Smart Parking System Using IoT

Date	**26-10-2023**
Team ID	**670**
Team Name	**Proj_223439_Team_2**
Project Name	**Smart Parking System Using IoT**

Introduction:

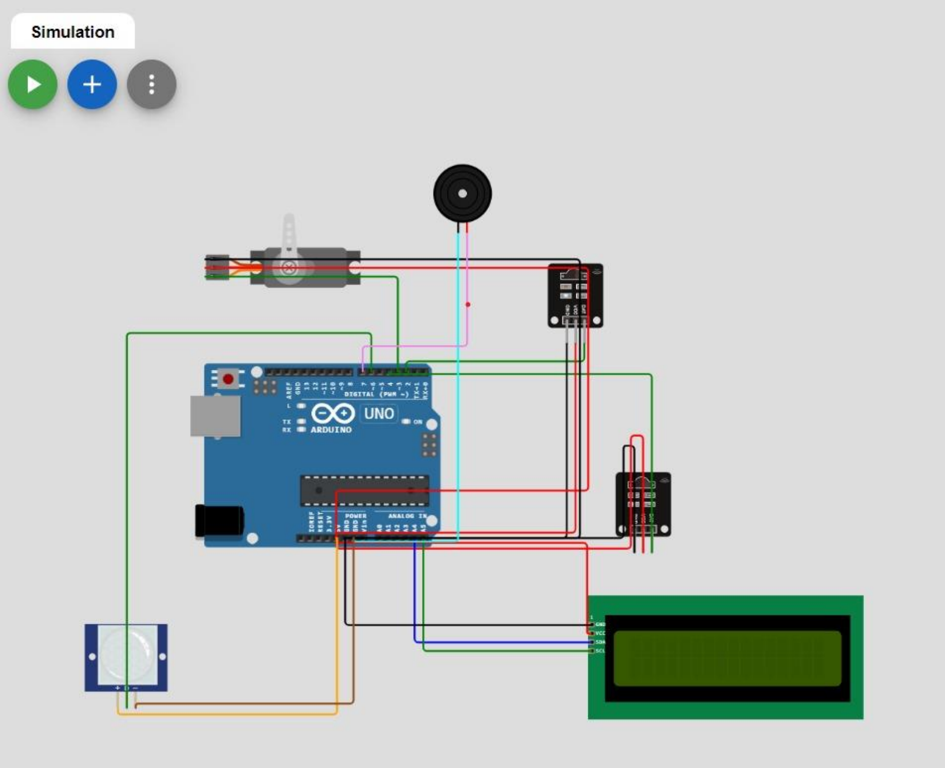
A Smart Parking System using IoT is a revolutionary technology that leverages the power of the Internet of Things to transform traditional parking management. This innovative system combines real-time sensor data and wireless connectivity to provide a seamless and efficient parking experience. By monitoring parking space availability, optimizing traffic flow, and enabling remote access, it simplifies the often frustrating and time-consuming task of finding a parking spot. Smart Parking Systems reduce congestion, enhance environmental sustainability, and improve overall urban mobility. Through sensors and connected devices, users can access real-time parking information and reserve spaces in advance, ensuring a hassle-free parking experience. This technology enhances city infrastructure, saves time, and reduces environmental impact by minimizing unnecessary traffic and emissions.

Source Code:

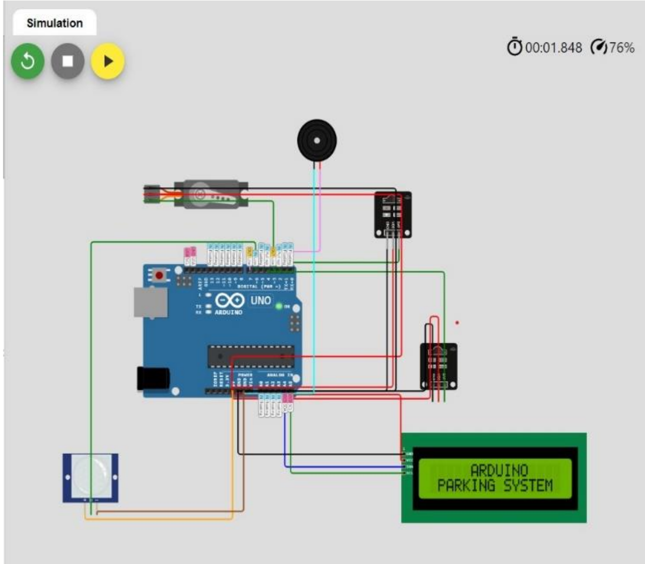
```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd (0x27, 16, 2);

#include <Servo.h>
Servo myservo1;

int IR1 = 2;
int IR2 = 4;
int SmokeDetectorPin = 6;
int BuzzerPin = 7;
int Slot = 4;

bool flag1 = false;
bool flag2 = false;

unsigned long lastLcdUpdate = 0;
unsigned long lcdUpdateInterval = 1000;

void setup () {
  lcd. begin (16, 2);
  lcd. backlight ();
  pinMode (IR1, INPUT);
  pinMode (IR2, INPUT);
  pinMode (SmokeDetectorPin, INPUT);
  pinMode (BuzzerPin, OUTPUT);

  myservo1.attach(3);
  myservo1.write(100);

  lcd. setCursor (0, 0);
  lcd.print ("     ARDUINO    ");
  lcd. setCursor (0, 1);
  lcd.print (" PARKING SYSTEM ");
  delay (2000);
  lcd. clear ();

  Serial.begin(9600);
}

void loop () {
  if (digitalRead (IR1) == LOW &&! flag1) {
    if (Slot > 0) {
      flag1 = true;
      if (! flag2) {
        myservo1.write(0);
        Slot--;
      }
    } else {
      displayMessage ("    SORRY :(    ", “Parking Full “);
    }
  }

  if (digitalRead (IR2) == LOW &&! flag2) {
    flag2 = true;
    if (! flag1) {
      myservo1.write(0);
      Slot++;
    }
  }

  if (flag1 && flag2) {
    delay (1000);
    myservo1.write(100);
    Serial.println("Servo returned to initial position.");
    flag1 = false;
    flag2 = false;
  }

  if (millis () - lastLcdUpdate >= lcdUpdateInterval) {
    updateLcdDisplay ();
    lastLcdUpdate = millis ();
  }
}

void updateLcdDisplay () {
  if (digitalRead (SmokeDetectorPin) == HIGH) {
    displayMessage ("   WARNING!   ", " Smoke Detected ");
    digitalWrite (BuzzerPin, HIGH);
  } else {
    displayMessage ("    WELCOME!    ", "Slot Left: " + String (Slot));
    digitalWrite (BuzzerPin, LOW);
  }
}
void displayMessage (const char *line1, const String &line2) {
  lcd. clear ();
  lcd. setCursor (0, 0);
  lcd.print(line1);
  lcd. setCursor (0, 1);
  lcd.print(line2);
}
```

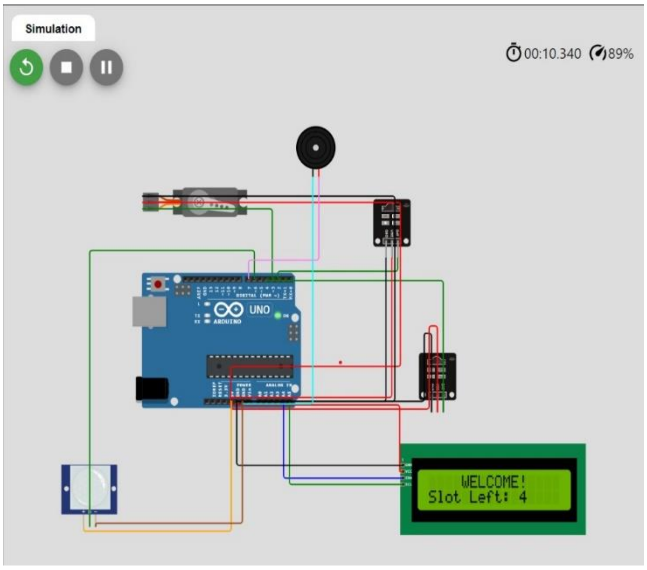
Circuit Diagram:

Before Simulation:



After Simulation:





Web Development Platform:

Html:

```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
  <div class="container">
    <h1 class="title">Parking System</h1>
    <div class="parking-info">
      <p>Slots Left: <span id="slotCount">0</span></p>
    </div>
    <div class="sensors">
      <div class="sensor">
        <p>IR Sensor 1</p>
        <button id="irSensor1Button" class="sensor-button">Vacant</button>
      </div>
      <div class="sensor">
```

```
        <p>IR Sensor 2</p>
        <button id="irSensor2Button" class="sensor-button">Vacant</button>
      </div>
    </div>
    <div class="smoke-sensor">
      <p>Smoke Detector</p>
      <button id="smokeSensorButton" class="sensor-button">OK</button>
    </div>
  </div>
  <script src="script.js"></script>
</body>
</html>
```

Css:

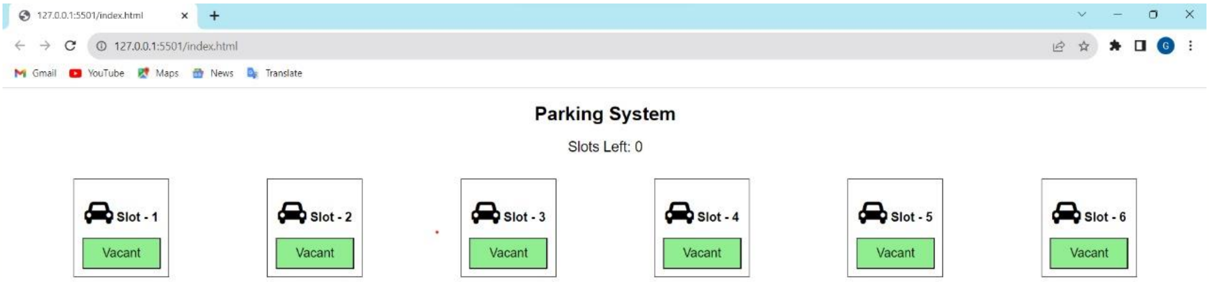
```
body {
  font-family: Arial, sans-serif;
}
.container {
  text-align: center;
  margin: 20px;
}
. title {
  font-size: 24px;
}
. parking-info {
  font-size: 18px;
}
. sensors {
  display: flex;
  justify-content: space-around;
  margin-top: 20px;
}
. sensor {
  border: 1px solid #333;
  padding: 10px;
  margin: 10px;
}
. sensor-button {
  width: 100px;
  height: 40px;
  background-color: lightgreen;
  font-size: 16px;
  cursor: pointer;
}
. smoke-sensor {
  margin-top: 20px;
}
. sensor-button. ok {
  background-color: lightgreen;
}
. sensor-button. alert {
background-color: red;
}
```

Javascript:

```
let slotCount = 0;
let irSensor1Occupied = false;
let irSensor2Occupied = false;
let smokeDetectorOk = true;
function updateDisplay () {
  document. getElementById("slotCount"). textContent = slotCount;
  document. getElementById("irSensor1Button"). textContent = irSensor1Occupied?
"Occupied”: "Vacant";
  document. getElementById("irSensor2Button"). textContent = irSensor2Occupied?
"Occupied”: "Vacant";
  document. getElementById("smokeSensorButton"). className = smokeDetectorOk?
"Sensor-button ok”: "sensor-button alert";
```

```
  document. getElementById("smokeSensorButton"). textContent = smokeDetectorOk? "OK”: "Smoke Detected";
}
document. getElementById("irSensor1Button"). addEventListener ("click", function () {
  if (! irSensor1Occupied) {
    irSensor1Occupied = true;
    slotCount--;
  } else {
    irSensor1Occupied = false;
    slotCount++;
  }
  updateDisplay ();
});
document. getElementById("irSensor2Button"). addEventListener ("click", function () {
  if (! irSensor2Occupied) {
    irSensor2Occupied = true;
    slotCount--;
  } else {
    irSensor2Occupied = false;
    slotCount++;
  }
  updateDisplay ();
});
document. getElementById("smokeSensorButton"). addEventListener ("click", function () {
  if (smokeDetectorOk) {
    smokeDetectorOk = false;
} else {
    smokeDetectorOk = true;
  }
  updateDisplay ();
});
updateDisplay ();
```

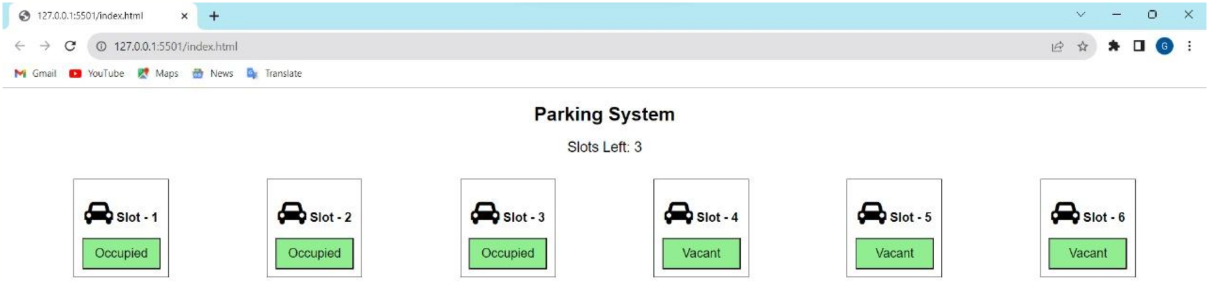
Output:



Parking System

Slots Left: 0

Slot - 1	Slot - 2	Slot - 3	Slot - 4	Slot - 5	Slot - 6
Vacant	Vacant	Vacant	Vacant	Vacant	Vacant



Parking System

Slots Left: 3

Slot - 1	Slot - 2	Slot - 3	Slot - 4	Slot - 5	Slot - 6
Occupied	Occupied	Occupied	Vacant	Vacant	Vacant

Conclusion:

The source code for Smart Parking System and simulation diagram has been successfully created. By using web development technologies like HTML, CSS, JAVASCRIPT to create a platform that displays real-time vacant slots for parking. We have designed the platform to receive and display real-time smart parking data, including occupancy rate and sensors.