Parking Intelligent

Projet Arduino ESP8266

Auteurs : Hassan SARIH TRAORE Moussa

December 11, 2024

Introduction

Ce projet implémente un système de stationnement intelligent utilisant le microcontrôleur ESP8266. Le système peut :

- Surveiller le nombre de véhicules entrant et sortant du parking.
- Afficher le nombre actuel de voitures et le statut du parking sur un écran LCD.
- Synchroniser l'heure en utilisant NTP et fournir des données en temps réel via une interface web.
- Publier des données sur un serveur MQTT.

Caractéristiques

- Contrôler les barrières à l'aide de servos en fonction des lectures des capteurs à ultrasons.
- Connectivité Wi-Fi pour servir une page HTML dynamique de surveillance.
- Gestion des erreurs avec retour d'information sur un écran LCD RGB.
- Synchronisation de l'horloge en temps réel via NTP.

Exigences matérielles

- ESP8266 (NodeMCU)
- Deux capteurs à ultrasons (HC-SR04 ou similaires)
- Deux moteurs servo

- Module LCD RGB
- Réseau Wi-Fi
- Alimentation

Exigences logicielles

- Arduino IDE (version 2.1.0 ou supérieure)
- Bibliothèques :
  - `ESP8266WiFi`
  - `PubSubClient`
  - `Servo`
  - `rgb_lcd`

Installation

1. Installez les bibliothèques requises dans l'IDE Arduino.
2. Clonez ou téléchargez les fichiers du projet.
3. Ouvrez le fichier `.ino` dans l'IDE Arduino.
4. Configurez vos identifiants WiFi et les détails du serveur MQTT dans le code :

```
const char* ssid = "VotreSSID";
const char* password = "VotreMotDePasse";
const char* mqttServer = "votre.serveur.mqtt";
const int mqttPort = 1883;
const char* mqttUser = "VotreNomUtilisateurMQTT";
const char* mqttPassword = "VotreMotDePasseMQTT";

```

Listing 1: Configuration WiFi et MQTT

5. Téléchargez le code sur la carte ESP8266.

Utilisation

Démarrage du système

- Allumez le module ESP8266.
- Connectez-vous au réseau Wi-Fi configuré.
- Accédez à l'interface web à l'adresse IP affichée sur le moniteur série ou l'écran LCD.
- Surveillez les entrées et sorties des véhicules en temps réel.

Interface Web

L'interface web fournit :

- Statistiques en direct : voitures actuelles, capacité maximale.
- Journaux d'événements avec horodatage et descriptions.
- Affichage de l'horloge en temps réel.

Gestion des erreurs

- Les déconnexions WiFi déclenchent des tentatives de reconnexion, avec un retour d'information affiché sur l'écran LCD.
- Les situations de surcapacité sont gérées en refusant l'entrée et en affichant une erreur.

Aperçu du code

Code principal Arduino

Le fichier principal `.ino` gère :

- Connexion WiFi
- Communication MQTT
- Configuration du serveur web
- Logique de stationnement

`Parking.cpp`

Gère :

- Contrôle des barrières et lectures des capteurs à ultrasons.
- Mises à jour LCD et gestion des erreurs.
- Journalisation des événements.

`Parking.h`

Définit la classe `Parking` et ses méthodes, ainsi que des fonctions utilitaires pour la gestion des événements et la synchronisation de l'heure.