

Dossier technique – Partie Personnelle

**Serre Automatique**

Système d’acquisition

Ce document regroupe les informations concernant les capteurs de températures et l’interface web « Temps Réel ».

Valentin Chevallier

21/03/2019



Table des matières

**1.** [Présentation du projet 3](#_Toc4140203)

1.1 [Diagramme d’exigences 3](#_Toc4140204)

1.1.1 Synoptique de la réalisation……………………………………………………………………………..4

1.2 Rappel de la tâche de l'étudiant ……………………………………………………………………………..[5](#_Toc4140206)

[1.2.1 Ensemble des tâches réalisées par l’etudiant 5](#_Toc4140207)

1.3 [Réalisation de la base de données 6](#_Toc4140208)

[2. Mise en Place des Capteurs de Température 7](#_Toc4140210)

2.1 Recherches et Documentation [7](#_Toc4140211)

2.1.1 Schéma de [Cablage 9](#_Toc4140212)

2.1.2 [Solution de développement 10](#_Toc4140213)

2.2 [Etalonnage du Capteur de Témpérature 11](#_Toc4140214)

2.2.1 Graphique de conversion [11](#_Toc4140215)

[3. Réalisation de l’interface web 12](#_Toc4140217)

[3.1 Analyse du besoin 13](#_Toc4140218)

3.1.1 Solution choisie [14](#_Toc4140219)

3.1.2 [Premier apercu 14](#_Toc4140220)

3.2 Dynamisation de l'interface [15](#_Toc4140221)

3.2.1 [Connexion et interactions avec la base de données 16](#_Toc4140222)

3.2.2 [Affichage Responsive 17](#_Toc4140223)

3.2.3 Aperçu final  [19](#_Toc4140224)

4. [Tests Unitaires 19](#_Toc4140225)

4.1 [Fonction de conversion(courant en température) 20](#_Toc4140226)

4.2 [Problèmes rencontrés 21](#_Toc4140227)

4.3 Fiches Recettes……………………………………………………………………………………………….22

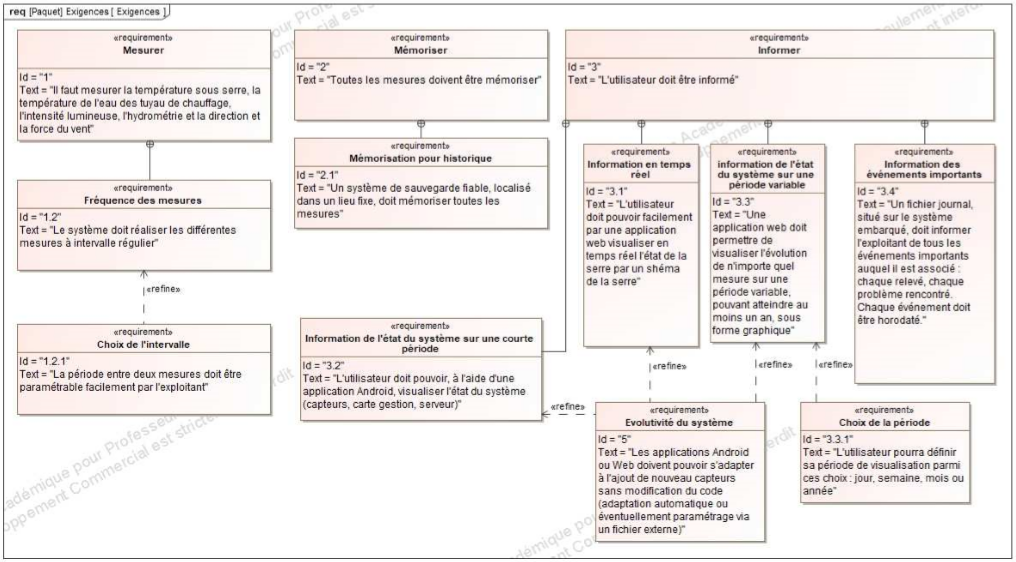
5. Bilan Personnel………………………………………………………………………....................................23

## 1.Présentation du projet

### Diagramme d’exigences

Le diagramme d’exigences ci-dessous représente les exigences fixées par le cahier des charges au tout départ du projet :

Parties concernant mes tâches dans le projet



Dans ce projet composé de 4 étudiants, 4 grand axes de développement se dégagent de ce diagramme :

-Les Mesures des différents capteurs et leur mémorisation

-La Supervision du système via une application Android

-L’interface Web qui doit à la foi posséder une partie temps réel (pour l’affichage en direct des mesures) et une partie périodisation

-La Gestion des capteurs et de leur mémorisation ainsi que la journalisation des évènements du système via une carte Raspberry

Le schéma suivant schématise ces différentes parties :

### Synoptique de la réalisation

Parties concernant mes tâches dans le projet

### 

**Killian Labattut**, l’étudiant n°1 est chargé de configurer et d’écrire le programme de gestion et de journalisation de la carte Raspberry, et également réaliser l’acquisition des mesures du capteur de vent, l’anémomètre.

**Lucas Minaud**, l’étudiant n°2 est chargé de réaliser la partie périodisation de l’interface web, donc l’affichage des mesures selon une période définie par l’utilisateur. Ainsi que l’acquisition de la luminosité et de l’humidité.

**Audran Raynal**, l’étudiant n°4 est chargé de réaliser l’application Android qui servira à la supervision du système pour permettre à l’utilisateur de savoir l’état de ce dernier, et l’acquisition de la pluviométrie.

## 1.2 Rappel de la tâche de l'étudiant

Pour ma part ma tâche au sein du projet, je devais réaliser la partie temps réel de l’interface.

Donc créer une interface épurée, simple de compréhension et d’utilisation qui permet à l’utilisateur de consulter les mesures acquises par les capteurs en temps réel, le tout en créant un site « responsive » et dynamique pour pouvoir récupérer les mesures contenues dans la base de données.

Je devais également acquérir la température ambiante dans la serre ainsi que la température des tuyaux d’eau de chauffage, avec des capteurs fournis au départ du projet.

Notre projet SFL5 étant en parallèle avec le projet SFL6 la base de données des deux projets est commune, j’ai donc été amené à collaborer avec un étudiant du groupe SFL6 pour réaliser cette dernière.

### 1.2.1 Ensemble des tâches réalisées par l’étudiant

A l’approche de la fin du temps imparti voici les taches réalisées :

L’Acquisition des mesures de température, le câblage et le programme d’acquisition sont fonctionnels et cohérent, l’installation sur la serre est également effectuée.

La base de données est correctement configurée et fonctionnelle.

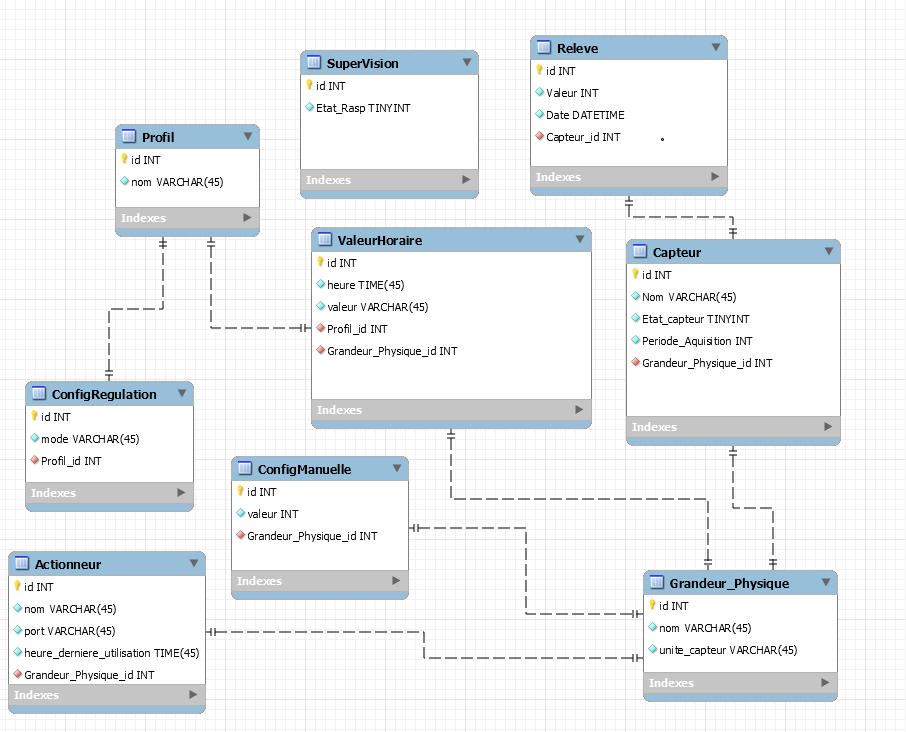
La partie temps réel de l’interface web est fonctionnelle, elle communique avec la base de données et récupère son contenu, elle reste cependant améliorable car elle ne possède par la fonctionnalité d’adaptation automatique.

L’étalonnage du capteur de température est effectué et cohérent.

## 1.3 Réalisation de la base de données

Comme cité précédemment je devais réaliser la base de données en association avec le projet SFL6 plus précisément l’étudiant **Goulven Perrin.**

J’ai donc réalisé sur « MySQL Workbench » un modèle relationnel pour schématiser la base de données :



Les seules tables utilisées par notre projet sont les tables Relevé, Capteur, et Grandeur Physique.

La table que j’utilise en majorité est la table relevé car elle contient les valeurs mesurées selon le capteur id renseigné.

## 2. Mise en Place des Capteurs de Température

### 2.1 Recherches et Documentation

Les deux capteurs de Température ont fait l’objet d’un choix limité, en effet les deux sondes était imposées par le cahier des charges.

Logiquement la plus grosse partie du travail fut de trouver la documentation concernant ces sondes :

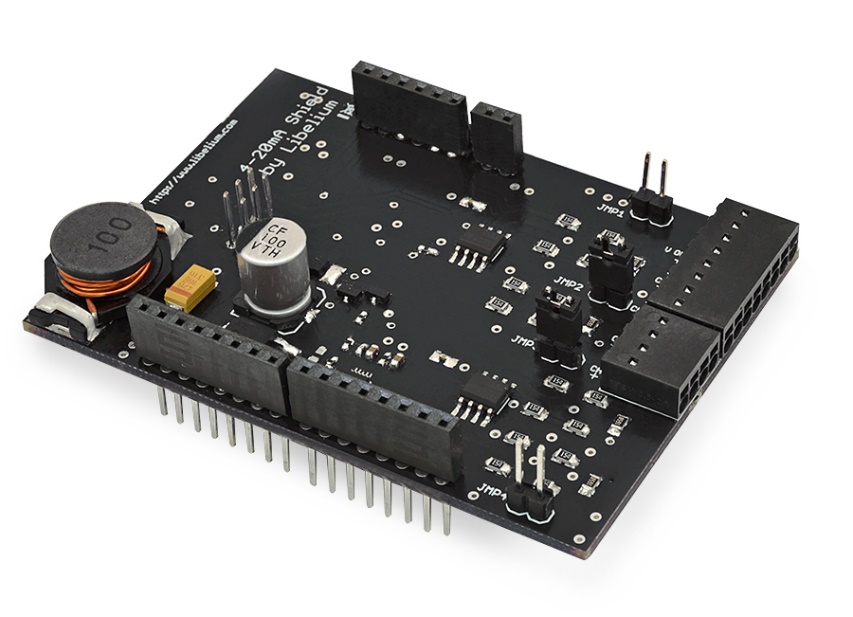


Après plusieurs recherches j’ai trouvé les informations ci-dessus, sachant que les deux sondes sont différentes sur un point : la plage de mesure, le premier va de 0°C à 45 °C (comme sur le document) et l’autre va de 0°C à 100°C.

Grace a la documentation on sait donc que les sondes sont des thermo-résistances, donc c’est la résistance qui évolue en fonction de la température. Elles sont donc analogiques, et la conversion numérique se fait avec une boucle 4-20Ma directement intégrée dans le capteur.

Il a donc fallu trouver un moyen de récupérer l’information numérique sur une carte de traitement : ici une carte Arduino.

La solution trouvée a donc été un Shield Arduino qui embarque une boucle 4-20Ma qui permet de récupérer la valeur de courant et de tension en sortie du capteur :

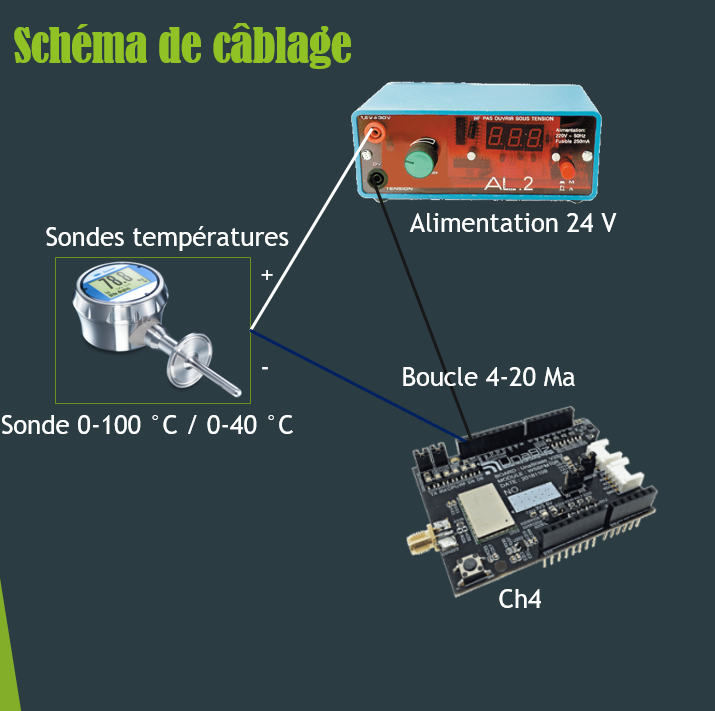


C’est un Shield de la marque « Libellium » qui était déjà présent dans l’établissement, de nouveau il a fallu trouver de la documentation pour trouver comment câbler le capteur.

En sachant que ce dernier a besoin d’une alimentation de 20 à 30 volts pour le fonctionnement nominal.

### 2.1.1 Schéma de câblage

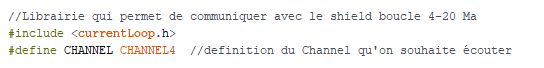
Avec l’aide de la documentation j’ai réalisé un schéma indiquant comment câbler les capteurs avec le Shield et une alimentation externe de 24 V :



Il faut donc créer une boucle d’alimentation avec la masse du capteur relié a la borne plus de l’entrée du Shield et mettre en commun la masse de l’alimentation avec celle du Shield.

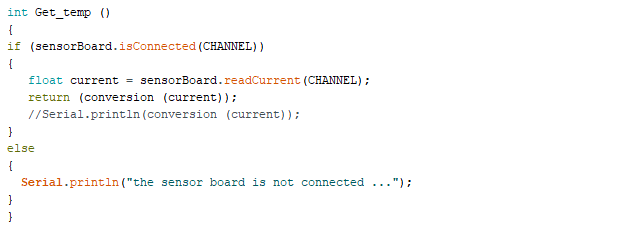
### 2.1.2 Solution de développement

Sur le schéma ci-dessus le « Ch4 » signifie Channel 4 car les entrées du Shield sont en fait des « Channel » numéroté de 1 à 4, cette information est utile pour le programme qui permettra d’interpréter les résultats :



La communication avec le Shield se fait grâce a la librairie montrée ci-dessus nommé « currentLoop.h ».

Cette librairie contient une fonction qui répond à l’objectif souhaité :



Cette fonction nommé « sensorBoard.readCurrent() » retourne la valeur du courant en Ma du Channel renseigné en paramètre.

J’ai également rajouté une fonction pour vérifier si le capteur est bien connecté, cette fonction se nomme « sensorBoard.isConnected() » cette fonction renvoie « True » si un appareil est branché sur le Channel renseigné.

Ma fonction « Get\_temp () » renvoie donc directement la valeur convertie de tension en température grâce à la fonction « conversion » que je vais expliquer ci-dessous.

### 2.2 Etalonnage des Capteurs de Température

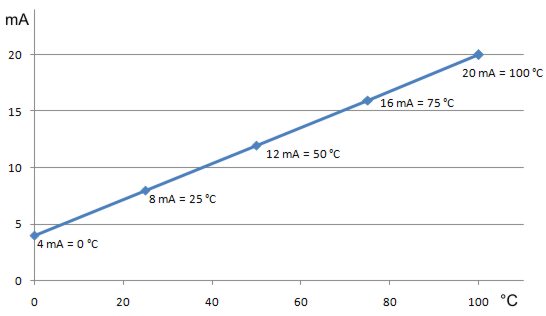
Les sondes fournies sont des sondes pt-100 cela veut dire qu’elles appartiennent à une certaine classe de capteur et souvent les sondes de ce genre utilisent des boucles 4-20 Ma.

Et dans ce cas le 4-20 Ma devient la plage de mesure du capteur, donc quand on a 4 Ma en sortie on est au plus bas mesurable et inversement à 20 Ma c’est la plus haute valeur.

### 2.2.1 Graphique de conversion

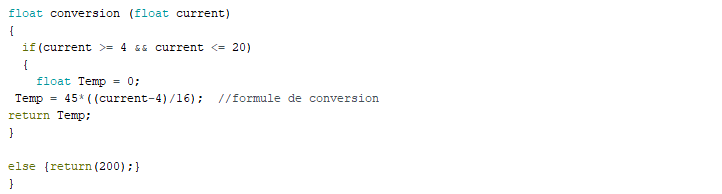
Pour créer une fonction de conversion il a donc fallu que je trouve une formule de conversion.

J’ai donc effectué une série de mesures pour obtenir une courbe de conversion pour trouver l’équation de cette courbe :



Les mesures ici étaient faite avec la sonde 0-100°C et on constate que la courbe est linéaire ce qui donne la formule suivante : **T = Tmax \* (Courant - 4 /16)**

Cette formule ayant été vérifié avec les deux sondes il ne restait plus qu’a la programmer dans une fonction :



La seule valeur qui peut être amenée à changer est la valeur Max (45 ou 100)

La présence d’une boucle « if () » est justifié pour éviter les erreurs, si on sort de la plage 4 – 20Ma la fonction renvoie une valeur excessive et le programme renverra donc une erreur s’il rencontre cette valeur.

Enfin il ne reste plus qu’a installer les capteurs dans la maquette de la serre et tester en conditions réelles et les résultats était cohérent la partie mesure de la température était terminée.

## 3. Réalisation de l’interface web

Au départ du projet la seule contrainte imposée pour l’interface web était le choix de l’outil de développement, les choix étais :

-**Standard**, une page web écrite a la main en HTML, PHP, CSS, Javascript. Une page dynamique stylisée avec les méthodes vues en cours.

-**Framework**, avec un nouvel outil, un framework nommé « Symfony » qui simplifie la réalisation de site web, mais qui demande une connaissance de l’outil et donc dans ce cas présent d’un apprentissage.



Symfony étant un framework il aurais permis beaucoup d’amélioration moderne, cependant le temps nécessaire a l’apprentissage était trop long, on a donc dû opter mon collègue et moi-même pour du développement web HTML, PHP.

### 3.1 Analyse du Besoin

Pour ma partie de l’interface en temps réel, j’ai d’abord réfléchi à comment présenter la page pour quelle soit agréable à utiliser, de plus je voulais que chaque mesure soit clairement visible et repérable directement (ex : icone indiquant de quel type est la mesure).

De plus dans une mesure de confort d’utilisation le site se doit d’être « responsive » c’est-à-dire qu’il s’adapte selon la taille de la fenêtre et enfin les mesures doivent être pouvoir être mise à jour, rafraichit par l’utilisateur.

### 3.1.1 Solution choisie

Pour l 'affichage j’ai d’abord pensé à une serre sur laquelle on pourrait cliquer, qui donnerait selon l’endroit ou on clique la mesure correspondante et je souhaitais rajouter un espace dans lequel l’utilisateur pourrait avoir des infos sur les capteurs.

Pour la partie affichage nous avons choisi d’utiliser un Template, un modèle préfait que nous modifierons selon nos besoins.



Et enfin pour la partie « responsive » comme pour beaucoup de site nous utiliserons Bootstrap.

### 3.1.2 Premier Aperçu



A cette version du site il n’est pas responsive et les valeurs des mesures ne sont pas affichées, voulant plus tard travailler dans le design j’ai gardé une palette de couleur pour rester dans le thème d’une serre

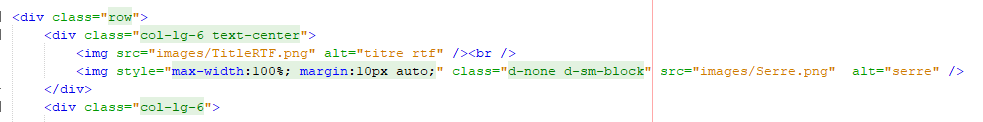
Les boutons « Info » placés serve a donné des informations sur les capteurs comme cité auparavant.

### 3.2 Dynamisation de l’interface

Pour dynamiser l’interface il faut commencer par indiquer que nous utiliserons Bootstrap dans la page grâce à cette ligne :



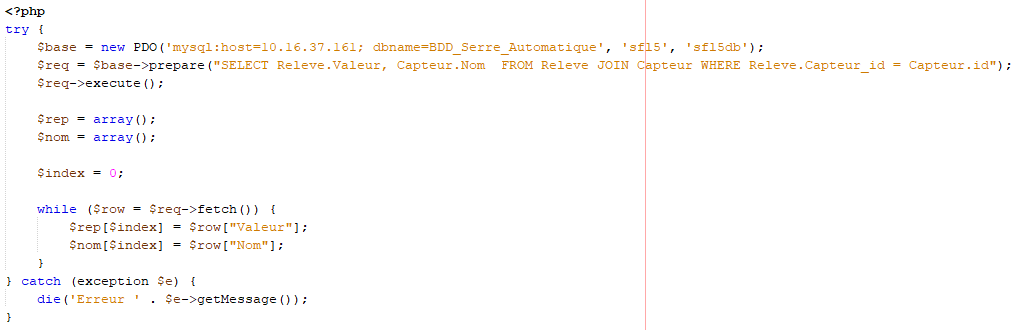
Une fois ce lien importé on peut utiliser des classe CSS qui permettent l’adaptation automatique des éléments de la page en fonction de la taille de cette dernière.



Dans cet exemple les classes « col-lg-6 » permettent cette adaptation.

### 3.2.1 Connexion et interactions avec la base de données

Le site devait grâce au PHP récupérer des valeurs contenues dans la base de données, pour ce faire j’utilise la commande suivante :



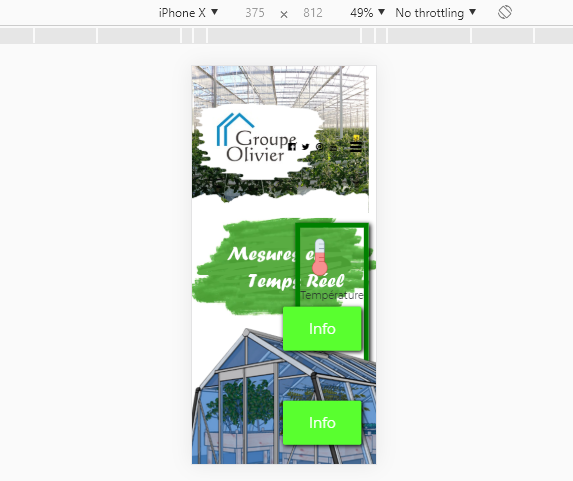
En créant un objet « pdo » qui contient les informations pour se connecter à la base de données donc, l’hôte, le nom de la base, et les informations de connections.

Ensuite en créant une variable « req » qui contient la requête SQL qui me permet d’obtenir uniquement les valeurs dont j’ai besoin, et enfin cette requête est exécuté avec la commande « execute () », vu qu’il y a plusieurs valeurs j’ai choisi de les stocker dans des array () qui sont des tableaux dynamiques.

Cette partie du code me permettant de récupérer les mesures pour les différents capteurs ainsi que leur nom pour savoir de quelle mesure il s’agit.

### 3.2.2 Affichage Responsive

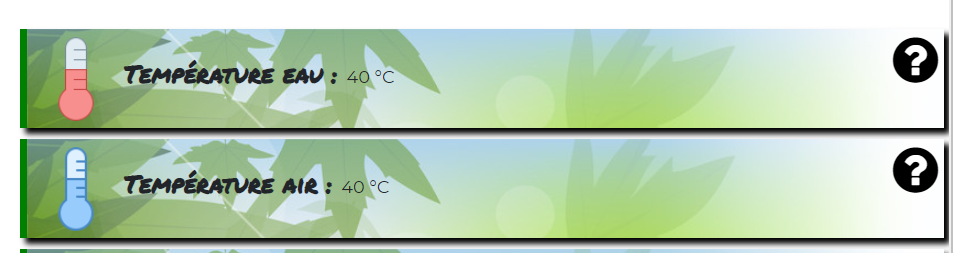
Pour permettre un affichage complètement « responsive » je me suis servi de l’outil de développement de google qui permet de le vérifier et voila ce que donne l’affichage responsive du site pour l’instant :



On constate très clairement que si un jour l’utilisateur souhaite consulter les mesures sur un support autre que l’ordinateur cela posera un problème.

Donc j’ai revu l’affichage en intégrant du jQuery pour la dynamisation et avec les ajouts en Bootstrap ont palier à ce problème.

Les cellules de tableaux s’ouvrent donc maintenant avec des effets d’animations jQuery :





En cliquant sur l’icône « ? » sa augmente la taille de la cellule et affiche des informations sur le capteur en question.

Et les cellules sont placée dynamiquement dans la page en blocs avec l’image de la serre qui sous une certaine largeur d’écran disparais pour ne laisser place qu’a l’essentiel de l’information.

### 3.2.3 Aperçu Final

Le site sous sa version finale de rendu de projet ressemble donc à cela :



Et le site maintenant s’adapte en fonction de la taille de la fenêtre.

Les valeurs des mesures sont identiques car il n’y a pas d’autre valeur encore renseigné dans la base de données que ces valeurs de tests.

Donc la partie Temps Réel du site Web est en plus grande partie terminée.

## 4.Test Unitaire

Le Test Unitaire demandé pour la revue 3 de projet est pour ma part portée sur la fonction de conversion du courant en température.

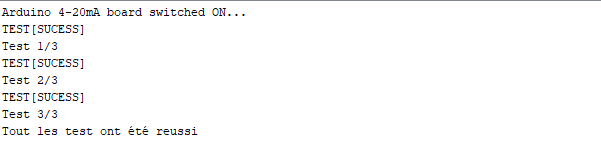
Ce test est effectué sous forme d’une fonction qui fait appel a cette fonction de conversion pour en tester le fonctionnement.

La fonction se présente comme cela :



En faisant plusieurs boucles « if » pour tester différentes valeurs de courant on renvoie « SUCESS » pour chaque test réussi et un compteur pour indiquer si tous les tests ont été eux aussi réussi.

Voici ce qu’on est censé attendre :



D’abord la confirmation que la carte est connectée et ensuite la confirmation de réussite des tests.

### 4.2 Problèmes Rencontrés

Tout au long du projet différents problèmes sont arrivés notamment pour ma partie au niveau du câblage des capteurs, les capteurs sont anciens et peu rependu. Il a été compliqué de trouver un semblant de documentation technique à leur sujet.

Leur alimentation était inconnue et le risque de griller une sonde était réel.

L’autre gros problème rencontré pour l’interface web fut dans le choix de l’outil de développement, nous avons essayé mon collègue et moi d’installer Symfony sans succès c’est la principale raison de l’abandon de ce choix.

### 4.3 Fiches Recettes

1-Acquérir

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nom** : Acquérir la valeur de température | | | | **Groupe :** SFL5 | |
| **Recette :** Technique | | | | | |
| **Objectif :** | | « Vérifier que le système récupère la bonne température » | | | |
| **Elément à tester** | | Carte Arduino, capteur ARIA pt-100, Shield Libellium. | | | |
| **Pré requis :** | | La carte et le capteur sont alimentée et le programme est lancé | | | |
|  | | | | | |
| **Scénario** | | | | | |
| Id 1 | Démarche | | Données | Comportement attendu | OK |
| 1 | Connecter le capteur au Shield Libellium sur le ch4 | |  | Le programme cesse d’envoyer « the sensor board is not connected ... » |  |
| 2 | Attendre quelques mesures envoyées par la carte | |  | Réception de plusieurs mesures quasi-similaires |  |
| 3 | Vérifier la température avec un thermomètre | |  | Les thermomètres indiquent les mêmes valeurs de température |  |

2-Serveur Web

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nom** : Accéder à l’interface Web TR | | | | **Groupe :** SFL5 | |
| **Recette :** Technique | | | | | |
| **Objectif :** | | Vérifier le bon fonctionnement de la page Temps Réel | | | |
| **Elément à tester** | | Serveur Web, base de données | | | |
| **Pré requis :** | | Le navigateur web est lancé | | | |
|  | | | | | |
| **Scénario** | | | | | |
| Id 2 | Démarche | | Données | Comportement attendu | OK |
| 1 | Taper dans la barre de navigation « 10.16.37.161 » | |  | La page d’accueil s’affiche |  |
| 2 | Cliquer sur le bouton « la serre en temp réel » | |  | La page Temps Réel S’ouvre et les valeurs sont affichées |  |
| 3 | Cliquer sur « actualiser les valeurs » | |  | La page se recharge partiellement et les valeurs sont Mises à jour. |  |

## 5. Bilan Personnel

Pour conclure sur cette fin de projet je dirais qu’il fut enrichissant aux vues de ce que j’y ai appris, aussi bien en termes techniques qu’en relation de travail a visée sans doute plus grande que celle du BTS. Ce fut un réel plaisir de travailler au côté de mes camarades.

Je rajouterais que l’objectif au départ avait un réel sens et une réelle application, j’ose espérer qu’a ce sujet notre travail sera utilisé a de bonne fin dans le milieu qui lui est destiné.

Je pense avoir répondu aux attentes placées en ma personne sur l’ensemble du temps passé en projets.

Je tiens également à remercier les professeurs qui nous ont accompagner sur toute la durée du projet et a qui nous devons beaucoup. Et également mes camarades de projets.