Beyond Boundaries :

Empowering Mobility with ESP32 Smart Cloud Vehicles

Lab Slot : L51+L52

Faculty Name: Sangeetha.A

Registration Number: Yogeshwar.jb - 21BEC0521

Alan Samuel - 21BEC0785

Alwin Solomon- 21BEC0763

AIM :

- The aim of this project is to develop a Smart Cloud Vehicle system using ESP32 and IOT, enabling precise location tracking and User friendly UI to select target locations.

OBJECTIVE :

1. Integrate the Esp32 board with a Cloud to accurately receive real-time Directions and Distance.

2. Develop Arduino code to receive Cloud feedback and interpret them for navigation of the Vehicle for UAV purposes.

3. Implement motor control algorithms to enable precise movement of the vehicle based on the Cloud Input .

4. Conduct thorough testing and calibration to ensure the system's accuracy and reliability.

Components:

ESP32 Board: The brain of the system, responsible for controlling and coordinating the various components.

Motor Driver: Enables control of the motors that drive the vehicle's movement.

Motors , Servo: Depending on the vehicle setup, DC motors or stepper motors can be used to control the movement.

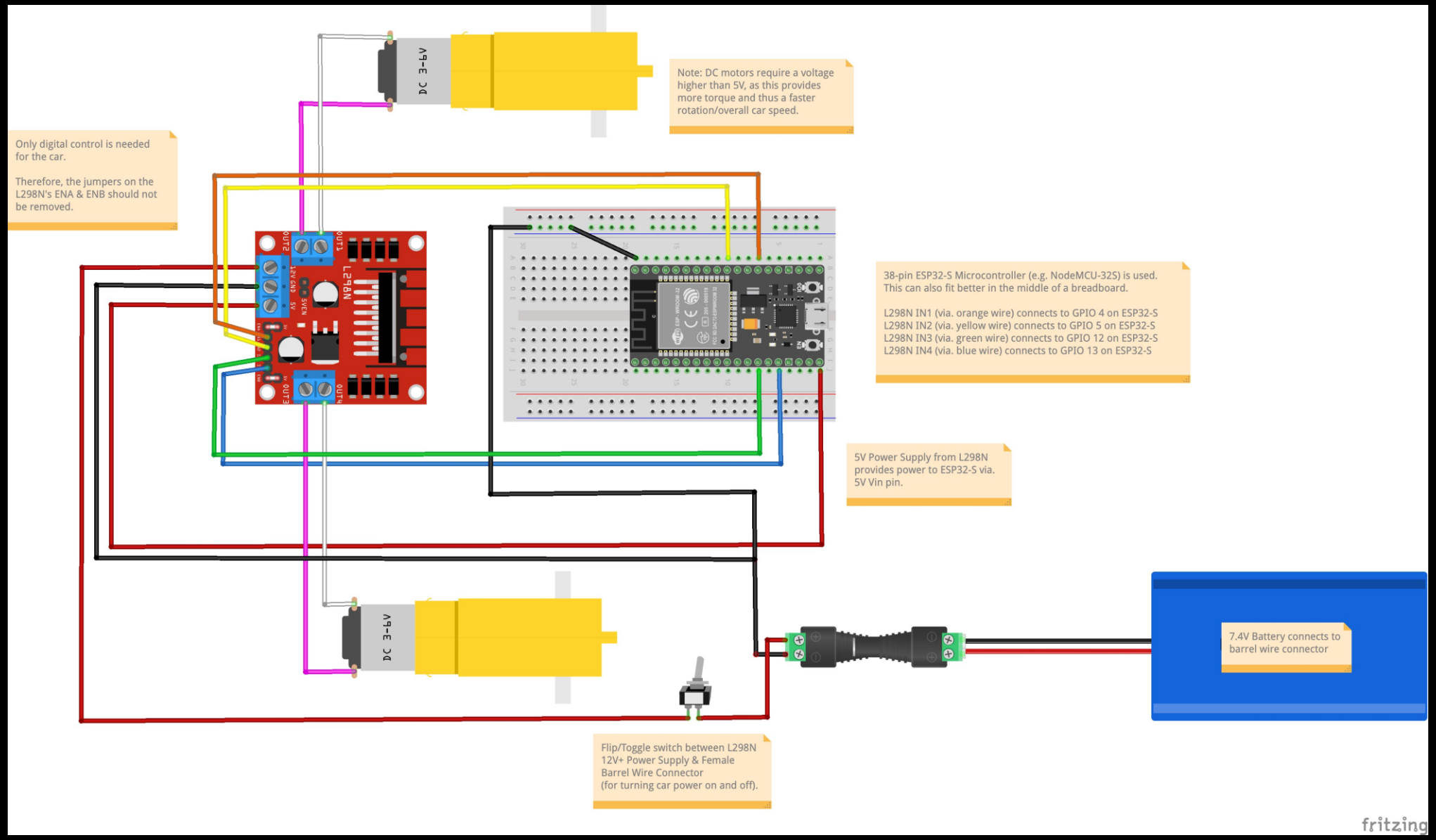
Sensors: Additional sensors may be required for obstacle detection and collision avoidance, such as ultrasonic sensors or infrared sensors.

Power Supply: Provides power to the Arduino board, GPS module, motor driver, and motors. It can be a battery pack or an external power source.

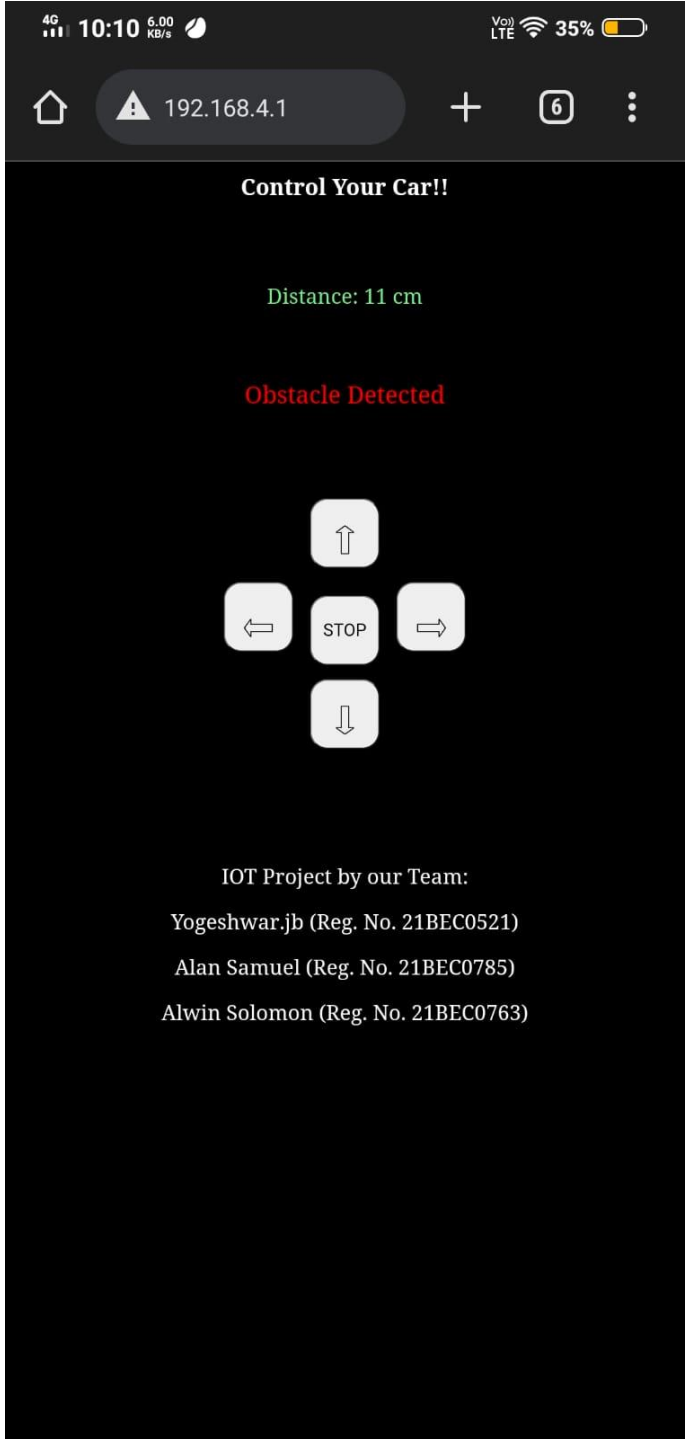
Breadboard : Used for prototyping and connecting the components together.

Jumper Wires: Connect the various components to the Arduino board and provide electrical connections.

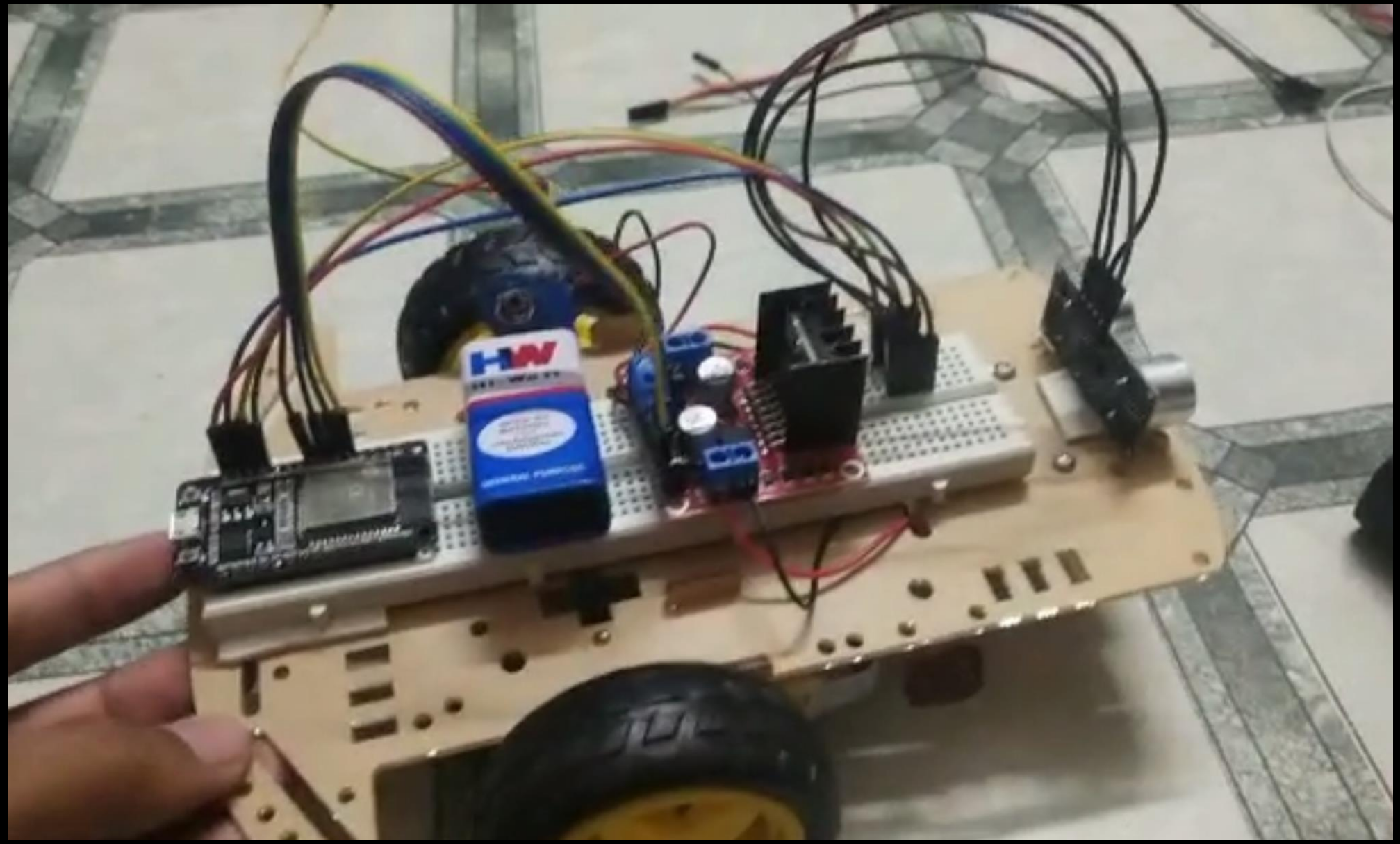
Basic Circuit with Motor:

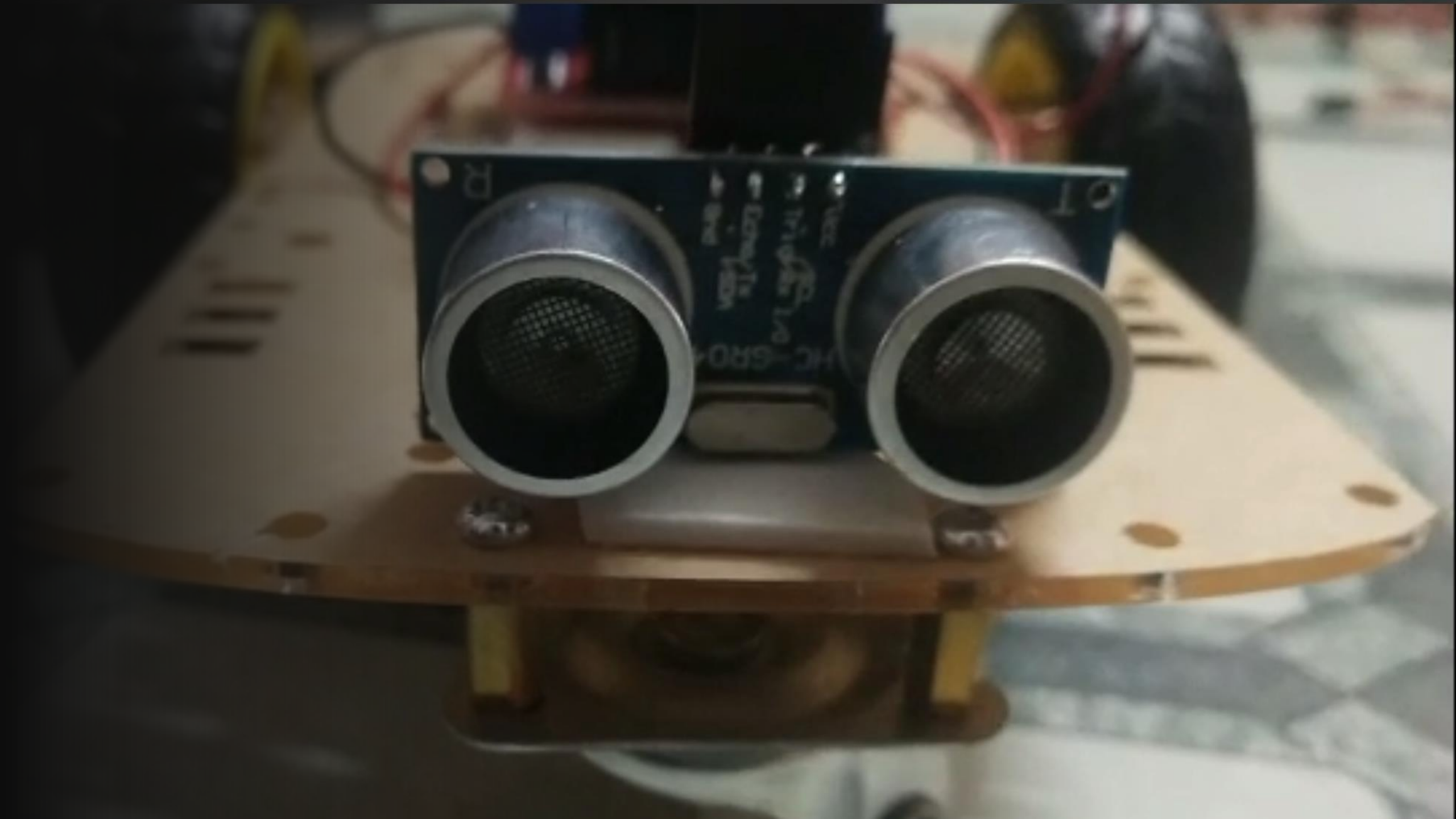


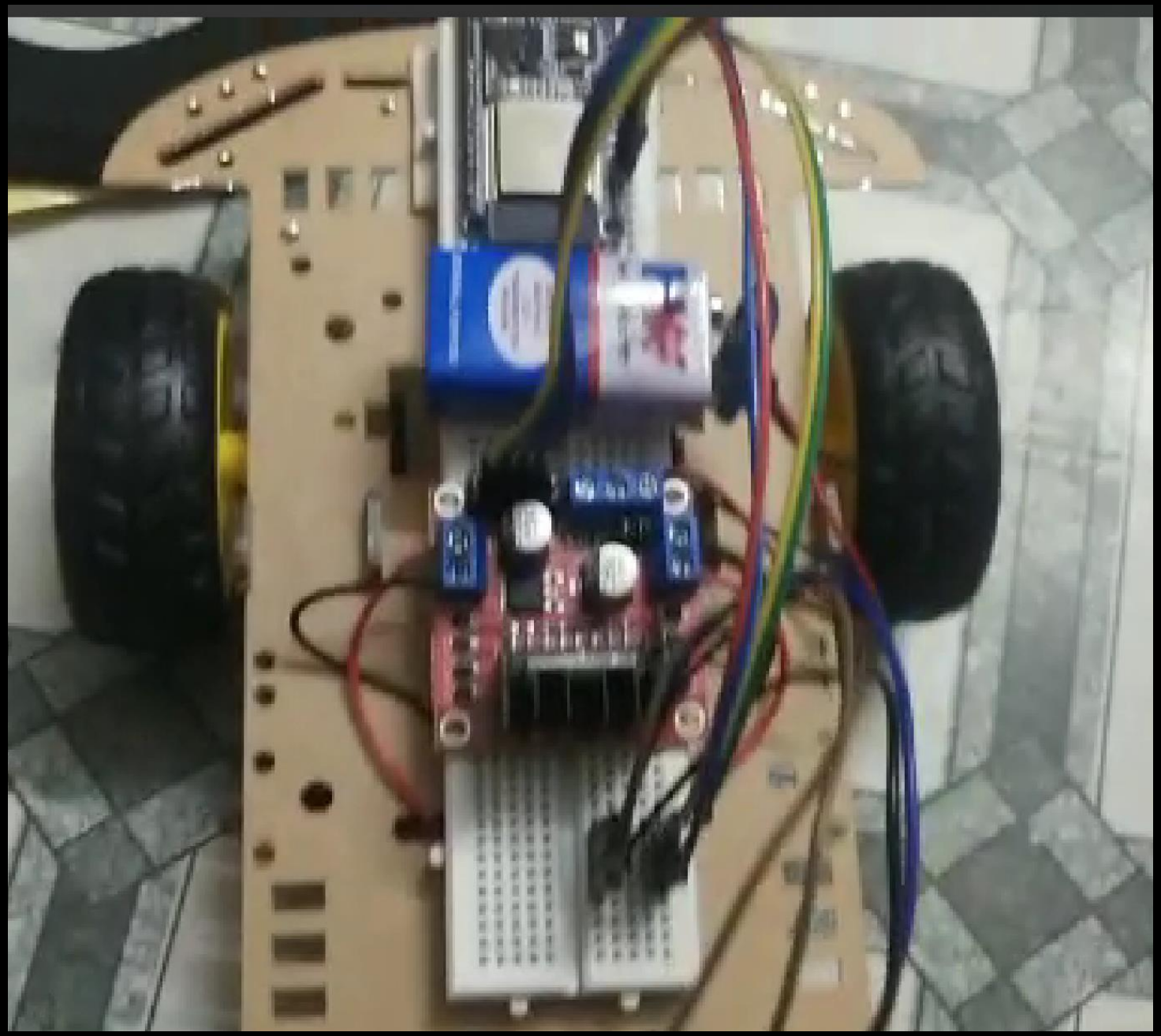
User Interface:



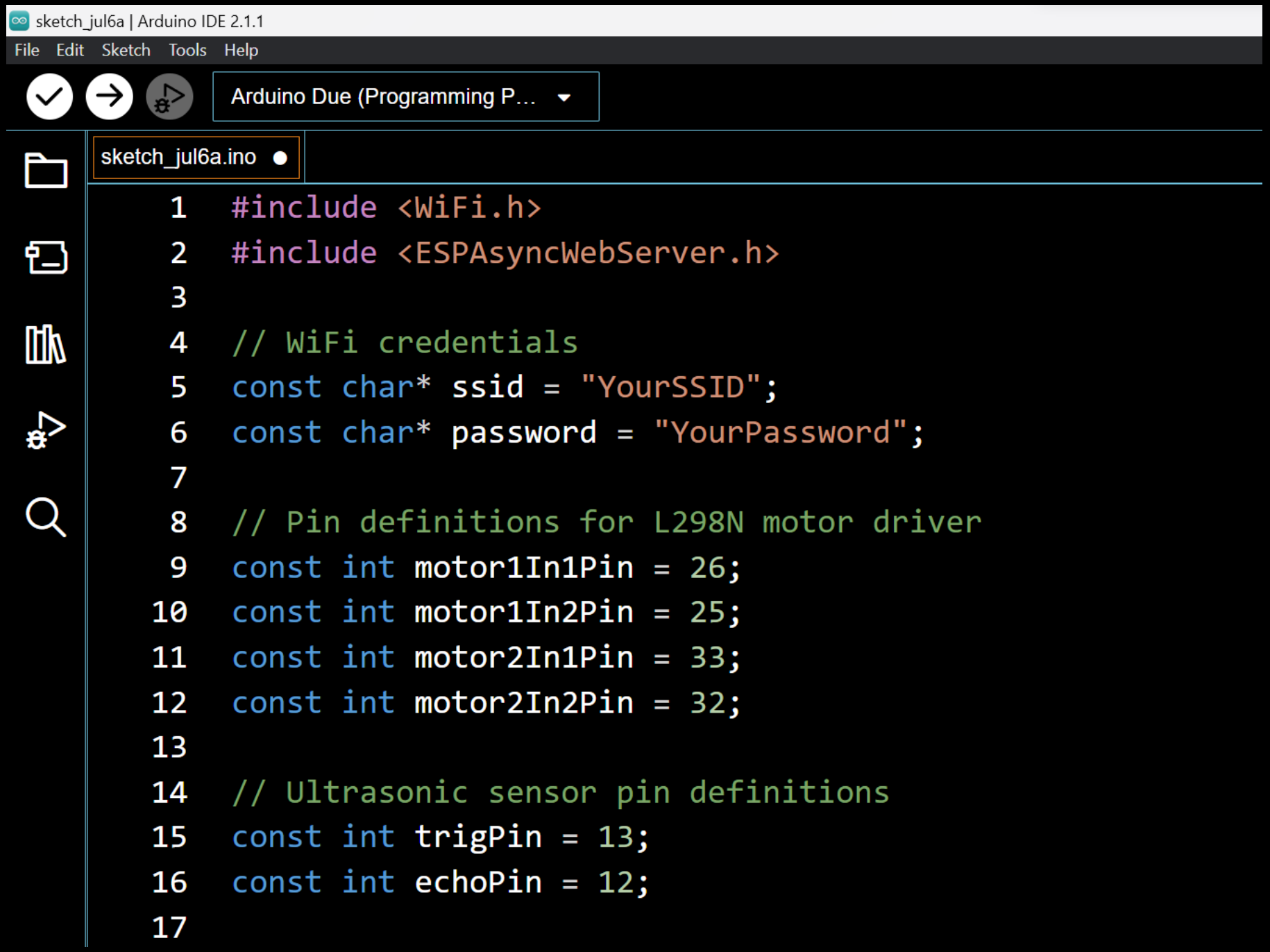
Vehicle model:




R
T



CODE :


sketch_jul6a | Arduino IDE 2.1.1
File Edit Sketch Tools Help
Arduino Due (Programming P...
sketch_jul6a.ino
#include <WiFi.h>
#include <ESPAsyncWebServer.h>
// WiFi credentials
const char* ssid = "YourSSID";
const char* password = "YourPassword";
// Pin definitions for L298N motor driver
const int motor1In1Pin = 26;
const int motor1In2Pin = 25;
const int motor2In1Pin = 33;
const int motor2In2Pin = 32;
// Ultrasonic sensor pin definitions
const int trigPin = 13;
const int echoPin = 12;

```
// HTML page content
const char* htmlPage = R"HTML(
<!DOCTYPE html>
<html>
<head>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <style>
    .arrows {
      font-size: 70px;
      color: black;
      position: center;
    }
    .stopButton {
      font-size: 25px;
      position: center;
      color: black;
    }
    .button {
      width: 100px;
      height: 100px;
      font-size: 20px;
      margin: 10px;
      border-radius: 25%;
    }
```

```
    .distance {
      color: lightgreen;
      font-size: 30px;
      text-align: center;
      margin-bottom: 10px;
    }
    .obstacle {
      color: red;
      font-size: 34px;
      text-align: center;
      margin-top: 20px;
    }
  </style>
</head>
<body class="noselect" align="center" style="background-color:black">
  <h1 style="color: white; text-align: center;">Control Your Car!!</h1>
  <p style="color: black; text-align: center;">hii</p>
  <p style="color: black; text-align: center;">hii</p>
  <p class="distance">Distance: <span id="distance">-</span></p>
  <p style="color: black; text-align: center;">hii</p>
  <p style="color: black; text-align: center;">hii</p>
  <p class="obstacle" id="obstacleText" style="display: none;">Obstacle Detected</p>
  <p style="color: black; text-align: center;">hii</p>
  <p style="color: black; text-align: center;">hii</p>
```

```
66
67      <button class="button" onclick="move('forward')"><span class="arrows">&#8679;</span></button>
68      <br>
69      <button class="button" onclick="move('left')"><span class="arrows">&#8678;</span></button>
70      <button class="button" onclick="move('stop')"><span class="stopButton">STOP</span></button>
71      <button class="button" onclick="move('right')"><span class="arrows">&#8680;</span></button>
72      <br>
73      <button class="button" onclick="move('backward')"><span class="arrows">&#8681;</span></button>
74      <p style="color: black; text-align: center;">hii</p>
75      <p style="color: black; text-align: center;">hii</p>
76      <p style="color: black; text-align: center;">hii</p>
77  <p style="color: white; text-align: center; font-size: 30px;">IOT Project by our Team:</p>
78  <p style="color: white; text-align: center; font-size: 30px;">Yogeshwar.jb (Reg. No. 21BEC0521)</p>
79  <p style="color: white; text-align: center; font-size: 30px;">Alan Samuel (Reg. No. 21BEC0785)</p>
80  <p style="color: white; text-align: center; font-size: 30px;">Alwin Solomon (Reg. No. 21BEC0763)</p>
81
82
83      <script>
84          function move(direction) {
85              var xhttp = new XMLHttpRequest();
86              xhttp.open("GET", "/move?dir=" + direction, true);
87              xhttp.send();
88          }
89
```

```
    function updateDistance(distance) {
      var distanceElement = document.getElementById("distance");
      var obstacleTextElement = document.getElementById("obstacleText");

      distanceElement.textContent = distance + " cm";

      if (distance < 10) {
        obstacleTextElement.style.display = "block";
      } else {
        obstacleTextElement.style.display = "none";
      }
    }

    setInterval(function() {
      var xhttp = new XMLHttpRequest();
      xhttp.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
          var distance = parseInt(this.responseText);
          updateDistance(distance);
        }
      };
      xhttp.open("GET", "/getDistance", true);
      xhttp.send();
    }, 1000);
  </script>
</body>
</html>
```

```
121
122 AsyncWebServer server(80);
123
124 void setup() {
125   Serial.begin(115200);
126
127   WiFi.softAP(ssid, password);
128   IPAddress IP = WiFi.softAPIP();
129   Serial.print("AP IP address: ");
130   Serial.println(IP);
131
132   pinMode(motor1In1Pin, OUTPUT);
133   pinMode(motor1In2Pin, OUTPUT);
134   pinMode(motor2In1Pin, OUTPUT);
135   pinMode(motor2In2Pin, OUTPUT);
136
137   pinMode(trigPin, OUTPUT);
138   pinMode(echoPin, INPUT);
139
140   server.on("/", HTTP_GET, [](AsyncWebServerRequest *request){
141     request->send(200, "text/html", htmlPage);
142   });
143
144   server.on("/move", HTTP_GET, [](AsyncWebServerRequest *request){
145     String direction = request->getParam("dir")->value();
146
```

```
144  server.on("/move", HTTP_GET, [](AsyncWebServerRequest *request){
145    String direction = request->getParam("dir")->value();
146
147    if (direction == "forward") {
148      digitalWrite(motor1In1Pin, HIGH);
149      digitalWrite(motor1In2Pin, LOW);
150      digitalWrite(motor2In1Pin, HIGH);
151      digitalWrite(motor2In2Pin, LOW);
152    } else if (direction == "backward") {
153      digitalWrite(motor1In1Pin, LOW);
154      digitalWrite(motor1In2Pin, HIGH);
155      digitalWrite(motor2In1Pin, LOW);
156      digitalWrite(motor2In2Pin, HIGH);
157    } else if (direction == "left") {
158      digitalWrite(motor1In1Pin, LOW);
159      digitalWrite(motor1In2Pin, HIGH);
160      digitalWrite(motor2In1Pin, HIGH);
161      digitalWrite(motor2In2Pin, LOW);
162    } else if (direction == "right") {
163      digitalWrite(motor1In1Pin, HIGH);
164      digitalWrite(motor1In2Pin, LOW);
165      digitalWrite(motor2In1Pin, LOW);
166      digitalWrite(motor2In2Pin, HIGH);
```

```
  } else if (direction == "stop") {
    digitalWrite(motor1In1Pin, LOW);
    digitalWrite(motor1In2Pin, LOW);
    digitalWrite(motor2In1Pin, LOW);
    digitalWrite(motor2In2Pin, LOW);
  }

  request->send(200, "text/plain", "OK");
});

server.on("/getDistance", HTTP_GET, [](AsyncWebServerRequest *request){
  long duration, distance;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = duration * 0.034 / 2;
```

```
186
187       if (distance < 10) {
188         digitalWrite(motor1In1Pin, LOW);
189         digitalWrite(motor1In2Pin, LOW);
190         digitalWrite(motor2In1Pin, LOW);
191         digitalWrite(motor2In2Pin, LOW);
192       }
193
194       request->send(200, "text/plain", String(distance));
195
196     });
197
198     server.begin();
199   }
200
201   void loop() {
202     // Code in the loop, if any
203   }
```

Source and Reference:

Form the following sites we have taken Gps Implementation and Esp32:

https://en.wikipedia.org/wiki/ESP32

https://how2electronics.com/esp32-gps-tracker-using-l86-gps-module-oled-display/

https://github.com/rakesh-i/ESP32-Autonomous-car

https://github.com/niekky/ESP32-Autonomous-car

https://www.reddit.com/r/esp32/comments/nqqt4v/autonomous_car_using_esp32_and_esp32_cam/

IEEE Papers:

https://ieeexplore-ieee-org.egateway.vit.ac.in/document/9725526

(Meenu Gupta; Rakesh Kumar; Raju Kumar Chaudhary; Jayshree Kumari)

https://ieeexplore-ieee-org.egateway.vit.ac.in/document/10113098

(Siddesh G K; Rakesh Kumar Patel; Sayan Maitra; Sabitabrata Bhattacharya; Shaik Moosa; Pattubala Pavan)

Referred Youtube Channels :

1. Steven Gong
2. Test Flight