

SIN

PROJET STI2D

**SESSION
2018**



**LYCÉE RENÉ GOSCINNY
500 ROUTE DES CROVES - 06340 DRAP**

**BACCALAURÉAT
SCIENCES ET TECHNOLOGIES DE L'INDUSTRIE
ET DU DÉVELOPPEMENT DURABLE**

**SPÉCIALITÉ
SYSTÈMES D'INFORMATION
ET NUMÉRIQUE**

PROJET :

Stores et Lumières Électriques Gérés Par Informatique

**RÉFÉRENCE :
S-AG-U-05**

**CANDIDAT
José SRIFI**

**MEMBRES DE L'ÉQUIPE PROJET
Baptiste BELLONE
Alexandre MICHEL
Valentin MINIÈRE**

Sommaire

I – Partie Générale.....	6
<i>A – Présentation du projet.....</i>	<i>6</i>
<i>B- Cahier des Charges.....</i>	<i>7</i>
<i>C – Documents usuels</i>	<i>8</i>
1 – Diagramme de Gantt	8
2 – Diagrammes de Cas D’utilisation	9
3 - Diagramme de séquence	10
4 - Diagramme des exigences,	11
<i>D – Matériel utilisé</i>	<i>12</i>
1 – Carte Raspberry Pi 3	12
2 – Carte Arduino Ethernet.....	12
3 – Shield Arduino Carte Moteur.....	12
4 – Shield Arduino Carte Relais.....	12
5 – Shield Arduino Carte Prototype.....	13
6 – Bandeau LED RGB	13
7 – Luxmètre	13
8 – Anémomètre.....	13
9 – Store.....	13
<i>E – Logiciels utilisés.....</i>	<i>14</i>
1 – Arduino	14
2 – Visual Studio Code	14
3 – Magic Draw	14
4 – Raspbian	14
5 – VNC Viewer	14
6 – HTML5 / CSS3 / PHP5 / JQuery/ Base de Données	15
<i>F – Liens avec le développement durable.....</i>	<i>16</i>
<i>G – Répartition des taches.....</i>	<i>17</i>
1 – Baptiste	17
2 – Alexandre.....	17
3 – Valentin.....	17
4 – José (moi).....	17

II – Partie Personnelle.....	18
<i>A – Réception et mise en place du matériel.....</i>	<i>18</i>
1 – Réception du matériel	18
2 – Installation du matériel.....	18
3 – Déroulement de l’installation	19
<i>B – Création du site internet.....</i>	<i>20</i>
1 – L’en-tête du site	21
2 – Le contenu de la page d’accueil.....	22
3 – Page d’accueil et page « À propos »	23
4 – Les scripts PHP	24
I – L’affichage des valeurs de notre base de données.....	24
II – Envoi des informations depuis le site vers Arduino et la base de données	25
III – Récupération des données pour la page d’accueil	26
5 – Animation CSS pour le chargement.....	27
<i>D – Problèmes Rencontrés</i>	<i>28</i>
III – Annexes	29
<i>A – Arborescence des fichiers du site internet</i>	<i>29</i>
<i>B – Répertoires des programmes du projet.....</i>	<i>30</i>
<i>C – Participation aux Olympiades de Sciences de l’ingénieur.....</i>	<i>31</i>

I – Partie Générale

A – Présentation du projet

Nous sommes une équipe de quatre élèves de Terminale STI2D (Science et Technologies de l'Industrie et du Développement Durable) spécialité SIN (Science de l'Information et du Numérique). Nous avons choisi de réaliser un projet de domotique consistant à piloter un store et des lumières par informatique.

Le but de ce projet est de nous faire découvrir les aspects de l'informatique, de l'électronique et de la maîtrise des énergies. En effet, depuis le début de notre année de première, nous n'avons abordé que de manière succincte ces domaines lors des TP et nous voulions en découvrir beaucoup plus.

Nous avons le choix entre plusieurs projets, mais aucun d'entre eux ne nous intéressait. Nous avons donc soumis notre idée à notre professeur : « **Piloter un store et des lumières via un site internet** ». Ce dernier a accepté de présenter notre projet à l'académie. Après son accord, nous avons commencé à réfléchir sur comment le réaliser.

Nous avons consacré plusieurs heures à la recherche du matériel, des contraintes, des possibilités et aussi des documents techniques correspondant à nos attentes, tout en gardant à l'esprit que nous avons un budget limité.

Pour notre projet, nous avons choisi le nom de **SLEGPI** :



B- Cahier des Charges

La domotique permet de centraliser le contrôle des différents systèmes et sous-systèmes de la maison, du lycée ou de l'entreprise (chauffage, volets roulants, porte de garage, portail d'entrée, prises électriques ...).

Vous aurez à contrôler plusieurs dispositifs à savoir :

- Un store automatique,
- Les lumières intérieures et extérieures.

Pour atteindre cet objectif, vous développerez une liaison Serveur/Client internet. Vous mettrez en œuvre une carte Raspberry et des Arduino organisées de la façon suivante :

- Un module Serveur à l'aide de la carte Raspberry connectée sur le réseau pédagogique (la réalité physique de votre étude se déroulant au lycée).
- Ce module Serveur comprendra également un module radio (Emetteur/Récepteur) recevra les valeurs fournies par le dispositif d'acquisition des valeurs
- Plusieurs cartes Arduino chacune munie de son module radio (Emetteur/Récepteur) pour l'acquisition des données et l'actionnement des parties opératives.

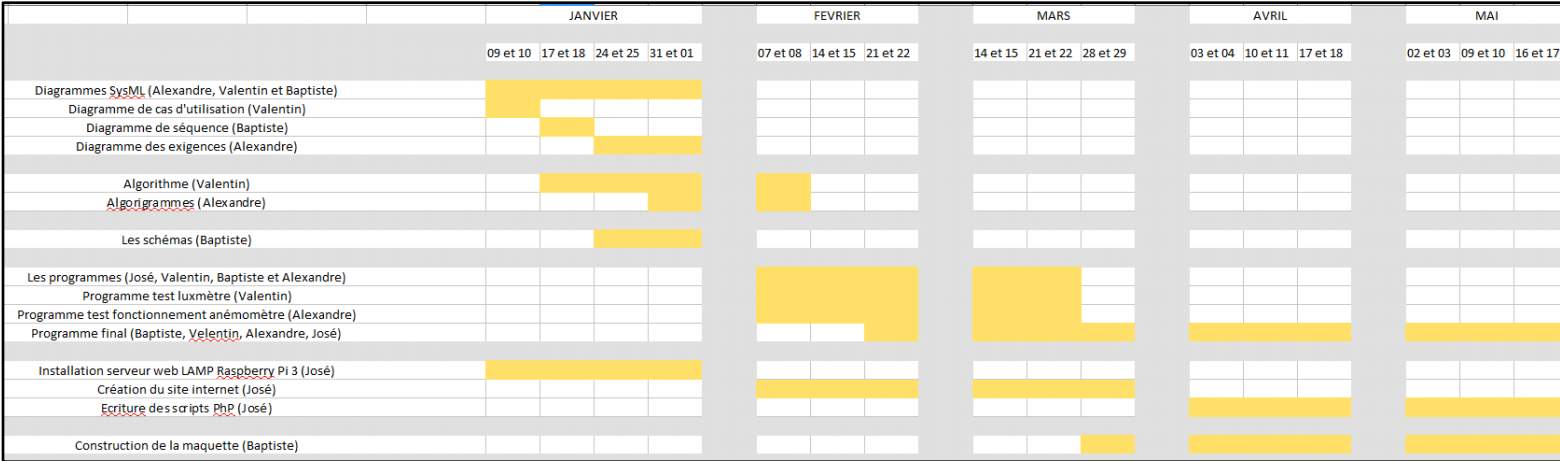
Vous aurez à réaliser le travail suivant :

- Concevoir et implanter un site internet dans le Serveur Raspberry (Serveur Web embarqué),
- Concevoir et implanter les programmes d'acquisition et de commande des parties opératives
- Réaliser les essais,
- Vérifier le bon fonctionnement de la liaison Serveur/Client,
- Rendre compte de toute cette étude dans votre dossier de présentation.

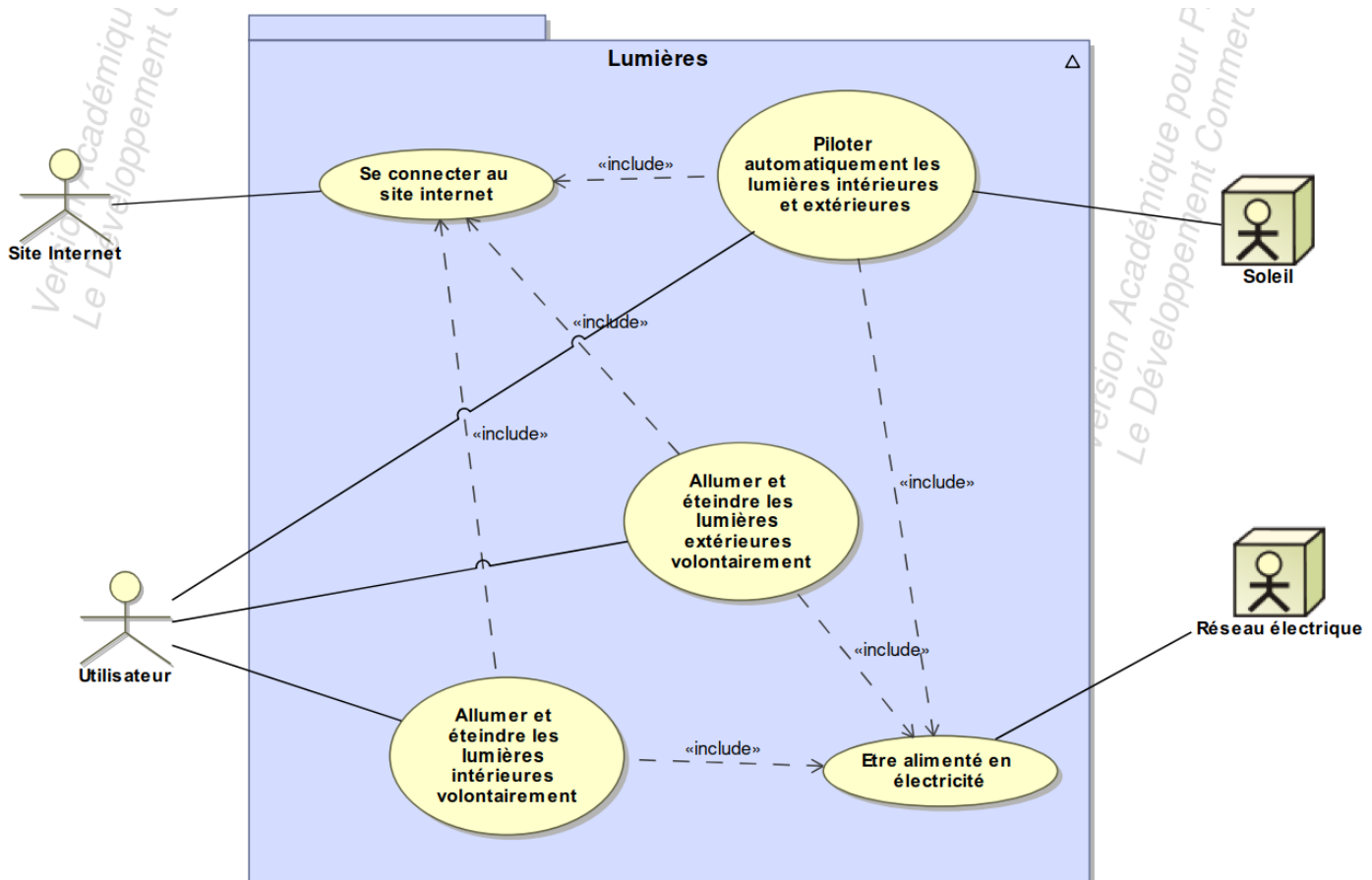
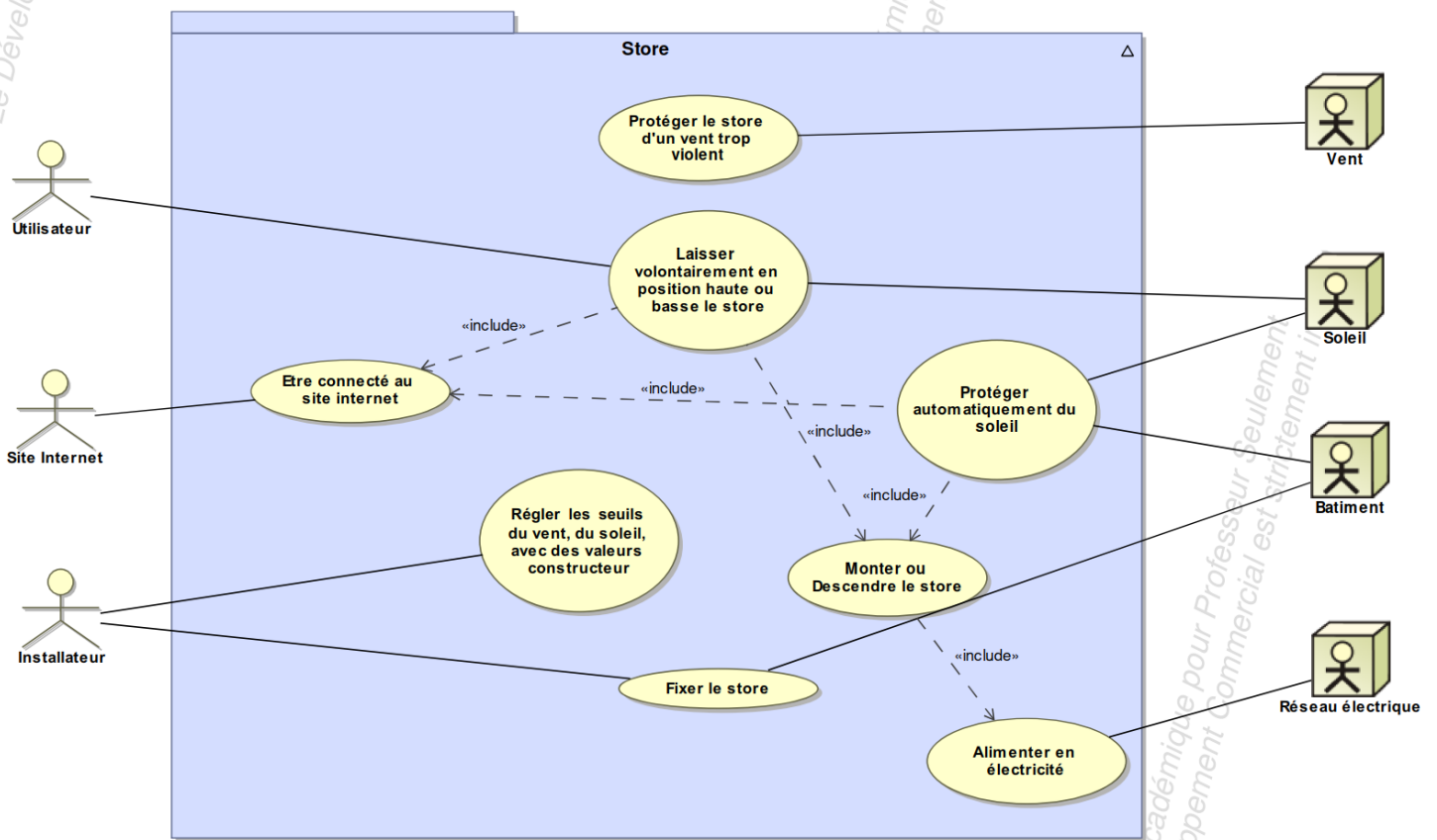
C – Documents usuels

Voici les documents usuels décrivant le fonctionnement du projet.

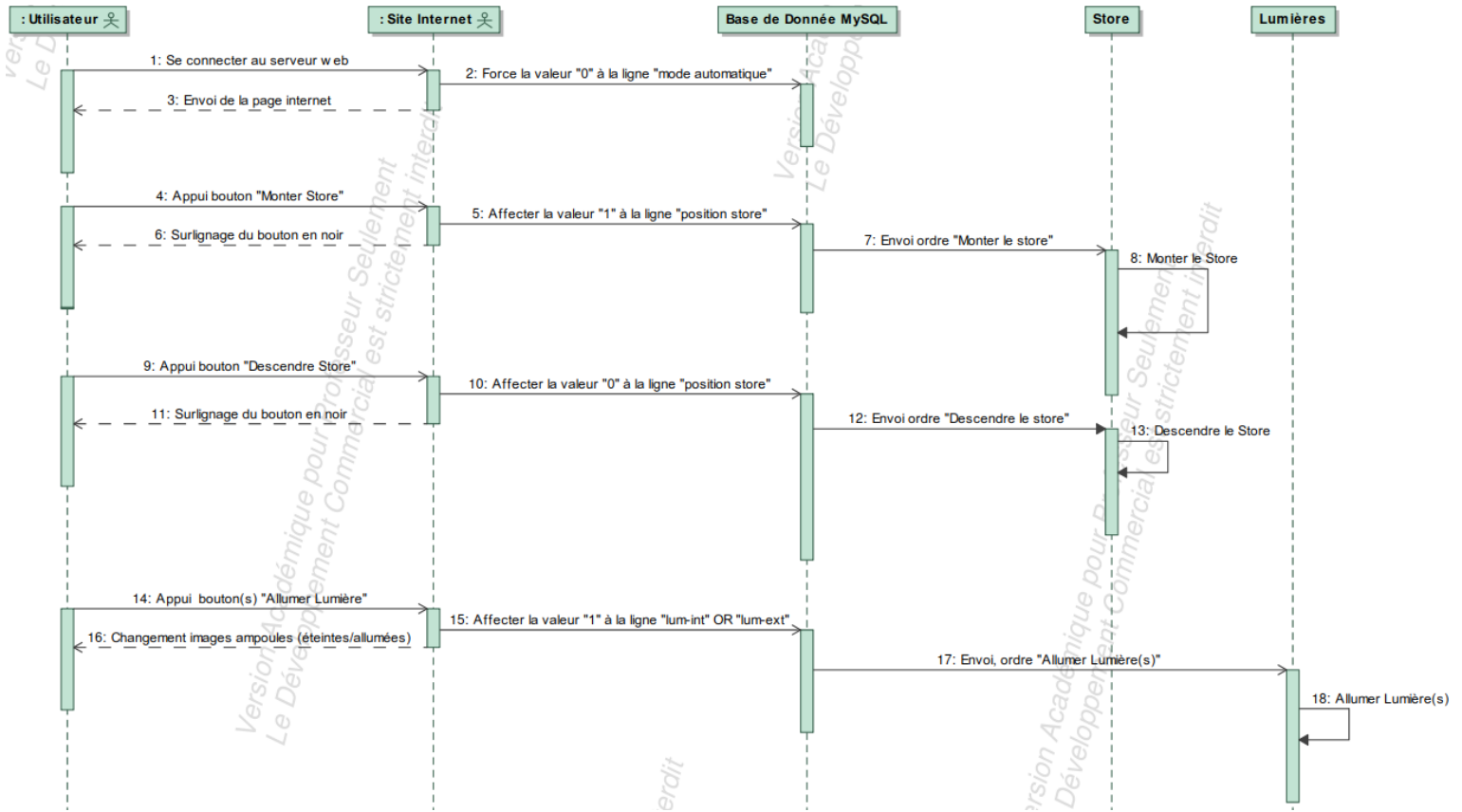
1 – Diagramme de Gantt



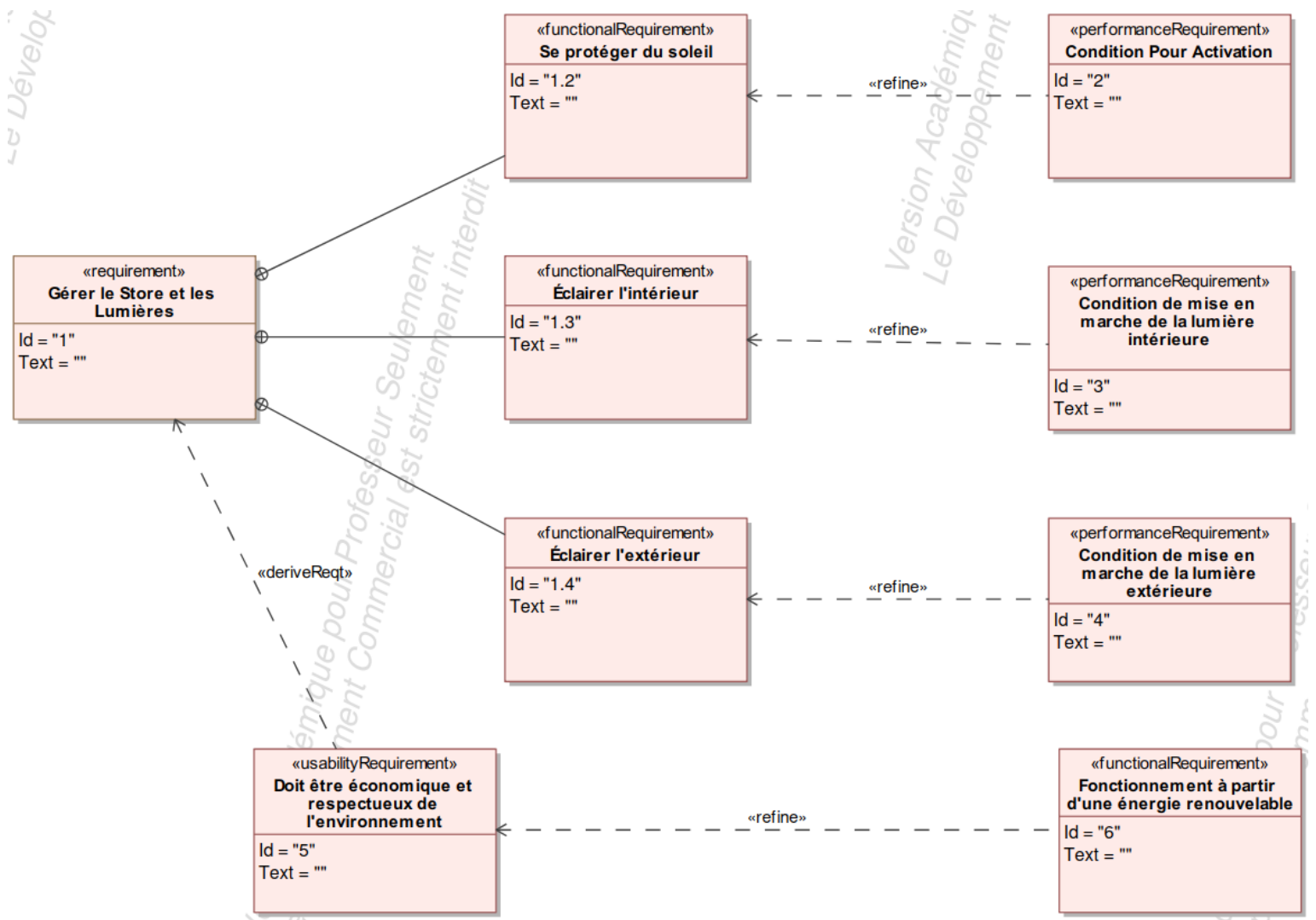
2 – Diagrammes de Cas D'utilisation



3 - Diagramme de séquence



4 - Diagramme des exigences,



D – Matériel utilisé

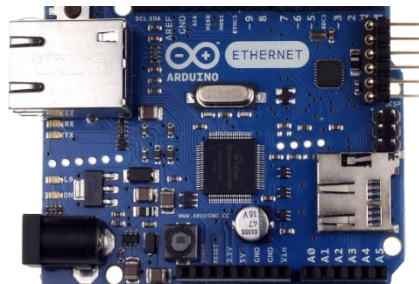
1 – Carte Raspberry Pi 3

Un Raspberry Pi est un nano-ordinateur (petit ordinateur) mono-carte. Cet ordinateur a la particularité d'avoir une taille proche de celle d'une carte de crédit. Cela nous a permis de la transporter facilement. Cette carte est principalement destinée pour l'apprentissage de la programmation informatique. Elle est compatible avec plusieurs OS (Système d'exploitation). Cela permet l'exécution de plusieurs distributions du système d'exploitation Open Source Linux (voir page 13).



Nous avons utilisé cette carte pour pouvoir héberger notre serveur web.

2 – Carte Arduino Ethernet

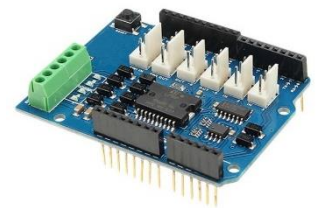


Un Arduino est une carte dotée d'un microcontrôleur qui peut être programmé de manière à effectuer des tâches très diverses comme la domotique, le pilotage d'un robot, de l'informatique embarquée, etc...

Nous avons choisi le modèle Ethernet pour pouvoir communiquer à l'aide d'un câble RJ45 à notre carte Raspberry Pi 3.

3 – Shield Arduino Carte Moteur

Un Shield carte moteur est un petit dispositif qui permet à une carte Arduino de gérer jusqu'à 4 moteurs continus et 2 moteurs pas-à-pas. Dans notre cas, il nous a permis de piloter le moteur de notre store.



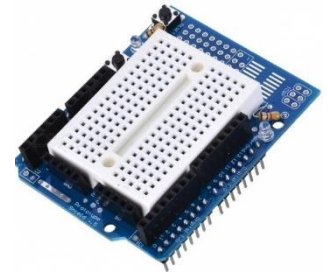
4 – Shield Arduino Carte Relais



Un relais est un organe électrique permettant la commutation de liaisons électriques. Il est chargé de transmettre un ordre depuis la partie opérative à la partie puissance d'un appareil électrique. Utile pour effectuer une commande de puissance de type tout ou rien, il nous a permis de transformer les 5 V de notre carte Arduino en 12 V pour nos Leds.

5 – Shield Arduino Carte Prototype

La carte Prototype est un moyen de tester nos capteurs avec Arduino sans souder de composants entre eux. Dans notre cas, il nous a permis de mettre en place les butées hautes et basses pour notre store.



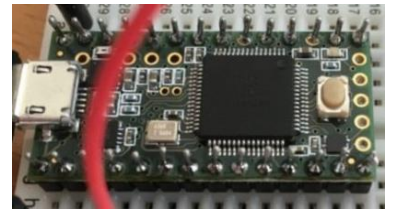
6 – Bandeau LED RGB

Nous avons utilisé des bandeaux LED RGB pour les lumières intérieures et extérieures car celles-ci fonctionnant en 12 V, elles sont compatibles avec notre carte relais.



7 – Luxmètre

Un luxmètre est un appareil permettant de mesurer la quantité de lumière dans l'environnement. Cette valeur s'exprime en lux ou en lumens. Dans notre cas, nous avons utilisé ce luxmètre pour mesurer la quantité de lumière avec notre carte Arduino afin de piloter automatiquement nos parties opératives.



8 – Anémomètre

Un anémomètre est un appareil permettant de mesurer la vitesse ou la pression du vent. Nous nous en sommes servis pour mesurer la quantité de vent avec notre carte Arduino afin de piloter automatiquement l'ouverture et la fermeture de notre store.



9 – Store

Nous avons choisi d'utiliser un store vénitien pour notre maquette car il s'agit du type de store qui pouvait le mieux s'assembler avec notre maquette.



E – Logiciels utilisés

1 – Arduino

Arduino est un logiciel open source permettant de piloter le microcontrôleur de notre carte Arduino Ethernet. Il fonctionne en langage C. Dans notre cas, nous nous en sommes servis pour envoyer les informations nécessaires au microcontrôleur de la carte Ethernet afin de piloter notre store et nos lumières. Notre programme permet de lire et d'écrire dans notre base de données.



2 – Visual Studio Code



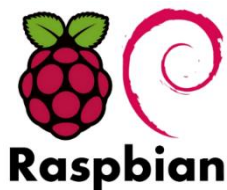
Visual Studio Code est un logiciel d'édition fichiers textuels (mais aussi de pages webs, de scripts Js). Nous nous en sommes servis pour créer les pages de notre site internet ainsi que les scripts PHP et Js. José a également installé une version sur notre carte Raspberry Pi 3.

3 – Magic Draw

Magic Draw est un logiciel de visualisation et de création de projets en SysML ou UML. Pour notre projet, nous nous en sommes servis pour créer nos « diagrammes de Cas d'utilisation » (page 8), « diagrammes de séquence » (page 9) et « diagramme des exigences » (page 10).



4 – Raspbian



Raspbian est une distribution open source basée sur Linux Debian et optimisée pour fonctionner sur un Raspberry Pi. Nous avons installé cette distribution sur notre Raspberry Pi pour sa stabilité.

5 – VNC Viewer

VNC Viewer est un logiciel de visualisation et de contrôle de l'environnement de bureau d'un ordinateur distant fonctionnant en SSH (Secure Shell). Il permet au logiciel client VNC de transmettre les informations de saisie du clavier et de la souris à l'ordinateur distant. Il nous a permis de travailler sur notre Raspberry Pi sans devoir monopoliser un écran, un clavier et une souris supplémentaires.



6 – HTML5 / CSS3 / PHP5 / JQuery/ Base de Données



Ce sont des langages informatiques. Nous nous sommes servis de HTML5, CSS3 et PHP 5 pour créer notre site internet, de Java Script pour récupérer des informations dans notre base de données afin de les afficher dans notre site internet. Nous avons créé et administré notre base de données MySQL à l'aide de PHPMyAdmin. Le jQuery nous a permis de dynamiser le site avec un rafraîchissement de page localisé.

F – Liens avec le développement durable

Le développement durable répond aux besoins de la population actuelle sans compromettre la capacité des générations futures à répondre aux leurs. Il s'agit d'une nouvelle conception de l'intérêt général, appliquée à la croissance économique et reconsidérée à l'échelle mondiale afin de prendre en compte les aspects environnementaux et sociaux d'une planète globalisée. Un projet s'installant dans le développement durable doit donc respecter les trois piliers fondamentaux de ce mode de développement, c'est à dire les piliers environnementaux (ne pas dégrader la biodiversité pour construire des usines, des résidences ...), économiques (ne doit pas être trop onéreux pour la population par rapport au type de prestations proposées) et sociaux (doit pouvoir être accessible à tous sans distinctions possible).

Notre projet respecte ces trois piliers. Un peu cher à l'achat mais rentable dans sur le long terme. Il fonctionne grâce à l'énergie solaire captée par un panneau solaire, énergie thermique convertie en énergie électrique qui sera ensuite stockée dans une batterie avant d'être utilisée pour faire fonctionner le système. Notre projet est destiné majoritairement pour les particuliers mais peut aussi être utilisé par des entreprises. De plus, il est peu encombrant et se fond bien dans le paysage, on ne le remarque quasiment pas.

G – Répartition des tâches

1 – Baptiste

Baptiste s'est occupé de la mise en place et du pilotage manuel des lumières intérieures et extérieures. Il a réalisé le diagramme de Séquence sur papier. Il aussi réalisé la maquette en bois avec des équerres, des planches de 30cm par 30 cm (soit 900 cm²) et des vis qu'il a achetées. Il a également fixé les planches en plexiglass que José a acheté. Il s'est occupé de mettre en place le pilotage du moteur pour notre store.

2 – Alexandre

Alexandre s'est occupé de la mise en place et du fonctionnement de l'anémomètre, a réalisé les diagrammes de Gantt et des exigences sur papier. Il s'est également occupé de mettre en place la base de données MySQL. Il a réalisé l'algorigramme de notre projet et le schéma de câblage.

3 – Valentin

Valentin s'est occupé de la mise en place et du fonctionnement du luxmètre. Il a également mis en place la connexion entre notre carte Arduino et notre base de données pour y écrire les informations pour que le site internet affiche en temps réel la quantité de vent et de lumière. Il a mis en place la connexion entre notre carte Arduino et notre base de données pour y lire les informations nécessaires au pilotage manuel. Il a réalisé sur papier le diagramme de cas d'utilisation et aussi l'algorithme du fonctionnement du projet.

4 – José (moi)

Je me suis occupé de l'installation et mise en place de notre carte Raspberry Pi 3, puis je l'ai configuré en tant que serveur web. Je me suis occupé de la création du site internet et dessiner les diagrammes des documents usuels sur Magic Draw. Enfin, à la fin de notre projet, je me suis chargé de filmer, puis monter la démonstration de son utilisation.

II – Partie Personnelle

A – Réception et mise en place du matériel

1 – Réception du matériel

Lorsque nous avons reçu notre matériel, je me suis occupé d'installer les versions des logiciels dont nous avons besoins pour notre projet (voir pages ZZ à TT) car, en plus d'être un des travaux pour notre diplôme du baccalauréat, nous avons participé aux olympiades de Sciences de l'Ingénieur qui s'est déroulée à Schneider Electric dans la zone industrielle de Carros. Nous avons donc reçu du matériel neuf (du clavier, à l'ordinateur en passant par tous nos différents capteurs listés pages 12 à 13).

2 – Installation du matériel

Pour notre carte Raspberry Pi 3, lorsque je l'ai mise sous-tension, je me suis rendu compte que la distribution préinstallée sur la carte n'était pas celle que nous désirions (il s'agissait de Arch linux qui été préinstallé et nous voulions utiliser Raspbian).

J'ai donc installé la version la plus récente de Raspbian sur notre carte (il s'agissait de la version ayant été distribuée en novembre 2017). J'ai fait en sorte e pouvoir avoir des permissions de super-utilisateur afin de procéder correctement à l'installation de ce dont nous avons besoin.

Une fois la distribution de Raspbian installée, j'ai téléchargé des paquets pour transformer notre carte en un véritable *(un paquet est une archive compressée comprenant des fichiers, des informations et des procédures nécessaires à l'installation d'un logiciel sur un système d'exploitation en s'assurant du bon fonctionnement de ce dernier)*.

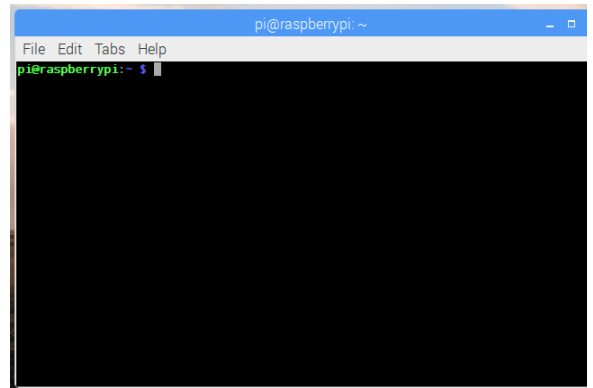
J'ai installé en premier lieu les paquets du logiciel VNC pour travailler sur notre carte « à distance » (c'est-à-dire sans lui dédier un écran, une souris et un clavier). Ensuite, j'ai installé les paquets contenant HTML5, CSS3, PHP5, jQuery et enfin PhpMyAdmin. Enfin, j'ai installé les paquets pour la version linux de notre éditeur de texte Visual Studio Code car l'éditeur de texte préinstallé sur notre carte Raspberry Pi 3 est doté de peu de fonctionnalités. Toutefois, j'ai régulièrement fait des mises à jour des paquets pour avoir la meilleure stabilité pour notre carte.

3 – Déroulement de l'installation

Pour ce faire, j'ai dû utiliser le terminal linux de notre carte.

Le terminal d'un ordinateur équipé d'un système d'exploitation basé sur Linux est un écran noir, qui est en attente d'une instruction, d'une commande Shell.

Une commande Shell est une instruction que l'ordinateur est capable d'interpréter et d'exécuter. C'est de cette manière que l'on peut installer des paquets et faire les mises à jour des logiciels déjà installé sur un ordinateur.



Un terminal linux

On installe un paquet avec la commande suivante :

```
Sudo apt-get install [nom du paquet/adresse web du paquet] -y
```

Sudo est une commande qui demande au système d'exécuter une action.

Apt-get est l'action de chercher un paquet.

Install est le champ qui demande d'installer un paquet.

-y permet de dire au terminal que nous sommes prêt d'installer le paquet désiré.

Ensuite, pour mettre à jour les paquets déjà installé on utilise la commande suivante :

```
Sudo apt-get update  
Sudo apt-get upgrade all
```

Update est le champ qui demande de vérifier quels paquets doivent être mis à jour.

Upgrade est le champ qui permet d'appliquer les mises à jour.

B – Création du site internet

Pour créer le site internet, il fallait que j'apprenne les langages HTML, CSS et PHP. J'avais quelques bases avec ces langages car j'avais réalisé mon propre site internet personnel deux années auparavant, et j'ai dû approfondir mes connaissances pour mener à bien notre projet.

L'HTML et le CSS sont des langages qui m'ont permis de créer le site en lui-même. L'HTML est celui qui permet d'écrire le texte, insérer des images, des liens, des tableaux, des formulaires...

Le CSS est un langage qui s'occupe de mettre en page ce que nous avons fait avec l'HTML, c'est-à-dire préciser ou mettre tel ou tel section du site, rajouter des couleurs, des animations...

Enfin PHP est un *langage orienté objet* fonctionnant sous forme de scripts. Il est très pratique car on peut insérer des pages html dans des fichiers PHP.

J'ai donc commencé avec la structure de base d'un site internet, puis j'ai construit pas à pas mon site.

1 – L'en-tête du site

J'ai commencé par créer l'en-tête du site en respectant la forme traditionnelle, c'est-à-dire :

- Une balise `<head>` pour dire au navigateur où commence l'en-tête et où il se termine.
- Plusieurs balises `<meta>` pour y inscrire des métadonnées utiles à mes fichiers CSS et Js.
- Une balise `<title>` pour inscrire le nom de la page dans l'onglet du navigateur.
- Plusieurs balises `<link>` pour inclure des fichiers à ma page.
- Des balises de Titre en veillant à les utiliser convenablement comme si je désirai mettre en ligne mon site et avoir une bonne visibilité sur les moteurs de recherches.
- Une partie pour la [navigation sur le site internet](#).

```
<html class="no-js" lang="fr">
<head>
  <!-- En tête du site -->
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Slegpi - Projet SIN 2018</title>
  <meta name="description" content="SLEGPI">
  <link href="css/hawthorne_type2_color3.css" rel="stylesheet">
  <link href="css/font-awesome.min.css" rel="stylesheet">
  <link href="css/displayers.css" rel="stylesheet">
  <link rel="shortcut icon" href="favicon.ico" type="image/x-icon">
  <link rel="icon" href="favicon.ico" type="image/x-icon">
  <script src="js/vendor/modernizr.js"></script>
  <div class="row">
    <div class="small-12 medium-12 large-12 small-centered columns">
      <header>
        <h1><a href="index.php">SLEGPI - Projet SIN 2018</a></h1>
        <h2><a href="index.php">Stores et Lumières Electriques Gérés Par Informatique</a></h2>
      </header>
    </div>
    <div class="small-12 medium-12 large-12 small-centered columns">
      <nav>
        <ul class="inline-list-custom">
          <li><a href="index.php" class="current">Piloter le système</a></li>
          <li><a href="about.php">A propos</a></li>
        </ul>
      </nav>
    </div>
  </div>
</head>
```

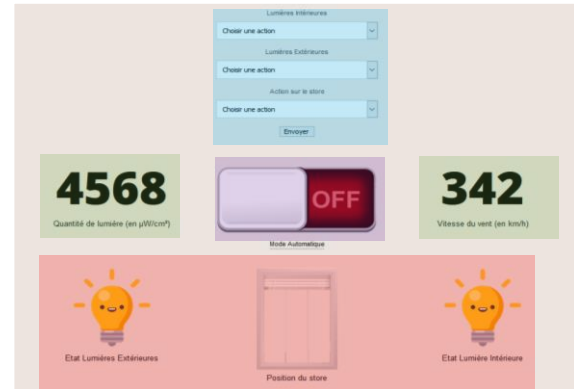
Ce qui nous permet d'obtenir ceci :



2 – Le contenu de la page d'accueil

En ce qui concerne la partie pour piloter le système, j'ai créé 9 sections organisées en rectangles de même taille. Voici cette page avec les sections importantes :

- La partie en **bleu** correspond à notre formulaire pour piloter le système en mode manuel.
- Les parties en **vert** correspondent à l'affichage des valeurs de notre base de données.
- La partie en **violet** correspond au bouton pour passer en mode automatique.
- La partie en **rouge** traite de l'affichage de l'état des parties opératives de notre système, mais j'explique son fonctionnement dans la partie consacrée à l'explication des scripts PHP page XX.



Voici les codes correspondant aux parties ci-dessus :

Partie	Code
<p>Le Formulaire</p>	<pre> <!-- Début Case 2 --> <div class="thumbnail"> <div class="thumbnail-img"> <form method="POST" action="send_sql.php" target="ma_popup" onsubmit="window.open('', 'ma_popup', 'width=1, height=1, crollbars=no, resizable=no ,directories=no ,location=no ,menubar=no ,... </div> <label for="Lumieres_Interieures">Lumières Intérieures</label> <select name="LI" id="Lum-Int"> <option value="">Choisir une action</option> <option value="non">Eteindre les lumières</option> <option value="oui">Allumer les lumières</option> </select> <label for="Lumieres_Exterieures">Lumières Extérieures</label> <select name="LE" id="Lum-Ext"> <option value="">Choisir une action</option> <option value="wrong">Eteindre les lumières</option> <option value="right">Allumer les lumières</option> </select> <label for="Action_sur_le_store">Action sur le store</label> <select name="S" id="Action-Store"> <option value="">Choisir une action</option> <option value="false">Descendre le Store</option> <option value="true">Monter le Store</option> </select> <input type="submit" name="Envoyer" value="Envoyer" id="send"/> </div> <!-- FIN Case 2 --> </pre>
<p>Affichage des valeurs de notre base de données</p> <p>+ Bouton de passage en mode automatique</p>	<pre> <!-- Début Case 4 --> <div class="thumbnail" id="r1-lux"> <div class="thumbnail-img"> <iframe src="display-lux.php" style="border:0px #ffffff hidden;" name="myIFrame" scrolling="no" frameborder="1" marginheight="0px" marginwidth="0px" height="120px" width="328px" allowfullscreen</iframe> </div> <div class="thumbnail-caption"> <p>Quantité de lumière (en µM/cm²)</p> </div> </div> <!-- FIN Case 4 --> <!-- Début Case 5 --> <div class="thumbnail"> <div class="thumbnail-img">
 <form method="POST" action="auto.php"> <input type="image" src="img/auto/off.png" border="0" alt="Submit" name="Envoyer" value="Envoyer" id="send" width="328", height="100"/> </form> </div> <div class="thumbnail-caption">Mode Automatique</div> </div> <!-- FIN Case 5 --> <!-- Début Case 6 --> <div class="thumbnail" id="r1-wind"> <div class="thumbnail-img"> <iframe src="display-vent.php" style="border:0px #ffffff hidden;" name="myIFrame" scrolling="no" frameborder="1" marginheight="0px" marginwidth="0px" height="120px" width="328px" allowfullscreen</iframe> </div> <div class="thumbnail-caption"> <p>Vitesse du vent (en km/h)</p> </div> </div> <!-- FIN Case 6 --> </pre>

3 – Page d'accueil et page « À propos »

SLEGPI - Projet SIN 2018

Stores et Lumières Electriques Gérés Par Informatique

Piloter le système A propos

Lumières Intérieures

Choisir une action

Lumières Extérieures

Choisir une action

Action sur le store

Choisir une action

2751


Quantité de lumière (en $\mu\text{W}/\text{cm}^2$)




Mode Automatique

342


Vitesse du vent (en km/h)



Etat Lumières Intérieures



Position du store



Etat Lumière Extérieures

SLEGPI - Projet SIN 2018

Stores et Lumières Electriques Gérés Par Informatique

Piloter le système A propos


A propos

Ceci est notre site internet pour piloter notre projet appelé SLEGPI, c'est à dire nos Stores et Lumières Electriques Gérés Par Informatique.

Notre Equipe

Notre équipe est composée de 4 lycéens :

- Alexandre
- Baptiste
- José
- Valentin



Localisation : Drap, France

Contact : jose.srifi@gmail.com

Résumés :

Travail d'Alexandre : [Télécharger le résumé](#)

Travail de Baptiste : [Télécharger le résumé](#)

Travail de José : [Télécharger le résumé](#)

Travail de Valentin : [Télécharger le résumé](#)

© 2018 SLEGPI. Projet SIN 2017/2018. jose.srifi@gmail.com

4 – Les scripts PHP

I – L'affichage des valeurs de notre base de données

Pour afficher la quantité de lumière et la vitesse du vent, j'ai dû écrire un petit script PHP pour aller lire dans notre base de données MySQL. Voici les scripts :

```
<?php
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "captbdd";
try
{
// Connection MySQL.
$dbdd = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
}
catch(Exception $e)
{
// Si il y a une erreur, arrêt du script.
die('Erreur : '.$e->getMessage());
}
// Récupération du contenu de la table capteurs
$reponse = $dbdd->query('SELECT lux FROM capteurs WHERE id=1');
?>
<!DOCTYPE html>
<html lang="fr" >
<head>
<link href="css/display-lux.css" rel="stylesheet">
</head>
<body>
<div class="displ-r1" id=lux>
<text>
<?php
while ($donnees = $reponse->fetch())
{
echo $donnees['lux'] . '<br />';
}
$reponse->closeCursor();
?>
</text>
</div>
</body>
</html>
```

Script pour lire et afficher la quantité de lumière

```
<?php
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "captbdd";
try
{
// Connection MySQL.
$dbdd = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
}
catch(Exception $e)
{
// Si il y a une erreur, arrêt du script.
die('Erreur : '.$e->getMessage());
}
// Récupération du contenu de la table capteurs
$reponse = $dbdd->query('SELECT vent FROM capteurs WHERE id=1');
?>
<!DOCTYPE html>
<html lang="fr" >
<head>
<link href="css/display-lux.css" rel="stylesheet">
</head>
<body>
<div class="displ-r1" id=vent>
<text>
<?php
while ($donnees = $reponse->fetch())
{
echo $donnees['vent'] . '<br />';
}
$reponse->closeCursor();
?>
</text>
</div>
</body>
</html>
```

Script pour lire et afficher la vitesse du vent

Leur fonctionnement est simple, je demande au programme de se connecter à notre base de données en lui attribuant un nom d'utilisateur, un mot de passe, l'adresse de notre base de données (ici localhost car la base de données et le site internet sont hébergés sur le même appareil).

Le nom de cette base et son format de caractères (ici UTF8 car nous pensions initialement devoir rentrer des mots avec caractères accentués pour « allumer » par exemple).

Ensuite, je leur demande d'exécuter une simple commande SQL pour aller chercher dans notre base de données la table *capteurs* et dans celle-ci, récupérer les valeurs *lux* et *vent* puis les afficher ensuite dans notre page d'accueil avec la balise `<iframe>` tout en lui appliquant un style pour l'adapter à la bonne taille.

La balise `<iframe>` permet d'insérer le contenu d'une page internet dans une autre.

II – Envoi des informations depuis le site vers Arduino et la base de données

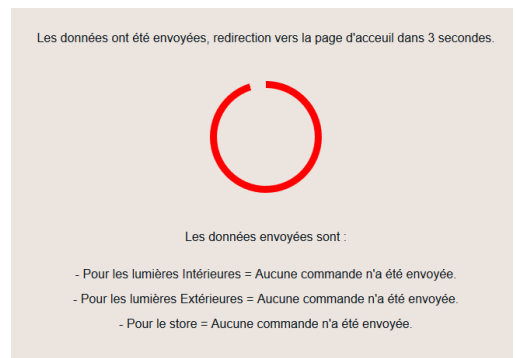
Pour envoyer les données depuis le site internet vers notre carte Arduino et notre base de données, j'ai d'abord dû récupérer les informations envoyées par notre formulaire (voir page 22).

```
if (isset($_POST["LI"])) $LI= $_POST["LI"]; else $LI ="";
if (isset($_POST["LE"])) $LE= $_POST["LE"]; else $LE ="";
if (isset($_POST["S"])) $S= $_POST["S"]; else $S ="";
```

Ensuite, je lie le script et ma base de données de la même manière que précédemment et en fonction de ce que je reçois, je demande au programme d'envoyer à notre carte Arduino une requête (Arduino sera alors en mode serveur), puis d'inscrire cette information dans notre base MySQL.

Sur cette page, j'ai ajouté une animation réalisée avec uniquement du CSS pour apporter un peu d'animation lors de l'envoi des données.

Voici le code correspondant à ces étapes :



```
<p>Les données ont été envoyées, redirection vers la page d'accueil dans 3 secondes.<br /><br />
<div class="radial-timer s-animate">
  <div class="radial-timer-half"></div>
  <div class="radial-timer-half"></div>
</div><br /><br />
Les données envoyées sont :<br />
<br /><br /> Pour les lumières Intérieures = <?php if ($LI == 'Y') {echo "Lumières Allumées";} elseif ($LI == 'non') {echo "Lumières Eteintes";} else {echo "Aucune commande n'a été envoyée.";}
<br /><br /> Pour les lumières Extérieures = <?php if ($LE == 'Z') {echo "Lumières Allumées";} elseif ($LE == 'wrong') {echo "Lumières Eteintes";} else {echo "Aucune commande n'a été envoyée.";}
<br /><br /> Pour le store = <?php if ($S == 'false') {echo "Abaissement du store";} elseif ($S == 'OUV') {echo "Ouverture du store";} else {echo "Aucune commande n'a été envoyée.";}</p>
<iframe src="http://10.66.240.59/LI-<?php echo $LI; ?>/LE-<?php echo $LE; ?>/S-<?php echo $S; ?>" height="1" width="1" Scrolling="no" frameborder="0"></iframe>
```

Code pour l'envoi vers notre carte Arduino

```
if ($LI == 'Y')
{
  $sql1 = "UPDATE `capteurs` SET `eInt` = '1' WHERE `capteurs`.`id` = 1";
  $stmt = $conn1->prepare($sql1);
  $stmt->execute();
}
elseif ($LI == 'non')
{
  $sql2 = "UPDATE `capteurs` SET `eInt` = '0' WHERE `capteurs`.`id` = 1";
  $stmt2 = $conn1->prepare($sql2);
  $stmt2->execute();
}

//=====
if ($LE == 'Z')
{
  $sql3 = "UPDATE `capteurs` SET `eExt` = '1' WHERE `capteurs`.`id` = 1";
  $stmt3 = $conn2->prepare($sql3);
  $stmt3->execute();
}
elseif ($LE == 'wrong')
{
  $sql4 = "UPDATE `capteurs` SET `eExt` = '0' WHERE `capteurs`.`id` = 1";
  $stmt4 = $conn2->prepare($sql4);
  $stmt4->execute();
}

//=====
if ($S == 'false')
{
  $sql5 = "UPDATE `capteurs` SET `Store` = '1' WHERE `capteurs`.`id` = 1";
  $stmt5 = $conn3->prepare($sql5);
  $stmt5->execute();
}
elseif ($S == 'OUV')
{
  $sql6 = "UPDATE `capteurs` SET `Store` = '0' WHERE `capteurs`.`id` = 1";
  $stmt6 = $conn3->prepare($sql6);
  $stmt6->execute();
}
```

Code pour l'envoi à notre base de données

III – Récupération des données pour la page d'accueil

Lorsque nous avons testé le site internet avec notre maquette, les valeurs envoyées depuis notre site internet se réinitialisaient lorsque nous rafraichissions notre page d'accueil. J'ai donc dû récupérer les valeurs envoyées par notre formulaire pour maintenir leur état pour notre carte Arduino.

Voici donc le code qui me permet de le faire :

```
// Connection MySQL.
$checkiframeeInt = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
}
catch(Exception $e)
{
}
// Si il y a une erreur, arret du script.
die('Erreur : '.$e->getMessage());
}
// Récupération du contenu de la table capteurs
$reponse = $checkiframeeInt->query('SELECT eInt FROM capteurs WHERE id=1');
$result = $reponse->fetch();
$count = $result[0];
//=====
try
{
}
// Connection MySQL.
$checkiframeeExt = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
}
catch(Exception $e)
{
}
// Si il y a une erreur, arret du script.
die('Erreur : '.$e->getMessage());
}
// Récupération du contenu de la table capteurs
$reponse1 = $checkiframeeExt->query('SELECT eExt FROM capteurs WHERE id=1');
$result = $reponse1->fetch();
$count1 = $result[0];
//=====
try
{
}
// Connection MySQL.
$checkiframeeStore = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
}
catch(Exception $e)
{
}
// Si il y a une erreur, arret du script.
die('Erreur : '.$e->getMessage());
}
// Récupération du contenu de la table capteurs
$reponse2 = $checkiframeeStore->query('SELECT Store FROM capteurs WHERE id=1');
$result = $reponse2->fetch();
$count2 = $result[0];
```

Partie 1 du code

```
if ($count == 1){
    $LI = 'V';
}
elseif ($count == 0){
    $LI = 'non';
}
else{
    $LI = '';
}

if ($count1 == 1){
    $LE = 'Z';
}
elseif ($count1 == 0){
    $LE = 'wrong';
}
else{
    $LE = '';
}

if ($count2 == 1){
    $S = 'false';
}
elseif ($count2 == 0){
    $S = 'OUV';
}
else{
    $S = '';
}
```

Partie 2 du code

5 – Animation CSS pour le chargement

Pour faire le cercle qui tourne pour le chargement après l'envoi des données, j'ai dû utiliser ce code que j'avais fait et publié sur codepen.io deux ans auparavant. :

```
html {
  background: #ECE5DF;
  text-align: center;
  padding-top: 36px;
}

body {
  display: inline-block;
}

.radial-timer {
  overflow: hidden;
  height: 144px;
  width: 144px;
  position: relative;
}

.radial-timer .radial-timer-half {
  height: 144px;
  width: 72px;
  border-radius: 72px 0 0 72px;
  background: red;
  position: absolute;
}

.radial-timer .radial-timer-half:nth-of-type(2) {
  z-index: 99999999;
  -webkit-transform-origin: center right;
  -webkit-transform: rotate(180deg);
}

.radial-timer .radial-timer-half:before {
  content: "";
  position: absolute;
  top: 9px;
  left: 9px;
  height: 126px;
  width: 63px;
  border-radius: 67.5px 0 0 67.5px;
  background: #ECE5DF;
}
```

Partie 1 du code

```
.radial-timer .radial-timer-half:after {
  content: "";
  position: absolute;
  background: #ECE5DF;
  height: 288px;
  width: 216px;
  left: -144px;
  top: -72px;
  -webkit-transform-origin: center right;
}

.radial-timer.s-animate {
  -webkit-transform-origin: center right;
}

.radial-timer.s-animate .radial-timer-half:nth-of-type(1):after {
  -webkit-animation: rotateLeftMask 3s infinite linear;
}

.radial-timer.s-animate .radial-timer-half:nth-of-type(2):after {
  -webkit-animation: rotateRightMask 3s infinite linear;
}

@-webkit-keyframes rotateLeftMask {
  0% {
    -webkit-transform: rotate(0deg);
  }
  50% {
    -webkit-transform: rotate(0deg);
  }
  100% {
    -webkit-transform: rotate(180deg);
  }
}

@-webkit-keyframes rotateRightMask {
  0% {
    -webkit-transform: rotate(0deg);
  }
  50% {
    -webkit-transform: rotate(180deg);
    visibility: hidden;
  }
  100% {
    -webkit-transform: rotate(180deg);
    visibility: hidden;
  }
}
```

Partie 2 du code

D – Problèmes Rencontrés

Lors de la création de mes scripts PHP, je n'avais pas installé les paquets de la fonction PDO, ce qui ne me permettait pas de lire dans ma base de données en utilisant une commande SQL.

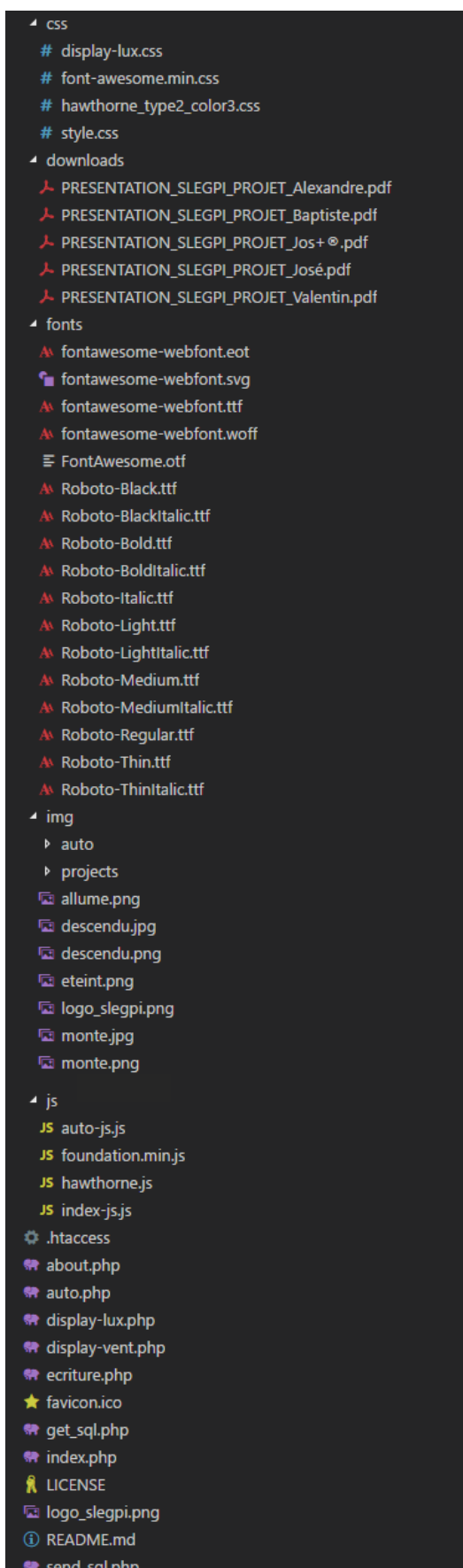
J'ai également eu des problèmes lors de l'installation de notre distribution de linux car, après une coupure de courant, notre carte micro-SD avait cramé.

Ensuite, lors des tests avec notre carte Raspberry et notre carte Arduino, lorsque nous actualisions la page, les valeurs du formulaire n'étaient pas conservées par le programme Arduino.

Enfin lorsque j'ai voulu installer WAMP server pour faire mes tests sur notre ordinateur, je n'avais pas les mêmes versions de PHP et de PHP My Admin que sur notre carte Raspberry Pi 3.

III – Annexes

A – Arborescence des fichiers du site internet



B – Répertoires des programmes du projet

Vous pouvez trouver l'intégralité des codes de notre projet en suivant ces liens :

Pour le site internet :

- <https://github.com/JunkJumper/SLEGPI-Website>

Pour les programmes Arduino

- <https://github.com/JunkJumper/SLEGPI-Arduino>

C – Participation aux Olympiades de Sciences de l'ingénieur

Notre professeur d'Enseignement Technologique Transversal nous a proposé de participer à ces olympiades pour avoir un retour de professionnels du secteur industriel et nous avons pris en considération leurs conseils afin de les appliquer pour le rendu final tel que le fonctionnement des envois entre la carte Arduino et notre serveur sur Raspberry Pi 3.

