CAPTEUR DE SECHERESSE CONNECTE

Dossier de Tests de Validation

Référence : CAPTEUR_SECHERESSE_CONNECTE / DTV_ CAPTEUR_SECHERESSE_CONNECTE -V1.0

Date : 11/12/2024

HISTORIQUE DES RÉVISIONS DU DTV

Version	Date	Commentaires
1.0	11/12/24	Version initiale

SOMMAIRE

1. INTRODUCTION

1.1 Objet

Le présent document constitue le dossier de test du sous-système capteur de sècheresse connecté

1.2 Documents de référence

Référence	Titre
CdCF-V1.0	Cahier des charges fonctionnelles du client
UC-Capteur-Secheresse -V1.0	Diagramme des cas d'utilisation du projet
REQ- Capteur-Secheresse -V1.0	Diagramme des exigences du projet

2. DESCRIPTION DE L'ENVIRONNEMENT DE TESTS

2.1 Configuration matérielle et logicielle

2.1.1 Généralités

Pour répondre à la demande du client concernant la surveillance de l'arrosage de ses plantes avec un système connecté basé sur Arduino, un environnement de tests complet a été configuré. Le projet repose sur un système avec le capteur Go Tronic GT110, qui mesure l'humidité du sol, intégré avec une carte Arduino UNO et une liaison série vers un serveur Node.js.

L'environnement matériel inclut une carte **Arduino Uno**, le capteur **Go Tronic GT110**, ainsi que trois LEDs (rouge, verte et bleue) destinées à indiquer l'état du sol localement. Le capteur **Go Tronic GT110** est utilisé pour la détection précise du taux d'humidité dans le sol. L'ensemble est connecté via une alimentation USB ou une alimentation externe, en fonction des besoins du projet. La liaison entre la carte Arduino et le serveur **Node.js** est établie par un câble USB afin de permettre la transmission des données en temps réel.

Pour la configuration logicielle, l'IDE Arduino est utilisé pour programmer la carte avec les bibliothèques nécessaires, tandis que le serveur **Node.js** est configuré pour recevoir les données collectées par l'Arduino via la liaison série. L'interface web (**IHM**) est conçue pour afficher ces données en temps réel et permettre à l'utilisateur de configurer des seuils personnalisés pour surveiller les différents états d'humidité : sol trop sec, état optimal ou sol trop humide.

Les tests visent à valider l'ensemble des fonctionnalités du projet, notamment :

- La collecte en temps réel des données avec le capteur **Go Tronic GT110**,
- L'envoi et le traitement des données via la liaison série avec **Node.js**,
- L'affichage dynamique sur l'interface web en temps réel,
- La possibilité pour l'utilisateur de configurer ses seuils d'humidité via l'**IHM**,
- Le déclenchement de la LED (rouge pour sol trop sec, verte pour état optimal et bleue pour sol trop humide) en fonction de l'état du sol.

Ainsi, l'environnement de test regroupe la partie matérielle (capteur **Go Tronic GT110**, carte Arduino, LEDs, alimentation) et logicielle (IDE Arduino, serveur **Node.js**, **IHM** web) pour valider l'intégration, la communication et les fonctionnalités attendues.

2.1.2 Configuration du Capteur De Secheresse Connecte et de l'application

Matériel : Microcontrôleur **Arduino Uno**, capteur d'humidité **Go Tronic GT110**, trois LEDs (rouge, verte, bleue) pour l'indication de l'état du sol, alimentation USB.

Logiciel : Programme développé en C++ avec l'IDE Arduino pour la lecture des données du capteur, transmission via liaison série à **Node.js**, et gestion de l'affichage via l'interface web.

Environnement : Tests réalisés avec la liaison série entre l'Arduino et **Node.js**, en simulant différentes valeurs d'humidité pour valider l'affichage en temps réel sur l'interface web et le comportement des LEDs.

3. FICHES DE TESTS

Le présent chapitre contient les fiches de tests suivantes :

3.1

1.1 COLLECTE_DES_DONNEES

Les tests suivants permettent de tester les fonctionnalités générales du capteur de sècheresse connecté

Réf. : FE1.1 **:** **Collecte des données en temps réel/ CAS_01**		
ENVIRONNEMENT DU TEST		
Matériel et logiciel intégrés : Arduino Uno, capteur Go Tronic GT110, LEDs et serveur Node.js.		
DESCRIPTION	**CRITERES D'ACCEPTATION**	**RESULTAT**
1 Initialisation du capteur	Le capteur doit envoyer des données valides	
2 Transmission des données en temps réel	Les données doivent être reçues sans délai notable	
3 Affichage des valeurs sur l'IHM	L'interface web doit afficher les données correctement	
4 LEDs changent d'état	Les LEDs reflètent correctement les seuils configurés	
5 Stabilité de la liaison série	La communication reste stable sur 10 minutes	
ETAT DU TEST		
Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐		
Observations :		

Réf. : FE1.2 **: Configurer les seuils d'humidité / CAS_02**		
ENVIRONNEMENT DU TEST		
Test en situation réelle avec seuils personnalisés.		
DESCRIPTION	**CRITERES D'ACCEPTATION**	**RESULTAT**
1 Définition des seuils via l'IHM	Les seuils doivent être acceptés et sauvegardés	
2 Correspondance des LEDs aux seuils	Les LEDs réagissent aux nouveaux seuils	
3 Test des seuils extrêmes	Aucun comportement anormal pour des seuils très hauts ou bas	
ETAT DU TEST		
Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐		
Observations :		

1.2 TRANSMISSION_NODE_JS

Les tests suivants permettent de tester les fonctions spécifiques à l'envoi et le traitement des données via la liaison série avec **Node.js**

Réf. : FE2.1 **:** **Transmission à Node.js / CAS_01**		
ENVIRONNEMENT DU TEST		
Arduino connecté en série avec un serveur Node.js.		
DESCRIPTION	**CRITERES D'ACCEPTATION**	**RESULTAT**
1 Initialisation de la communication	Le serveur reçoit un premier signal valide	
2 Débit de transmission stable	Les données sont transmises sans pertes	
ETAT DU TEST		
Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐		
Observations :		