
Available at: https://www.geeksforgeeks.org/all-about-hc-05-bluetooth-module-connection-with-android/

Gillis, A. S., 2023. *tech target.* [Online] Available at: https://www.techtarget.com/iotagenda/definition/Internet-of-Things-IoT

https://www.protoexpress.com/blog/whats-the-meaning-of-schematic-diagram/, 2024. *Whats the meaning of schematic diagram.* [Online] Available at: https://www.protoexpress.com/blog/whats-the-meaning-of-schematic-diagram/

InterviewBit, 2024. *Introduction.* [Online]
Available at: https://www.interviewbit.com/blog/system-architecture/

javaTpoint, 2021. *Arduino UNO.* [Online]
Available at: https://www.javatpoint.com/arduino-uno

javaTpoint, 2024. *Arduino Switch.* [Online]
Available at: https://www.javatpoint.com/arduino-switch#:~:text=Switches%20are%20used%20to%20turn,the%20need%20for%20splice%20wire

Jost, D., 2023. *What is an Ultrasonic Sensor?.* [Online]
Available at: https://www.fierceelectronics.com/sensors/what-ultrasonic-sensor

LastMinuteEngineers.com, 2024. *Interfacing HC05 Bluetooth Module with Arduino.* [Online]
Available at: https://lastminuteengineers.com/hc05-bluetooth-arduino-tutorial/

Lucid Software Inc, 2024. *What is a flowchart?.* [Online]
Available at: https://www.lucidchart.com/pages/what-is-a-flowchart-tutorial#:~:text=A%20flowchart%20is%20a%20diagram,easy%2Dto%2Dunderstand%20diagrams.

Miro, 2024. *An introduction to block diagrams.* [Online]
Available at: https://miro.com/diagramming/what-is-a-block-diagram/

MYTECTUTOR, 2023. *L293D Motor Driver Shield for Arduino.* [Online]
Available at: https://mytectutor.com/l293d-motor-driver-shield-for-arduino/

OKW Gehäusesysteme, 2024. *A9156001 BATTERY HOLDER, 2 X AA.* [Online]
Available at: https://www.okw.com/en/Battery-compartments-and-holders-button-cell-holders/A9156001.htm

RS Components Ltd, 2024. *MIKROE-512, 150mm Insulated Breadboard Jumper Wire in Black, Blue, Brown, Green, Grey, Orange, Purple, Red, White,.* [Online]
Available at: https://uk.rs-online.com/web/p/breadboard-jumper-wires/7916454

Wiltronics Research Pty Ltd , 2023. *What Are Jumper Wires: Know by Colour, Types and Uses.* [Online]
Available at: https://www.wiltronics.com.au/wiltronics-knowledge-base/what-are-jumper-wires/

Zhaowei Machinery & Electronics Co., 2023. *What are DC Gear Motors, and How Do They Work?.* [Online]
Available at: https://www.zwgearbox.com/blog/what-is-dc-gear-motor-and-how-does-it-work

8. Appendix

8.1 Source Code

- Source Code for Bluetooth/ Voice Control

```
#include <AFMotor.h>
#include <SoftwareSerial.h>

SoftwareSerial bluetoothSerial(9, 10); // RX, TX

//initial motors pin
AF_DCMotor motor1(1, MOTOR12_1KHZ);
AF_DCMotor motor2(2, MOTOR12_1KHZ);
AF_DCMotor motor3(3, MOTOR34_1KHZ);
AF_DCMotor motor4(4, MOTOR34_1KHZ);

char command;

void setup()
{
  bluetoothSerial.begin(9600);  //Set the baud rate to your Bluetooth module.
}

void loop() {
  if (bluetoothSerial.available() > 0) {
    command = bluetoothSerial.read();

    Stop(); //initialize with motors stoped
```

```
    switch (command) {
      case 'U':
        forward();
        break;
      case 'D':
        back();
        break;
      case 'L':
        left();
        break;
      case 'R':
        right();
        break;
    }
  }
}

void forward()
{
  motor1.setSpeed(255); //Define maximum velocity
  motor1.run(FORWARD);  //rotate the motor clockwise
  motor2.setSpeed(255); //Define maximum velocity
  motor2.run(FORWARD);  //rotate the motor clockwise
  motor3.setSpeed(255); //Define maximum velocity
  motor3.run(BACKWARD);  //rotate the motor clockwise
  motor4.setSpeed(255); //Define maximum velocity
  motor4.run(FORWARD);  //rotate the motor clockwise
```

```
}

void back()
{
  motor1.setSpeed(255); //Define maximum velocity
  motor1.run(BACKWARD); //rotate the motor anti-clockwise
  motor2.setSpeed(255); //Define maximum velocity
  motor2.run(BACKWARD); //rotate the motor anti-clockwise
  motor3.setSpeed(255); //Define maximum velocity
  motor3.run(FORWARD); //rotate the motor anti-clockwise
  motor4.setSpeed(255); //Define maximum velocity
  motor4.run(BACKWARD); //rotate the motor anti-clockwise
}

void left()
{
  motor1.setSpeed(255); //Define maximum velocity
  motor1.run(BACKWARD); //rotate the motor anti-clockwise
  motor2.setSpeed(255); //Define maximum velocity
  motor2.run(BACKWARD); //rotate the motor anti-clockwise
  motor3.setSpeed(255); //Define maximum velocity
  motor3.run(BACKWARD);  //rotate the motor clockwise
  motor4.setSpeed(255); //Define maximum velocity
  motor4.run(FORWARD);  //rotate the motor clockwise
}

void right()
{
```

```
  motor1.setSpeed(255); //Define maximum velocity
  motor1.run(FORWARD);  //rotate the motor clockwise
  motor2.setSpeed(255); //Define maximum velocity
  motor2.run(FORWARD);  //rotate the motor clockwise
  motor3.setSpeed(255); //Define maximum velocity
  motor3.run(FORWARD); //rotate the motor anti-clockwise
  motor4.setSpeed(255); //Define maximum velocity
  motor4.run(BACKWARD); //rotate the motor anti-clockwise
}

void Stop()
{
  motor1.setSpeed(0);  //Define minimum velocity
  motor1.run(RELEASE); //stop the motor when release the button
  motor2.setSpeed(0);  //Define minimum velocity
  motor2.run(RELEASE); //rotate the motor clockwise
  motor3.setSpeed(0);  //Define minimum velocity
  motor3.run(RELEASE); //stop the motor when release the button
  motor4.setSpeed(0);  //Define minimum velocity
  motor4.run(RELEASE); //stop the motor when release the button
}
```

- Source Code for Obstacle Detection

```
#include <Servo.h>
#include <AFMotor.h>
#define Echo A0
#define Trig A1
#define motor 10
#define spoint 103
int length;
int Left;
int Right;
int Lt = 0;
int Rt = 0;
int L1 = 0; //Lt1
int R1 = 0; // Rt1
Servo servo;
//Motors pin initial
AF_DCMotor dcmotor1(1, MOTOR12_1KHZ);
AF_DCMotor dcmotor2(2, MOTOR12_1KHZ);
AF_DCMotor dcmotor3(3, MOTOR34_1KHZ);
AF_DCMotor dcmotor4(4, MOTOR34_1KHZ);
void setup() {
 pinMode(Trig, OUTPUT);
 pinMode(Echo, INPUT);
 servo.attach(motor);
}
void loop() {
 Obstacle();
```

```
}

void Obstacle() {
  length = ultrasonic();
  if (length <= 12) {
    Stop();
    back();
    delay(500);
    Stop();
    Lt = leftsee();
    servo.write(spoint);
    delay(800);
    Rt = rightsee();
    servo.write(spoint);
    if (Lt < Rt) {
      right();
      delay(500);
      Stop();
      delay(200);
    } else if (Lt > Rt) {
      left();
      delay(500);
      Stop();
      delay(200);
    }
  } else {
    forward();
  }
```

```
}

// Function for distance reading of Ultrasonic Sensor
int ultrasonic() {
  digitalWrite(Trig, LOW);
  delayMicroseconds(4);
  digitalWrite(Trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(Trig, LOW);
  long t = pulseIn(Echo, HIGH);
  long cm = t / 29 / 2; // Distance converted time
  return cm;
}
void forward()
{
  dcmotor1.setSpeed(255); // Defining the max velocity
  dcmotor1.run(FORWARD);  // Clockwise rotation of the motor
  dcmotor2.setSpeed(255); // Defining the max velocity
  dcmotor2.run(FORWARD);  // Clockwise rotation of the motor
  dcmotor3.setSpeed(255); // Defining the max velocity
  dcmotor3.run(BACKWARD);  // Clockwise rotation of the motor
  dcmotor4.setSpeed(255); // Defining the max velocity
  dcmotor4.run(FORWARD);  // Clockwise rotation of the motor
}

void back()
{
  dcmotor1.setSpeed(255); // Defining the max velocity
```

```
  dcmotor1.run(BACKWARD); // Anti-clockwise rotation of the motor
  dcmotor2.setSpeed(255); // Defining the max velocity
  dcmotor2.run(BACKWARD); // Anti-clockwise rotation of the motor
  dcmotor3.setSpeed(255); // Defining the max velocity
  dcmotor3.run(FORWARD); // Anti-clockwise rotation of the motor
  dcmotor4.setSpeed(255); // Defining the max velocity
  dcmotor4.run(BACKWARD); // Anti-clockwise rotation of the motor
}

void left()
{
  dcmotor1.setSpeed(255); // Defining the max velocity
  dcmotor1.run(BACKWARD); // Anti-clockwise rotation of the motor
  dcmotor2.setSpeed(255); // Defining the max velocity
  dcmotor2.run(BACKWARD); // Anti-clockwise rotation of the motor
  dcmotor3.setSpeed(255); // Defining the max velocity
  dcmotor3.run(BACKWARD);  // Anti-clockwise rotation of the motor
  dcmotor4.setSpeed(255); // Defining the max velocity
  dcmotor4.run(FORWARD);  // Clockwise rotation of the motor
}

void right()
{
  dcmotor1.setSpeed(255); // Defining the max velocity
  dcmotor1.run(FORWARD);  // Clockwise rotation of the motor
  dcmotor2.setSpeed(255); // Defining the max velocity
  dcmotor2.run(FORWARD);  // Clockwise rotation of the motor
  dcmotor3.setSpeed(255); // Defining the max velocity
```

```
  dcmotor3.run(FORWARD); // Clockwise rotation of the motor
  dcmotor4.setSpeed(255); // Defining the max velocity
  dcmotor4.run(BACKWARD); // Anti-clockwise rotation of the motor
}

void Stop()
{
  dcmotor1.setSpeed(0);  // Defining the min velocity
  dcmotor1.run(RELEASE); // Release the button when stopping the motor
  dcmotor2.setSpeed(0);  // Defining the min velocity
  dcmotor2.run(RELEASE); // Release the button when stopping the motor
  dcmotor3.setSpeed(0);  // Defining the min velocity
  dcmotor3.run(RELEASE); // Release the button when stopping the motor
  dcmotor4.setSpeed(0);  // Defining the min velocity
  dcmotor4.run(RELEASE); // Release the button when stopping the motor
}
int rightsee() {
  servo.write(20);
  delay(800);
  Left = ultrasonic();
  return Left;
}
int leftsee() {
  servo.write(180);
  delay(800);
  Right = ultrasonic();
  return Right;
}
```

8.2 Requirement Analysis

- Arduino Uno

Arduino UNO is an open-source platform for creating electronics. An ATmega328P microprocessor serves as its foundation. The board is made up of shields, additional circuitry, and digital and analog input/output (I/O) pins. Six analog pin inputs, fourteen digital pins, a USB port, a power jack, and an ICSP (In-Circuit Serial Programming) header are all included in the Arduino UNO. The Integrated Development Environment, or IDE, is the basis for its programming. It is compatible with both offline and online environment (javaTpoint, 2021).

- L293D Motor Driver Shield

The L293D functions as a dual-channel H-Bridge motor driver with four H-Bridges in total, enabling the control of a pair of DC motors or a single stepper motor, with each H-bridge capable of delivering up to 0.6A to the motor. It is based on the L293D IC and can operate 2 stepper motors, 2 servo motors, and 4 bi-directional DC motors (MYTECTUTOR, 2023).

- HC-05 Bluetooth Module

The HC-05 employs serial communication to interface with electronics, commonly facilitating file exchange among compact devices like mobile phones via short-range wireless links. Operating within the 2.45GHz frequency band, it achieves data transfer rates up to 1 Mbps within a 10-meter range and the HC-05 module is compatible with power supplies ranging from 4 to 6 volts (geeksforgeeks, 2023).

- Ultrasonic Sensor

The ultrasonic sensor is an electronic device that uses ultrasonic sound waves to measure the distance to a target object and converts the reflected signals into electrical impulses. These ultrasonic waves travel quickly, outpacing the speed of sound. The transmitter and the receiver are the two fundamental components of the sensor. In addition to measuring distance, ultrasonic sensors are used in many other domains, including fluid level monitoring, object detection, and proximity sensing (Jost, 2023).

- Servo Motor

A servo motor is a type of electric motor that reacts to precise commands from a controller about position, speed, or torque. The servo motor, which consists of three essential components (motor, sensor, and controller), functions by processing two input signals: a feedback signal and a set point signal. By deciphering and responding to these signals, the controller guarantees accurate and fluid motor actions (Electrical4U, 2023).

- DC Geared Motor

A DC Gear Motor is a direct current speed reduction motor that combines a DC motor and gearbox. A gearbox is typically used to limit the motor shaft speed and increase the motor's output torque. DC gear motors use direct current power to reduce output speed and increase torque output at the same time. Higher torque can be achieved in DC geared motors due to the gears that are connected within the motor's structure (Zhaowei Machinery & Electronics Co., 2023).

- Jumper Wires

With connector pins on both ends, jumper wires are a handy tool for connecting circuits without soldering. Male-to-male, female-to-female, and male-to-female are the three variations available to meet different connectivity requirements. Additionally, these wires display two different head shapes: round and square. The male connector, also called a plug, has a solid pin for center conduction while the female connector, referred to as a jack, has a center conductor that is perforated to accommodate the male pin, allowing for flexible and adjustable electrical connections (Wiltronics Research Pty Ltd , 2023).

- Battery and Battery Holder

A battery is a component that uses chemical reactions to store electrical energy and supply power to different electronics. A battery holder, on the other hand, is a physical item that holds the battery in position and has a chamber designed to accommodate a certain kind of battery. It offers a practical and transportable power solution for projects by having electrical connections that link the battery to the electronic device or platform.

- Switch

Switches are essential components of an Arduino circuit because they allow users to toggle devices and establish connections. By adjusting the slider, the Arduino slide-switch makes it easier to switch between the open (ON) and closed (OFF) positions. This device allows or disables the circuit's current flow without the need for split wires. Slide switches effectively regulate the electrical route and are especially helpful for small-circuit applications. They are essential parts of a variety of Arduino projects (javaTpoint, 2024).

- Breadboard

All a breadboard is a piece of board used for circuit building or prototyping. It enables you to build circuits without wielding by arranging components and wiring on the board. Your connections ate handled by the holes in the breadboard, which electrically connect components or wires inside the board while also firmly grasping them where you place them. The speed and simplicity of use are excellent for learning and rapid circuit prototyping. High frequency and more intricate circuits are not as good candidates for breadboarding (Crowell, 2024).

8.3 Design Diagram

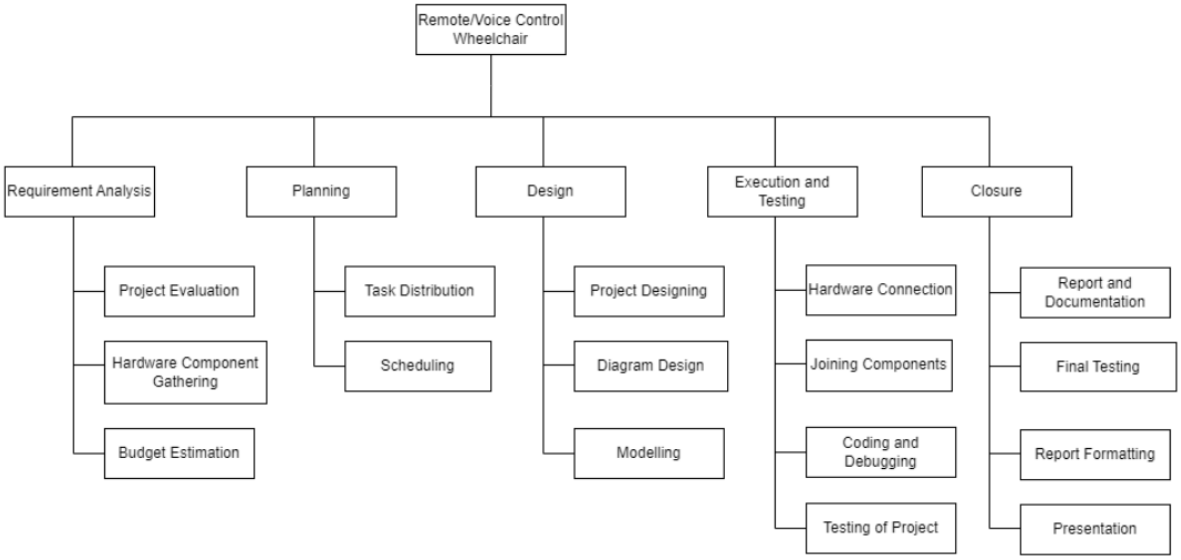

Figure 55: WBS of whole Prototype.

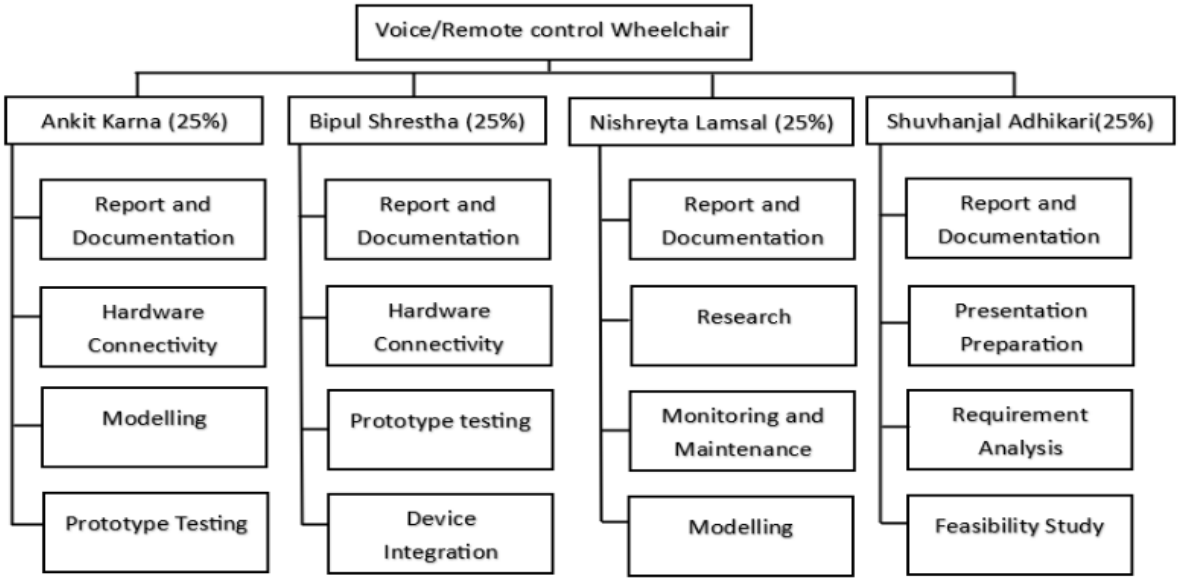

Figure 56: Individual task distribution.