**🚗 Smart Parking System using ESP32**

A real-time **Smart Parking System** powered by **ESP32** and an **ultrasonic sensor** to monitor parking spot occupancy. The system features a live web interface with dynamic updates via **WebSocket** communication, making parking management efficient and hassle-free.

**🌟 Features**

* **Ultrasonic Sensor**: Detects whether the parking spot is occupied or vacant.
* **LED Indicators**:
  + 🟥 **Red LED**: Spot is **Occupied**.
  + 🟩 **Green LED**: Spot is **Vacant**.
* **Real-Time WebSocket Updates**: Keeps the web interface synced with the parking status.
* **Browser-Based Dashboard**: User-friendly, visually appealing, and dynamic!

**🛠️ Components Required**

| **Component** | **ESP32 Pin** |
| --- | --- |
| Ultrasonic Sensor TRIG | GPIO 13 |
| Ultrasonic Sensor ECHO | GPIO 25 |
| Red LED | GPIO 2 |
| Green LED | GPIO 16 |

**🔧 Hardware Setup**

1. **Connect the Ultrasonic Sensor**:
   * TRIG to GPIO 13
   * ECHO to GPIO 25
2. **Connect the LEDs**:
   * Red LED to GPIO 2 (with 220Ω resistor)
   * Green LED to GPIO 16 (with 220Ω resistor)
3. Use a breadboard and jumper wires for easy prototyping.

**🚀 Getting Started**

**🖥️ 1. ESP32 Code Setup**

1. Clone or download this repository.
2. Open esp32\_parking.ino in the **Arduino IDE**.
3. Install the required library:
   * WebSocketsServer (via the Arduino Library Manager).
4. Update the WiFi credentials:

const char\* ssid = "Your\_SSID";

const char\* password = "Your\_PASSWORD";

1. Upload the code to your ESP32 and open the Serial Monitor to get its **IP Address**.

**🌐 2. Host the Web Page**

**Option 1: GitHub Pages (Online Hosting)**

1. Upload index.html to a GitHub repository.
2. Enable GitHub Pages under repository **Settings > Pages**.
3. Use the provided GitHub Pages URL to access the web page.
4. Update the WebSocket URL in index.html:

const ws = new WebSocket('ws://<ESP32\_IP>:81');

**Option 2: Local Hosting**

1. Save index.html in a folder.
2. Open a terminal and navigate to the folder:

cd /path/to/folder

1. Start a local server:

python -m http.server 8000

1. Open your browser and go to:

http://localhost:8000

**🛠️ Usage Instructions**

1. **Power On**: Power the ESP32 and ensure it connects to the specified WiFi network.
2. **Open the Web Page**: Access the hosted dashboard (either via GitHub Pages or locally).
3. **Monitor in Real-Time**:
   * 🟢 **Green LED**: Spot is Vacant.
   * 🔴 **Red LED**: Spot is Occupied.

**❓ Troubleshooting**

* **No WebSocket Connection**:
  + Ensure the ESP32 and the browser are on the same network.
  + Double-check the WebSocket URL in the index.html file.
* **LEDs Not Working**:
  + Verify the hardware connections.
  + Check the LED polarity and resistor values.

**✨ Contributions**

We welcome contributions! Feel free to fork the repository, create a branch, and submit a pull request.

**💡 Future Enhancements**

* Add support for multiple parking spots.
* Implement a database to track historical parking data.
* Integrate with IoT platforms for remote monitoring.