```
/*
SENSOR-BASED PARKING + MOTOR CONTROL SYSTEM
Hardware:
- Arduino Mega 2560
- 4 Ultrasonic Sensors (HC-SR04)
- GPS Module (NEO-6M)
- Bluetooth HC-05
- L298N Motor Driver (2 DC motors)
- Buzzer, LEDs
*/

#include <SoftwareSerial.h>

// === Bluetooth (SoftwareSerial) ===
SoftwareSerial BT(10, 11);  // RX, TX

// === GPS (SoftwareSerial) ===
SoftwareSerial GPS(12, 13); // RX, TX

// === Ultrasonic Sensors ===
const int trigFront = 32, echoFront = 33;
const int trigBack  = 34, echoBack  = 35;
const int trigLeft  = 36, echoLeft  = 37;
const int trigRight = 38, echoRight = 39;

// === Indicators ===
const int buzzerPin = 30;
const int ledFront = 31;
const int ledBack  = 40;
```

```
const int ledSide  = 41;

// === Motor Driver (L298N) ===
const int IN1 = 22;
const int IN2 = 23;
const int IN3 = 24;
const int IN4 = 25;
const int ENA = 6;  // PWM
const int ENB = 7;  // PWM



// === Read distance ===
int getDistance(int trigPin, int echoPin) {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  long duration = pulseIn(echoPin, HIGH, 30000);
  if (duration == 0) return 500;

  return (duration * 0.034 / 2);
}



// === Setup ===
void setup() {
  Serial.begin(9600);
```

```
  BT.begin(9600);
  GPS.begin(9600);

  // Ultrasonic Pins
  pinMode(trigFront, OUTPUT); pinMode(echoFront, INPUT);
  pinMode(trigBack,  OUTPUT); pinMode(echoBack,  INPUT);
  pinMode(trigLeft,  OUTPUT); pinMode(echoLeft,  INPUT);
  pinMode(trigRight, OUTPUT); pinMode(echoRight, INPUT);

  // Indicators
  pinMode(buzzerPin, OUTPUT);
  pinMode(ledFront, OUTPUT);
  pinMode(ledBack,  OUTPUT);
  pinMode(ledSide,  OUTPUT);

  // Motor Driver
  pinMode(IN1, OUTPUT); pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT); pinMode(IN4, OUTPUT);
  pinMode(ENA, OUTPUT); pinMode(ENB, OUTPUT);

  Serial.println("System Initializing...");
}


// === Motor Movement Functions ===
void moveForward(int speed = 180) {
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH);
```

```
  digitalWrite(IN4, LOW);
  analogWrite(ENA, speed);
  analogWrite(ENB, speed);
}

void moveBackward(int speed = 180) {
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
  analogWrite(ENA, speed);
  analogWrite(ENB, speed);
}

void stopMotors() {
  analogWrite(ENA, 0);
  analogWrite(ENB, 0);
}


// === LED + Buzzer + Motor Reaction ===
void processSensor(int distance, int ledPin) {

  if (distance <= 15) {
    digitalWrite(ledPin, HIGH);
    digitalWrite(buzzerPin, HIGH);
    stopMotors();

  } else if (distance <= 30) {
```

```
    digitalWrite(ledPin, HIGH);
    tone(buzzerPin, 2100, 150);
    stopMotors();

  } else if (distance <= 60) {
    digitalWrite(ledPin, HIGH);
    tone(buzzerPin, 1400, 250);

  } else {
    digitalWrite(ledPin, LOW);
  }
}


// === Main Loop ===
void loop() {

  int dFront = getDistance(trigFront, echoFront);
  int dBack  = getDistance(trigBack,  echoBack);
  int dLeft  = getDistance(trigLeft,  echoLeft);
  int dRight = getDistance(trigRight, echoRight);

  // Serial Output
  Serial.print("F:"); Serial.print(dFront);
  Serial.print(" B:"); Serial.print(dBack);
  Serial.print(" L:"); Serial.print(dLeft);
  Serial.print(" R:"); Serial.println(dRight);

  // Send to Bluetooth
```

```
  BT.print("Front:"); BT.print(dFront);
  BT.print("cm Back:"); BT.print(dBack);
  BT.print("cm Left:"); BT.print(dLeft);
  BT.print("cm Right:"); BT.println(dRight);

  // GPS Forwarding to Bluetooth
  if (GPS.available()) {
    char c = GPS.read();
    BT.write(c);
  }

  // Motor Logic (simple demo)
  if (dFront > 40) {
    moveForward();
  } else {
    stopMotors();
  }

  // Warning System
  processSensor(dFront, ledFront);
  processSensor(dBack,  ledBack);
  processSensor(dLeft,  ledSide);
  processSensor(dRight, ledSide);

  delay(200);
}
```