**#include <SoftwareSerial.h>**

**// Bluetooth connections**

**const int bluetoothTx = 2;**

**const int bluetoothRx = 3;**

**SoftwareSerial bluetooth(bluetoothTx, bluetoothRx);**

**// Motor control pins**

**const int ENA = 7; // Speed control for Motor1 (PWM)**

**const int Motor1A = 5;**

**const int Motor1B = 6;**

**const int ENB = 11; // Speed control for Motor2 (PWM)**

**const int Motor2A = 9;**

**const int Motor2B = 10;**

**// Buzzer pin (Horn)**

**const int hornPin = 12;**

**// Speed control variable**

**int speedValue = 250; // Adjust speed (0-255)**

**char currentCommand = 'S'; // Default stop**

**void setup() {**

**Serial.begin(9600);**

**bluetooth.begin(9600);**

**// Set motor pins as outputs**

**pinMode(ENA, OUTPUT);**

**pinMode(Motor1A, OUTPUT);**

**pinMode(Motor1B, OUTPUT);**

**pinMode(ENB, OUTPUT);**

**pinMode(Motor2A, OUTPUT);**

**pinMode(Motor2B, OUTPUT);**

**// Set horn pin as output**

**pinMode(hornPin, OUTPUT);**

**stopMotors();**

**stopHorn();**

**Serial.println("Bluetooth Control Ready (Push Button Mode)");**

**}**

**void loop() {**

**if (bluetooth.available()) {**

**currentCommand = bluetooth.read();**

**Serial.print("Command received: ");**

**Serial.println(currentCommand);**

**}**

**executeCommand(currentCommand);**

**if (!bluetooth.available()) {**

**stopMotors();**

**}**

**}**

**// Execute movement commands**

**void executeCommand(char command) {**

**switch (command) {**

**case 'F': moveForward(); break;**

**case 'B': moveBackward(); break;**

**case 'L': turnLeft(); break;**

**case 'R': turnRight(); break;**

**case 'W': activateHorn(); break;**

**default: stopMotors(); break;**

**}**

**}**

**// Stop all motors**

**void stopMotors() {**

**analogWrite(ENA, 0);**

**analogWrite(ENB, 0);**

**digitalWrite(Motor1A, LOW);**

**digitalWrite(Motor1B, LOW);**

**digitalWrite(Motor2A, LOW);**

**digitalWrite(Motor2B, LOW);**

**Serial.println("Motors stopped");**

**}**

**// Move forward with speed control**

**void moveForward() {**

**analogWrite(ENA, speedValue);**

**analogWrite(ENB, speedValue);**

**digitalWrite(Motor1A, HIGH);**

**digitalWrite(Motor1B, LOW);**

**digitalWrite(Motor2A, HIGH);**

**digitalWrite(Motor2B, LOW);**

**Serial.println("Moving forward at speed " + String(speedValue));**

**}**

**// Move backward with speed control**

**void moveBackward() {**

**analogWrite(ENA, speedValue);**

**analogWrite(ENB, speedValue);**

**digitalWrite(Motor1A, LOW);**

**digitalWrite(Motor1B, HIGH);**

**digitalWrite(Motor2A, LOW);**

**digitalWrite(Motor2B, HIGH);**

**Serial.println("Moving backward at speed " + String(speedValue));**

**}**

**// Turn left**

**void turnLeft() {**

**analogWrite(ENA, speedValue);**

**analogWrite(ENB, speedValue);**

**digitalWrite(Motor1A, HIGH);**

**digitalWrite(Motor1B, LOW);**

**digitalWrite(Motor2A, LOW);**

**digitalWrite(Motor2B, HIGH);**

**Serial.println("Turning left");**

**}**

**// Turn right**

**void turnRight() {**

**analogWrite(ENA, speedValue);**

**analogWrite(ENB, speedValue);**

**digitalWrite(Motor1A, LOW);**

**digitalWrite(Motor1B, HIGH);**

**digitalWrite(Motor2A, HIGH);**

**digitalWrite(Motor2B, LOW);**

**Serial.println("Turning right");**

**}**

**// Activate horn (Buzzer)**

**void activateHorn() {**

**digitalWrite(hornPin, HIGH);**

**Serial.println("Horn activated (Buzzer sound)");**

**delay(1000);**

**stopHorn();**

**}**

**// Stop the horn (Buzzer)**

**void stopHorn() {**

**digitalWrite(hornPin, LOW);**

**Serial.println("Horn deactivated (Buzzer off)");**

**}**