NIYIGENA PATRICK

Project title: Smart Parking Assistant System

Project Overview
This project implements a simple **parking management system** using an ultrasonic sensor to detect cars, a servo motor as the gate, an LCD display for status messages, and LEDs to indicate slot availability.

Components Required

- **Arduino UNO / compatible board**
- **Ultrasonic Sensor (HC-SR04)**
- **Servo motor (SG90 or similar)**
- **LCD 16x2 with I²C adapter** (address 0x20 or 0x3F depending on module)
- **Red LED** (indicates slot occupied)
- **Green LED** (indicates slot available)
- **Resistors (220Ω)** for LEDs
- **Jumper wires** and **breadboard** / PCB

Pin Connections

Component Pin	on Arduino
Ultrasonic Trig	D6
Ultrasonic Echo	D7
Red LED	D8
Green LED	D9
Servo Signal	D10
LCD (I²C)	A4 (SDA), A5 (SCL)

Power: 5V from Arduino powers the ultrasonic sensor, LCD, and servo (ensure servo current is supported). All grounds (GND) must be connected.

How It Works

1. **Distance Measurement**
    - The HC-SR04 ultrasonic sensor measures the distance to detect if a car is

present.
    - If distance < **20 cm (threshold)** → car is detected.
2. **LED Indicators**
    - **Red LED ON**: Slot occupied.
    - **Green LED ON**: Slot free.
3. **Servo Gate**
    - When a car is detected: Servo rotates to 90° → gate opens.
    - When no car: Servo rotates back to 0° → gate closed.
4. **LCD Display Messages**
    - Shows “Parking FULL” when a car is present.
    - Shows “Slot Available” when no car is detected.
    - Second line displays measured distance in cm.

Code Explanation

- **Libraries used**:
    - `LiquidCrystal_I2C.h` → controls the 16x2 LCD via I²C.
    - `Servo.h` → drives the gate servo.
- **Functions**:
    - `readDistanceCm()` → triggers the ultrasonic sensor and calculates distance.
    - `setup()` → initializes LCD, sets pin modes, and closes gate initially. ○ `loop()` → continuously reads distance, updates LEDs, moves servo, and prints status on LCD.
- **Key variables**:
    - `threshold = 20` cm → minimum distance to detect a car.
    - `distanceCm` → stores live distance readings.

Testing Procedure

1. Upload code to Arduino using Arduino IDE.

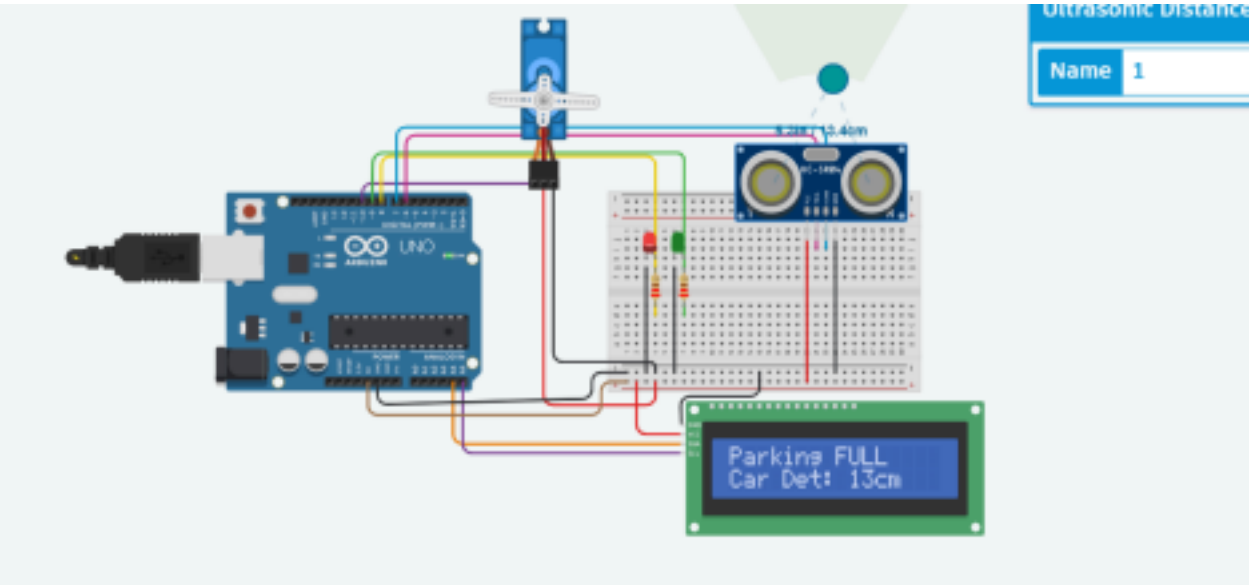
Code (Simplified)

```
#include <LiquidCrystal_I2C.h>
#include <Servo.h>


#define LCD_ADDR 0x20 // Or 0x3F depending on your I2C module
LiquidCrystal_I2C lcd(LCD_ADDR, 16, 2);
```

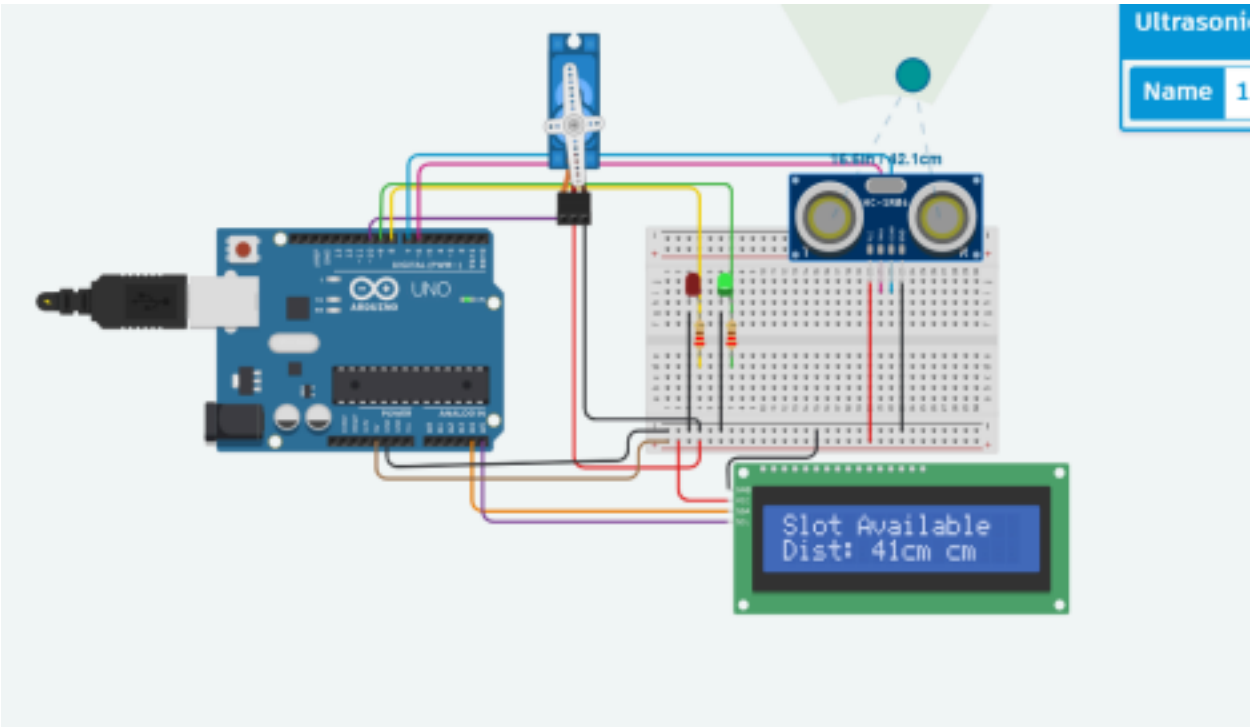
```
Servo gateServo;
const int trigPin = 6; // Ultrasonic Trigger
const int echoPin = 7; // Ultrasonic Echo
const int redLED = 8; // Slot Occupied
const int greenLED = 9; // Slot Free
const int servoPin = 10; // Servo signal pin


int distanceCm = 0;
int threshold = 20; // Distance in cm to detect car


int readDistanceCm() {
 digitalWrite(trigPin, LOW);
 delayMicroseconds(2);
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);

 long duration = pulseIn(echoPin, HIGH, 30000UL);
if (duration == 0) return 999; // out of range  return
(int)(duration * 0.034 / 2.0); // convert to cm }

void setup() {
 pinMode(trigPin, OUTPUT);
 pinMode(echoPin, INPUT);
 pinMode(redLED, OUTPUT);
 pinMode(greenLED, OUTPUT);

 gateServo.attach(servoPin);
 gateServo.write(0); // Start with gate closed

 lcd.init();
 lcd.backlight();
 lcd.setCursor(0,0);
 lcd.print("Parking System");
 delay(2000);
 lcd.clear();
```

```
}
void loop() {
 distanceCm = readDistanceCm();

 if (distanceCm < threshold) {
 // Car detected → Occupied
 digitalWrite(redLED, HIGH);
 digitalWrite(greenLED, LOW);
 gateServo.write(90); // Open gate

 lcd.setCursor(0,0);
 lcd.print("Parking FULL ");
 lcd.setCursor(0,1);
 lcd.print("Car Det: ");
 lcd.print(distanceCm);
 lcd.print("cm ");
 }
 else {
 // No car → Free
 digitalWrite(redLED, LOW);
 digitalWrite(greenLED, HIGH);
 gateServo.write(0); // Close gate

 lcd.setCursor(0,0);
 lcd.print("Slot Available ");
 lcd.setCursor(0,1);
 lcd.print("Dist: ");
 lcd.print(distanceCm);
 lcd.print("cm ");
 }

 delay(300);
}
```

2. Place an object (simulate a car) within 20 cm of the ultrasonic sensor.
    - Red LED turns ON.
    - Green LED turns OFF.
    - Servo rotates to 90° (gate opens).
    - LCD shows **“Parking FULL”** and distance.



3. Remove the object.
    - Green LED turns ON.
    - Red LED turns OFF.
    - Servo rotates to 0° (gate closed).
    - LCD shows **“Slot Available”**.



Conclusion

This project demonstrates a working prototype of a **smart parking assistant system**

with gate control, visual indicators, and live LCD feedback using Arduino and basic sensors.