Sommaire

[I. Réalisation du projet 2](#_Toc9457193)

[A. Rappel de la tâche de l’étudiant 2](#_Toc9457194)

[Rappel du cahier des charges 2](#_Toc9457195)

[Tâche réalisée 2](#_Toc9457196)

[Contraintes liées au développement 2](#_Toc9457197)

[B. Communication entre les différents éléments 3](#_Toc9457198)

[Synoptique 3](#_Toc9457199)

[Communication 3](#_Toc9457200)

[C. Informations permettant la connexion au serveur 4](#_Toc9457201)

[II. Réalisation du programme de l’application 4](#_Toc9457202)

[D. Croquis application 5](#_Toc9457203)

[E. Connexion MYSQL 7](#_Toc9457204)

[Fichier PHP 7](#_Toc9457205)

[III. Brumisateur 8](#_Toc9457206)

[A. Schéma électrique du câblage 8](#_Toc9457207)

[B. Evolution de l’hygrométrie 9](#_Toc9457208)

[IV. Test unitaire 10](#_Toc9457209)

[V. Fiches recettes 11](#_Toc9457210)

[A. Application Android 11](#_Toc9457211)

[VI. Conclusion 13](#_Toc9457212)

[A. Communication de groupe 13](#_Toc9457213)

[B. Vue critique du projet 13](#_Toc9457214)

[C. Connaissances apportées 13](#_Toc9457215)

[D. Tâches restantes 14](#_Toc9457216)

[E. Poursuite d’études 14](#_Toc9457217)

[VII. Annexe 15](#_Toc9457218)

# Réalisation du projet

## Rappel de la tâche de l’étudiant

### Rappel du cahier des charges

La serre doit pouvoir réguler automatiquement les trois grandeurs physiques qui, pour rappel, sont :

* l’Hygrométrie
* la Luminosité
* la Température.

L’utilisateur a le droit de choisir deux modes sur le système : le mode profil et le mode manuel.

Pour se faire, une carte de gestion (Raspberry) accueille le système et une carte Arduino fait le lien entre la carte de gestion et les actionneurs. Le système doit savoir quel est le mode utilisé par la serre :

* le mode profil : celui-ci récupère une valeur dans la base de données en fonction de l’heure ainsi que la grandeur physique.
* le mode manuel : ce dernier récupère une valeur dans la base de données en fonction de la grandeur physique.

### Tâche réalisée

Ma partie consiste en le développement d’une application Android permettant à l’utilisateur possédant la serre, de visualiser les mesures des capteurs en temps réel, et de changer les modes à souhait. Si le mode choisi est manuel, il n’y a qu’à rentrer les valeurs d’hygrométrie, de luminosité et température, à atteindre. Si un capteur est ajouté, il faudra les rentrer également pour le mode manuel.

Celle-ci consiste également en l’étude de l’impact du brumisateur sur le système, à savoir le développement d’une application permettant d’activer ou de désactiver l’actionneur, et d’envoyer les informations à la base de données.

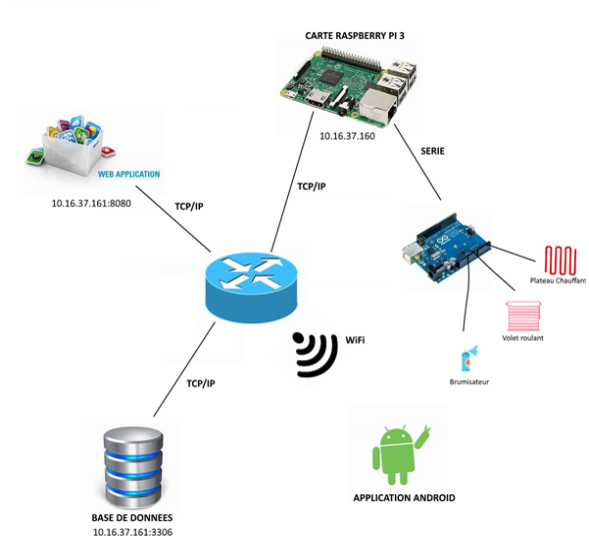
### Contraintes liées au développement

L’application doit être développé sur Android Studio, dans le langage Java.

Un téléphone Android est logiquement utilisé pour les tests pratiques.

## Communication entre les différents éléments

### Synoptique



1. Schéma du système de régulation

Ma partie du projet est entourée en rouge. Ces tâches consistent en la création d’une application gérant à distance les valeurs des capteurs à atteindre et leur activation. Pour que l’application fonctionne, il faudra être connecté au domaine sur lequel le serveur est installé. Cette connexion se fera via Wifi.

### Communication

La communication entre la base de données et l’application se fait grâce à une connexion Wifi permettant d’accéder au domaine. L’application appelle un fichier PHP stocké sur le serveur sfl6 qui va chercher les données dans la base de données du projet et qui ensuite encode ces informations dans un fichier format JSON pour que l’application les récupère.

## Informations permettant la connexion au serveur

Pour recevoir les informations nécessaires au fonctionnement de l’application, celle-ci doit être connectée au réseau informatique du secteur SN, via la Wifi.

Le serveur est configuré de la manière suivante :

* Adresse IP : 10.16.37.161
* Nom de la base de données : BDD\_Serre\_Automatique
* Nom utilisateur : sfl6
* Mot de passe : sfl6db

Ce sont donc les informations permettant au code PHP de trouver la base de données et de s’y connecter en tant qu’utilisateur, pour récupérer les informations voulues.

Le nom de la connexion Wifi sur lequel se connecter avec le téléphone test est : Wifi-BTSSN.

# Réalisation du programme de l’application

L’application mobile a donc été réalisée avec le logiciel Android Studio avec le langage JAVA.

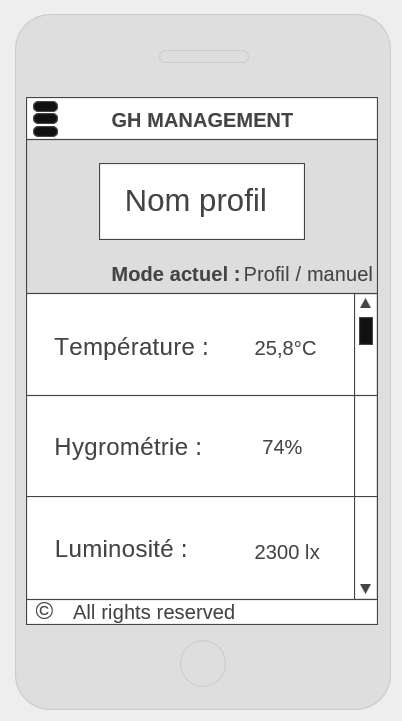
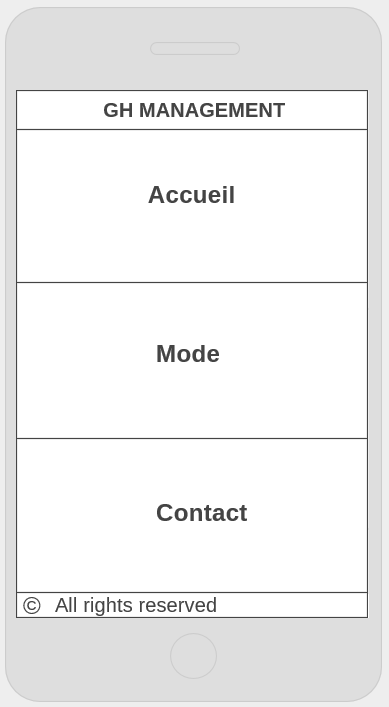
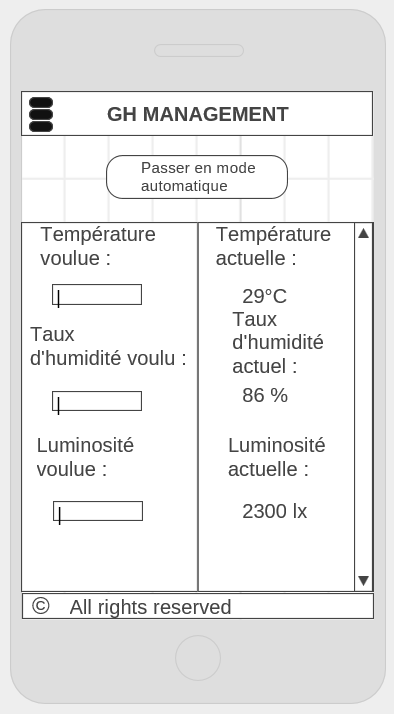
J’ai développé cette application sous l’API 22, (version 5.1 d’Android).

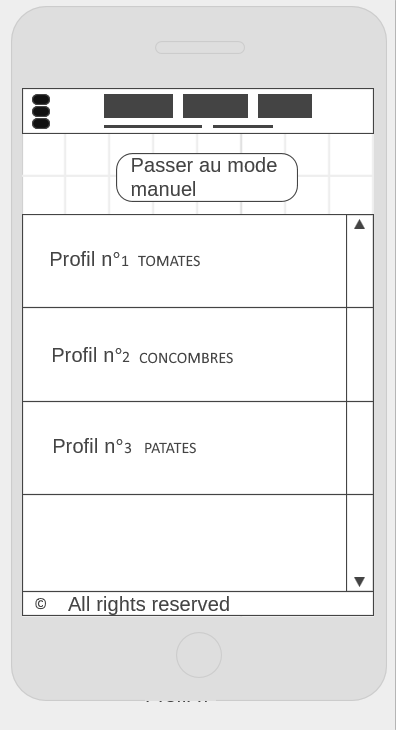
J’ai choisi cette version car je voulais qu’elle soit accessible pour la plupart des utilisateurs tout en restant assez jeunes pour avoir quelques fonctionnalités.

En admettant que ce projet serve un jour à un agriculteur, il y a des chances que cette personne ne possède pas un téléphone très haut de gamme. L’application doit donc pouvoir supporter le plus de plateformes possibles.

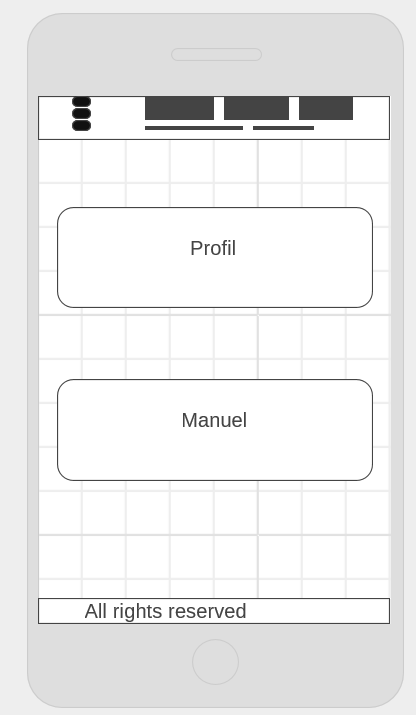
## Croquis application

Voici donc les différents croquis de l’application fait au préalable, pour me donner une idée du visuel de l’application :

1. Page d’accueil de l’application puis activité après avoir appuyé sur le bouton menu
2. Activité mode manuel



1. Affichage de l’activité « mode profil »



1. Affichage du mode à choisir (après avoir appuyé sur le bouton « mode » dans le menu.

Il y a donc quatre différents visuels sur l’application. La page d’accueil, qui s’affiche au démarrage de celle-ci, le menu, qui s’affiche à l’appui du bouton menu, le menu mode, qui s’affiche après avoir choisi le bouton « mode » dans le menu, et l’affichage du mode manuel ou profil selon celui qui est choisi.

## Connexion MYSQL

L’application appelle un fichier PHP stocké sur le serveur sfl6 qui va chercher les données dans la base de données du projet et qui ensuite encode ces informations dans un fichier format JSON pour que l’application les récupère.

### Fichier PHP

Le contenu du fichier PHP se présente comme ceci :

**<?php**

**$con=mysqli\_connect("10.16.37.161","sfl6","sfl6db","BDD\_Serre\_Automatique");**

**if (mysqli\_connect\_errno())**

**{**

**echo "Failed to connect to MySQL: " . mysqli\_connect\_error();**

**}**

**$sql = "SELECT \* FROM Actionneur";**

**if ($result = mysqli\_query($con, $sql))**

**{**

**$resultArray = array();**

**$tempArray = array();**

**while($row = $result->fetch\_object())**

**{**

**$tempArray = $row;**

**array\_push($resultArray, $tempArray);**

**}**

**echo json\_encode($resultArray);**

**}**

**mysqli\_close($con);**

**?>**

Ce bout de code PHP permet donc d’établir la connexion entre la base de données et l’application sous forme d’un fichier JSON après avoir récupéré les informations stockées dans la base de données.

Premièrement, il crée une variable contenant les informations permettant de se connecter à celle-ci.

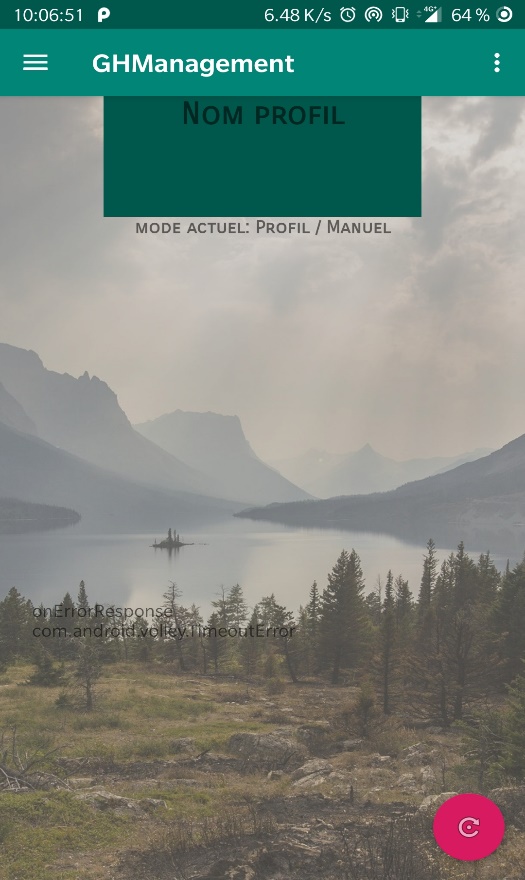
Ensuite, une condition if() permet de directement renvoyer une erreur et stopper la lecture du code en cas d’erreurs dans la variable « $con » contenant les informations de connexion. Sinon, on continue la lecture du code, avec la création d’une variable « $sql » permettant d’envoyer une requête à la base de données MYSQL. Cette requête sélectionne les informations à récupérer dans celle-ci.

On crée une variable « \*$result » qui vérifie le résultat obtenu. Enfin, les dernières variables servent à stocker les informations une fois vérifiées dans un tableau et ranger les informations une fois stockées par ordre du plus récent. Une fois tout ceci passé, les informations sont encodées sous un format JSON et envoyées à l’application.

**[{"id":"1","port":"5","name":"Brumisateur","physicalId":"1"},{"id":"2","port":"6","name":"Moteur","physicalId":"2"},{"id":"3","port":"7","name":"Plateau Chauffant","physicalId":"3"}]**

Voici un exemple d’informations récupérées sur une base de données encodées via PHP sous le format JSON.

Ici on récupère les id, port, noms, et id physiques de la table « actionneur » du projet.



Bouton affichant des paramètres si besoin est

Bouton permettant d’ouvrir le menu

Bouton affichant les informations récupérées dans la base de données

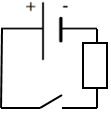
1. Page d’accueil application android

Voici donc la page d’accueil de l’application Android, c’est l’activité qui s’affiche au lancement de l’application.

# Brumisateur

Module d’alimentation

## Schéma électrique du câblage



Module de puissance

Brumisateur

1. Schéma de câblage de l’actionneur intégré au système (Brumisateur))

Pour activer le brumisateur, il suffit de lui envoyer du courant. Ce brumisateur fonctionne avec un courant de 18 V. Un module de puissance, c'est un ensemble de semi-conducteurs de puissance interconnectée entre eux dans le même boitier. Il permet de commander un élément de puissance à courant continu exemple : Moteur, résistance, ampoule. Le module de puissance utilisé dans ce système est le K\_AP\_MPWR. Il possède un MOSFET IRL520N, pour changer son état, il suffit d’envoyer une tension. On peut donc dire qu’il fonctionne comme un interrupteur ON/OFF.

## Evolution de l’hygrométrie

Lors de l’activation ou la désactivation des actionneurs (ici brumisateur), un temps d’attente débute. Pendant ce temps, toute action est impossible. Cette attente est stockée dans la base de données pour chaque actionneur. La colonne qui concerne cette période s’appelle « Periode\_Action » Pour déterminer le temps d’attente, des tests ont été établis.

# Test unitaire

# Fiches recettes

## Application Android

|  |  |
| --- | --- |
| **Nom :** Lancer l’application Android | |
| **Recette :** Technique | |
| **Objectif** | Savoir si l’application Android fonctionne |
| **Elément à tester** | Connexion MYSQL |
| **Pré requis** | Android Studio, plus un téléphone sous Android |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Scénario** | | | | |
| **Id** | **Démarche** | **Données** | **Comportement attendu** | **OK** |
| **1** | Taper dans la barre de recherche le nom du logiciel à ouvrir | Android Studio | Le nom du logiciel s’affiche en suggestion dans la fenêtre intégrée à Windows qui s’ouvre. | ❑ |
| **2** | Cliquer sur la suggestion qui s’affiche en premier | Android Studio, icône verte avec un compas | L’application est lancée. | ❑ |
| **3** | Brancher le téléphone au pc | Téléphone Android | Le téléphone est branché au pc,  Il reçoit une notification lui demandant la nature de ce lien. | ❑ |
| **4** | Définir le téléphone sur « partage de connexion via USB » | Fenêtre de notification Android | L’ordinateur se connecte au réseau du lycée. | ❑ |
| **5** | Une fois le projet ouvert, appuyer sur les touches « Maj + F10 » | Projet GHManagement ouvert | Une fenêtre proposant l’appareil auquel se connecter pour la compilation apparaît. | ❑ |
| **6** | Appuyer sur la touche « Entrée » | Clavier ordinateur | Le projet compile sur le premier téléphone choisi. | ❑ |
| **7** | Appuyer sur le bouton « Parse » | Bouton blanc « Parse » | Les informations de la table « Actionneur » de la base de données MYSQL de notre projet s’affiche sur l’application. | ❑ |

|  |  |  |  |
| --- | --- | --- | --- |
| Rapport de test | Testé par : L’Etudiant 3 | Le : 24/04/2019 | |
| **Conformité**  ❑ Excellente  ❑ Moyenne  ❑ Faible | | | |
| **Commentaire :** | | | **Approbation :** |

# Conclusion

## Communication de groupe

Durant tout le projet nous avons essayé de mettre en place une cohésion de groupe. Pour cela, plusieurs outils et méthodes ont été utilisés :

* La plateforme GitHub et le logiciel GitHub Desktop qui nous a permis de structurer nos ressources et de trier nos données tout ça sous une plateforme réseau accessible à distance.
* Diagramme de Gantt pour l’éclaircissement de nos objectifs et la planification de nos tâches.
* Des chartes graphiques sous Word et sous PowerPoint communes ont été créées au préalable pour faciliter et optimiser notre travail.
* Des rendus hebdomadaires ont été effectués durant le projet pour rendre compte de l’avancée.

Ces méthodes m’ont permis d’obtenir une seconde expérience de groupe à travers un gros projet (le dernier étant en terminale STI2D) et de renforcer mes liens avec les autres étudiants. Ces échanges nous ont permis de mieux comprendre le fonctionnement du projet surtout sur des points techniques, notamment sur la base de données qui est commune avec le groupe SFL5 « Serre Automatique Système d’acquisition ».

Je remercie les étudiants MAGONI Baptiste et PERRIN Goulven pour leur participation au projet.

## Vue critique du projet

Ce projet-ci m’a paru être assez intéressant, du moins au premier abord. C’est un super exemple de système qu’on pourrait appliquer au domaine de la domotique dans un futur proche par exemple, j’ai beaucoup aimé ce lien.

Maintenant, je ne pense pas que l’entreprise ayant commandité ce projet soit réellement intéressé par celui-ci, ce qui est un peu décevant. Le côté pratique d’un projet rajoute de la motivation, j’ai trouvé ça dommage.

## Connaissances apportées

Je n’avais jamais développé d’application Android, ce qui fut une expérience peu commune car j’ai passé un temps fou à d’abord m’habituer au langage Java qui, sous Android était un petit peu modifié, et donc plus difficile à prendre en main.

Je n’avais également jamais codé en classe sous python

Durant le projet, je suis allé aussi m’informer auprès des autres étudiant à propos de leur projet, puisque nous étions beaucoup à avoir comme tâche de développer une application mobile, alors nous nous sommes mutuellement aidés pour obtenir de meilleurs résultats.

## Tâches restantes

Au moment de la rédaction de ce rapport, il me reste à :

* Etudier l’impact du brumisateur dans la serre, à savoir, le temps que le brumisateur mettra à augmenter l’humidité dans la serre jusqu’à un certain seuil.
* Faire en sorte de pouvoir écrire via l’application dans la base de données.
* Développement des dernières activités (onglets) de l’application.

Ce que je pourrais améliorer par la suite :

* Peaufinage du design
* Ajout automatique de capteurs

## Poursuite d’études

A la suite de plusieurs tests, j’ai donc été accepté pour un master en informatique à l’école Sup’ESAIP, une licence ASSTRI à l’école du CNAM sécurité défense à Saint-Brieuc, et pour finir, un bachelor informatique auprès de l’école IMIE. Seule la formation du CNAM n’est pas en alternance. Je pense rester à Nantes pour poursuivre ma formation dans le domaine de l’informatique à l’école IMIE.

J’ai déjà quelques pistes, mais il ne me reste qu’à trouver une entreprise pour mon parcours en alternance.

J’ai dans l’objectif futur d’obtenir des compétences en management pour pouvoir par la suite gérer une équipe de techniciens dans le domaine de l’informatique.

# Annexe

1. Fichier PHP

**<?php**

**$con=mysqli\_connect("10.16.37.161","sfl6","sfl6db","BDD\_Serre\_Automatique");**

**if (mysqli\_connect\_errno())**

**{**

**echo "Failed to connect to MySQL: " . mysqli\_connect\_error();**

**}**

**$sql = "SELECT \* FROM Actionneur";**

**if ($result = mysqli\_query($con, $sql))**

**{**

**$resultArray = array();**

**$tempArray = array();**

**while($row = $result->fetch\_object())**

**{**

**$tempArray = $row;**

**array\_push($resultArray, $tempArray);**

**}**

**echo json\_encode($resultArray);**

**}**

**mysqli\_close($con);**

**?>**

Annexe 2 Méthode JSONParse

protected void jsonParse() {  
 String url2 = "http://88.136.244.147:8080/regulation/actuator.php";  
 String url = "http://10.16.37.161/AndroidStudio/connecPHPv2.php";  
 StringRequest stringRequest = new StringRequest(Request.Method.*POST*, url, new Response.Listener<String>() {  
 @Override  
 public void onResponse(String response) {  
 try {  
 JSONArray jsonArray = new JSONArray(response);  
  
 for (int i = 0; i < jsonArray.length(); i++) {  
 JSONObject actionneur = jsonArray.getJSONObject(i);  
 int id = actionneur.getInt("id");  
 int port = actionneur.getInt("port");  
 String name = actionneur.getString("nom");  
 int pe\_action = actionneur.getInt("Periode\_Action");  
 int phys\_measure\_id = actionneur.getInt("Grandeur\_Physique\_id");  
 //mTextViewResult.append("\n" + String.valueOf(id) + ", " + name + ", " + String.valueOf(port) + String.valueOf(pe\_action) + ", " + phys\_measure\_id + "\n\n");  
 TextView tv = new TextView(MainActivity.this);  
 tv.setText("\n" + String.*valueOf*(id) + ", " + name + ", " + String.*valueOf*(port) + String.*valueOf*(pe\_action) + ", " + phys\_measure\_id + "\n\n");  
 tv.setBackground(ContextCompat.*getDrawable*(getApplicationContext(), R.drawable.*textview\_border*));  
 tv.setTextColor(Color.*RED*);  
 tv.setTypeface(Typeface.*DEFAULT\_BOLD*);  
 ll.addView(tv);  
 }  
 } catch (JSONException e) {  
 e.printStackTrace();  
 TextView tv = new TextView(MainActivity.this);  
 tv.setText("EXCEPTION");  
 tv.setBackground(ContextCompat.*getDrawable*(getApplicationContext(), R.drawable.*textview\_border*));  
 tv.setTypeface(Typeface.*DEFAULT\_BOLD*);  
 ll.addView(tv);