Abris de recharge – Consommation électrique

Dossier de Tests de Validation

Référence : ConsoElectriqueAbris/DTV_ ConsoElectriqueAbris-V3.0

Date : 07/05/2024

HISTORIQUE DES RÉVISIONS DU DTV

Version	Date	Commentaires
1.0	09/02/24	Version initiale
2.0	23/02/24	Version révisée
3.0	07/05/24	Version finale

SOMMAIRE

1. INTRODUCTION

1.1 Objet

Le présent document constitue le dossier de test et de validation du système de consommation électrique de l'abris à bicycles rechargeables.

1.2 Documents de référence

Référence	Titre
Diagramme des cas d'utilisation simplifié	Diagramme des cas d'utilisation du projet
Diagramme des exigences	Diagramme des exigences du projet

2. DESCRIPTION DE L'ENVIRONNEMENT DE TESTS

2.1 Configuration matérielle et logicielle

2.1.1 Généralités

Ce cahier de recette sert à valider le fonctionnement du système de consommation électrique de l'abris à bicycles rechargeables avant sa livraison au client pour son utilisation ; Le bon fonctionnement du système de récupération des données de consommation, notamment leur affichage sur le site web. L'ensemble des fonctionnalités du système sont validées par ce document.

A l'issue de l'exécution du plan de tests prévu dans ce document, le document ainsi obtenu est transformé en RTV (Rapport de Tests de Validation), ce RTV permettra de connaître l'état de validation du système au moment où il a été réalisé. En cas d'identification de non-conformités, le RTV servira à la correction de celle-ci ; il faut donc, lorsqu'une non-conformité est constatée, détailler au maximum le problème constaté et dans quel cas celle-ci s'est présentée. La section observation de chaque cas de test sert à détailler les conditions d'apparition des non-conformités.

Comme indiqué précédemment le plan de tests prévu va permettre de valider l'ensemble du fonctionnement du système tel que celui-ci est prévu dans les documents de spécification à savoir :

- Le diagramme des cas d'utilisation validé avec le client, qui recense l'ensemble des fonctionnalités du système à valider ;
- Le diagramme des exigences validé avec le client, qui recense l'ensemble des contraintes que le système doit respecter.

Le plan de test décrit dans ce document vérifie que l'ensemble des demandes du client et des spécifications sont respectées. L'objectif étant de vérifier que le produit est conforme aux attentes du client.

2.1.2 Configuration du système pour l'exécution du plan de test

Le système de consommation électrique de l'abris à bicycles rechargeables utilisé pour l'exécution du plan de tests doit être dans sa configuration d'usine afin de s'assurer qu'aucune manipulation ne puisse altérer le résultat du plan de tests prévu.

2.1.3 Configuration du site web pour l'exécution du plan de test

Le site web de l'abri utilisé pour l'exécution du plan de tests doit être accessible et fonctionnel. Il doit être configuré de manière à ce que les données de consommation puissent être récupérées et affichées correctement.

3. FICHES DE TESTS

Le présent chapitre contient les fiches de tests suivantes :

3.1 Programmes C++

3.1.1 Récupérer les informations des box à vélo

Les tests suivants permettent de vérifier la bonne récupération des données des box à vélo en amont.

Réf. : FE1.1 **:** **Récupérer les informations de chacune des box à vélo / Récupérer les données des capteurs**		
ENVIRONNEMENT DU TEST		
Le système de consommation électrique de l'abris à bicycles rechargeables étant dans sa configuration d'usine. Nous devons dans un premier temps vérifier qu'il est possible de récupérer les informations des différents capteurs de l'abris. Les tests seront effectués à l'intérieur d'un programme C++.		
DESCRIPTION	**CRITERES D'ACCEPTATION**	**RESULTAT**
1 Récupération des informations.	Les informations sont récupérées et sont utilisables (sans pertes ou fausses informations).	
2 Erreur de récupération des informations	Un message d'erreur apparait pour indiquer une erreur lors de la récupération des informations et indiquer de quel capteur l'erreur vient.	
3 Reformatage des informations.	Les informations sont correctement reformatées et prêtent à être utilisées.	
ETAT DU TEST		
Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐		
Observations :		

Réf. : FE1.2 **:** **Récupérer les informations de chacune des box à vélo / Connexion au Node.js**

ENVIRONNEMENT DU TEST

Le système de consommation électrique de l'abris à bicycles rechargeables étant dans sa configuration d'usine. Nous devons de vérifier qu'il est possible de se connecter au serveur Node.js. Les tests seront effectués à l'intérieur d'un programme C++.

	DESCRIPTION	CRITERES D'ACCEPTATION	RESULTAT
1	Connexion au serveur Node.js.	La connexion au serveur Node.js est établie et est opérationnelle.	
2	Erreur de connexion au serveur Node.js	Un message d'erreur apparait si la connexion au serveur Node.js n'a pas pu être établie.	
3	Reconnexion au serveur Node.js.	- Le programme C++ tente de se reconnecter à intervalles réguliers. - La connexion au serveur Node.js est rétablie et est opérationnelle.	
4	Déconnexion du serveur Node.js.	La connexion au serveur Node.js est rompue.	

ETAT DU TEST

Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐

Observations :

Réf. : FE1.3 **: Récupérer les informations de chacune des box à vélo / Envoi des données au Node.js**

ENVIRONNEMENT DU TEST

Le système de consommation électrique de l'abris à bicycles rechargeables étant dans sa configuration d'usine. Nous devons de vérifier qu'il est possible d'envoyer les données au serveur Node.js. Les tests seront effectués à l'intérieur d'un programme C++.

DESCRIPTION	CRITERES D'ACCEPTATION	RESULTAT
1 Afficher les données à envoyer.	Les données à envoyer sont affichées.	
2 Gérer la file d'attente.	Si la connexion au serveur Node.js n'est pas établie : - Si la file d'attente est pleine, le couple d'information le plus ancien est retiré et un message apparait. - Sinon, le couple d'information qui n'a pas pu être envoyé est ajouté à la file d'attente et un message apparait. Si la connexion au serveur Node.js est établie : - Le couple d'informations est ajouté à la file d'attente et un message apparait.	
3 Envoi des données au serveur Node.js.	Toutes les données de la file d'attente sont correctement envoyées au serveur Node.js.	
4 Erreur d'envoi des données au serveur Node.js.	Un message d'erreur apparait pour dire que les données n'ont pas pu être envoyées.	

ETAT DU TEST

Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐

Observations :

3.1.2 Récupérer les informations du capteur de luminosité

Les tests suivants permettent de vérifier la bonne récupération des données du capteur de luminosité en amont.

Réf. : FE2.1 **:** **Récupérer les informations du capteur de luminosité / Récupérer les données du capteur**		
ENVIRONNEMENT DU TEST		
Le système de consommation électrique de l'abris à bicycles rechargeables étant dans sa configuration d'usine. Nous devons dans un premier temps vérifier qu'il est possible de récupérer les informations du capteur de luminosité. Les tests seront effectués à l'intérieur d'un programme C++.		
DESCRIPTION	**CRITERES D'ACCEPTATION**	**RESULTAT**
1 Récupération des informations.	Les informations sont récupérées et sont utilisables (sans pertes ou fausses informations).	
2 Erreur de récupération des informations	Un message d'erreur apparait pour indiquer une erreur lors de la récupération des informations et indiquer de quel capteur l'erreur vient.	
3 Reformatage des informations.	Les informations sont correctement reformatées et prêtent à être utilisées.	
ETAT DU TEST		
Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐		
Observations :		

Réf. : FE2.2 **:** **Récupérer les informations du capteur de luminosité / Connexion au Node.js**

ENVIRONNEMENT DU TEST

Le système de consommation électrique de l'abris à bicycles rechargeables étant dans sa configuration d'usine. Nous devons de vérifier qu'il est possible de se connecter au serveur Node.js. Les tests seront effectués à l'intérieur d'un programme C++.

	DESCRIPTION	CRITERES D'ACCEPTATION	RESULTAT
1	Connexion au serveur Node.js.	La connexion au serveur Node.js est établie et est opérationnelle.	
2	Erreur de connexion au serveur Node.js	Un message d'erreur apparait si la connexion au serveur Node.js n'a pas pu être établie.	
3	Reconnexion au serveur Node.js.	- Le programme C++ tente de se reconnecter à intervalles réguliers. - La connexion au serveur Node.js est rétablie et est opérationnelle.	
4	Déconnexion du serveur Node.js.	La connexion au serveur Node.js est rompue.	

ETAT DU TEST

Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐

Observations :

Réf. : FE2.3 **: Récupérer les informations du capteur de luminosité / Envoi des données au Node.js**

ENVIRONNEMENT DU TEST

Le système de consommation électrique de l'abris à bicycles rechargeables étant dans sa configuration d'usine. Nous devons de vérifier qu'il est possible d'envoyer les données au serveur Node.js. Les tests seront effectués à l'intérieur d'un programme C++.

DESCRIPTION	CRITERES D'ACCEPTATION	RESULTAT
1 Afficher les données à envoyer.	Les données à envoyer sont affichées.	
2 Gérer la file d'attente.	Si la connexion au serveur Node.js n'est pas établie : - Si la file d'attente est pleine, le couple d'information le plus ancien est retiré et un message apparait. - Sinon, le couple d'information qui n'a pas pu être envoyé est ajouté à la file d'attente et un message apparait. Si la connexion au serveur Node.js est établie : - Le couple d'informations est ajouté à la file d'attente et un message apparait.	
3 Envoi des données au serveur Node.js.	Toutes les données de la file d'attente sont correctement envoyées au serveur Node.js.	
4 Erreur d'envoi des données au serveur Node.js.	Un message d'erreur apparait pour dire que les données n'ont pas pu être envoyées.	

ETAT DU TEST

Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐

Observations :

3.2 Serveur Node.js

3.2.1 Connexion aux programmes C++

Les tests suivants permettent de vérifier la connexion entre le serveur Node.js et les deux programmes C++.

Réf. : FE3.1 **:** **Serveur Node.js / Connexion aux programmes C++**

ENVIRONNEMENT DU TEST

Le serveur Node.js sera testé sur une machine virtuelle (VM) configurée à l'avance dans un mode de fonctionnement normal. Les tests seront effectués à l'intérieur d'un programme Javascript.

	DESCRIPTION	CRITERES D'ACCEPTATION	RESULTAT
1	Connexion aux programmes C++.	Le serveur Node.js est connecté aux deux programmes C++.	
2	Erreur lors de la connexion aux programmes C++.	Un message d'erreur apparait indiquant une erreur de connexion aux programmes C++ et précisant de quel programme l'erreur vient.	
3	Reconnexion aux programmes C++.	- Le serveur Node.js tente de se reconnecter à intervalles réguliers. - La connexion aux programmes C++ est rétablie et est opérationnelle.	
4	Déconnexion des programmes C++.	La connexion aux programmes C++ est rompue.	

ETAT DU TEST

Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐

Observations :

3.2.2 Récupérer données des programmes C++

Les tests suivants permettent de vérifier la bonne récupération des données des deux programmes C++ en amont.

Réf. : FE3.2 **:** **Serveur Node.js / Récupérer les données des programmes C++**

ENVIRONNEMENT DU TEST

Le serveur Node.js sera testé sur une machine virtuelle (VM) configurée à l'avance dans un mode de fonctionnement normal. Les tests seront effectués à l'intérieur d'un programme Javascript.

	DESCRIPTION	CRITERES D'ACCEPTATION	RESULTAT
1	Récupération des informations.	Les informations sont récupérées et sont utilisables (sans pertes ou fausses informations).	
2	Erreur de récupération des informations	Un message d'erreur apparait pour indiquer une erreur lors de la récupération des informations et indiquer de quel programme C++ l'erreur vient.	
3	Reformatage des informations.	Les informations sont correctement reformatées et prêtent à être utilisées.	

ETAT DU TEST

Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐

Observations :

3.2.3 Calculer les proportions et le moyennes de temps vert

Les tests suivants permettent de vérifier les calculs des proportions et des moyennes de proportions de temps vert.

Réf. : FE3.3 **:** **Serveur Node.js / Calculer les proportions et les moyennes de temps vert**		
ENVIRONNEMENT DU TEST		
Le serveur Node.js sera testé sur une machine virtuelle (VM) configurée à l'avance dans un mode de fonctionnement normal. Les tests seront effectués à l'intérieur d'un programme Javascript.		
DESCRIPTION	**CRITERES D'ACCEPTATION**	**RESULTAT**
1 Incrémenter les variables de temps de 5000 si les variables sont égales à 1.	Les variables de temps sont correctement incrémentées.	
2 Arrondir les proportions à deux chiffres après la virgule.	Les proportions sont arrondies à deux chiffres après la virgule.	
3 Stocker les proportions dans un tableau et faire leur moyenne.	Le tableau est bien créé et les moyennes stockées à l'intérieur.	
ETAT DU TEST		
Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐		
Observations :		

3.2.4 Envoi des données aux API CRUD

Les tests suivants permettent de vérifier l'acheminement des données du serveur Node.js jusqu'aux API CRUD des groupes Abris - Gestion Accès / Consommation électrique.

Réf. : FE3.4 **:** **Envoyer les données à l'API CRUD Consommation électrique**

ENVIRONNEMENT DU TEST

Le serveur Node.js sera testé sur une machine virtuelle (VM) configurée à l'avance dans un mode de fonctionnement normal. Les tests seront effectués à l'intérieur d'un programme Javascript.

	DESCRIPTION	CRITERES D'ACCEPTATION	RESULTAT
1	Vérification des données.	Toutes les données sont définies, aucune donnée ne manque. Un message s'affiche si une valeur a changée.	
2	Encapsuler les données au format JSON.	Les données sont correctement encapsulées au format JSON.	
3	Réception de la réponse.	La réponse de l'API est positive et les données sont bien envoyées.	

ETAT DU TEST

Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐

Observations :

Réf. : FE3.5 **:** **Envoyer les données à l'API CRUD Gestion Accès**

ENVIRONNEMENT DU TEST

Le serveur Node.js sera testé sur une machine virtuelle (VM) configurée à l'avance dans un mode de fonctionnement normal. Les tests seront effectués à l'intérieur d'un programme Javascript.

	DESCRIPTION	CRITERES D'ACCEPTATION	RESULTAT
1	Vérification des données.	Toutes les données sont définies, aucune donnée ne manque. Un message s'affiche si une valeur a changée.	
2	Encapsuler les données au format JSON.	Les données sont correctement encapsulées au format JSON.	
3	Réception de la réponse.	La réponse de l'API est positive et les données sont bien envoyées.	

ETAT DU TEST

Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐

Observations :

3.3 API

Réf. : FE4.**1 : API**

ENVIRONNEMENT DU TEST		
Vérifier que l'API fonctionne correctement		
DESCRIPTION	**CRITERES D'ACCEPTATION**	**RESULTAT**
1 Afficher les données	aller a l'adresse http://192.168.65.12:8080/select si les informations de la base de données sont affichées alors c'est bon .	
2 Ajouter des données	Depuis le site de test unitaire écrire les informations demandées puis cliquer sur "Ajouter"	
3 Modifier les données	Depuis le site de test unitaire cliquer sur "modifier" puis rentrer la nouvelle information	
4 Supprimer les données	Depuis le site de test unitaire cliquer sur Supprimer	
ETAT DU TEST		
Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐		
Observations :		

3.4 Composant React

Réf. : FE5.1 **:** **Composant React de Visualisation des Courbes de Consommation**

ENVIRONNEMENT DU TEST

Dans cette procédure de test, nous vérifions le bon fonctionnement du composant React pour visualiser les courbes de consommation dans le projet Abri.

DESCRIPTION	CRITERES D'ACCEPTATION	RESULTAT
1 Le composant React développé est intégré au projet Abri.	Le composant est correctement ajouté au code source du projet Abri, et il est accessible dans l'IHM web.	
2 Le composant appelle les API pour récupérer les valeurs nécessaires à l'affichage des courbes de consommation sous forme de graphique.	Les valeurs récupérées sont correctement traitées et affichées dans le graphique.	
3 Le composant appelle les API pour récupérer les valeurs des panneaux en temps réel, sans rafraîchissement de l'application web.	Les valeurs des panneaux sont actualisées en temps réel sur l'IHM web sans nécessiter de rafraîchissement de la page.	
4 Le composant simule les données si les API ne sont pas disponibles.	Lorsque les API ne sont pas accessibles, le composant génère des données de manière simulée pour maintenir la fonctionnalité de l'IHM web.	

ETAT DU TEST

Etat du test : Accepté ☐ Refusé ☐ Accepté sous Réserve ☐

Observations :