HAUTE ÉCOLE DE LA COMMUNAUTÉ FRANÇAISE EN HAINAUT
Catégorie Technique
8a Avenue Victor Maistriau, 7000 Mons

Rapport

Applications sur Android

Projet : application de monitoring sur A.P.I. et commandes à distance

Cammarata Alessio
Bloc 3 - Développement

Titulaire : monsieur Scopel

Année académique : 2016 - 2017

Table des matières

Introduction

Dans le cadre du cours de "Applications sur Android", un projet a été demandé aux élèves : mettre en place une application de monitoring qui pourrait se connecter en Wi-Fi à un automate via un smartphone. Cette application permettrait de visualiser les informations de cet automate, mais également de préparer le terrain pour des modifications futures.

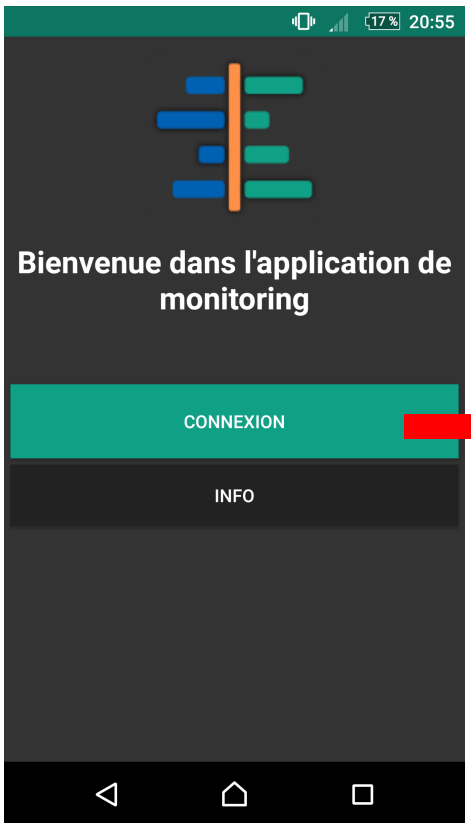
Présentation du projet

Pour réaliser cette application, j'ai utilisé **Android Studio** (logiciel qui permet de créer des applications pour Android, via Windows ou Mac), **ProcesSimS7 Pro** (logiciel utilisé pour concevoir des réseaux électroniques mis à disposition par la *HEH Campus Technique*), ainsi que **SIMATIC Manager** et **WinLC RTX** (qui permettent de créer les liaisons entre la machine et le Wi-Fi, transformant l'ordinateur en automate).

Avec toutes ces ressources à disposition, l'application finale aura plusieurs fonctions :

- supervision à distance des processus industriels (via une interface soignée);
- gestion des utilisateurs (lecture et lecture/écriture);
- modification depuis l'application de l'adresse IP souhaitée, ainsi que les numéros de rack et de slot souhaités;
- modification des variables spécifiques dans l'automate (via une modification future);
- permettre d'avoir un maximum d'informations concernant l'automate auquel l'application est connectée.

Captures d'écran



Lorsque l'application se lance, une interface sobre apparait. Le logo de l'application, un message de bienvenue, et deux boutons.

Le premier bouton "Connexion" mène à la vue du même nom permettant de se connecter.

Le second bouton "Info" ouvre une petite fenêtre qui apporte des informations sur l'application.

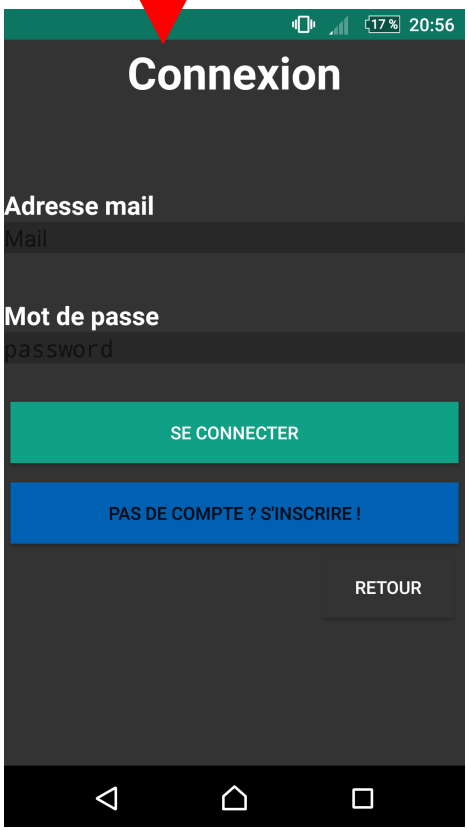
Dans la vue "Connexion", un premier EditText permet à l'utilisateur de remplir ce champ via son adresse mail.

Un second EditText doit être rempli avec un mot de passe (celui en accord avec l'adresse mail).

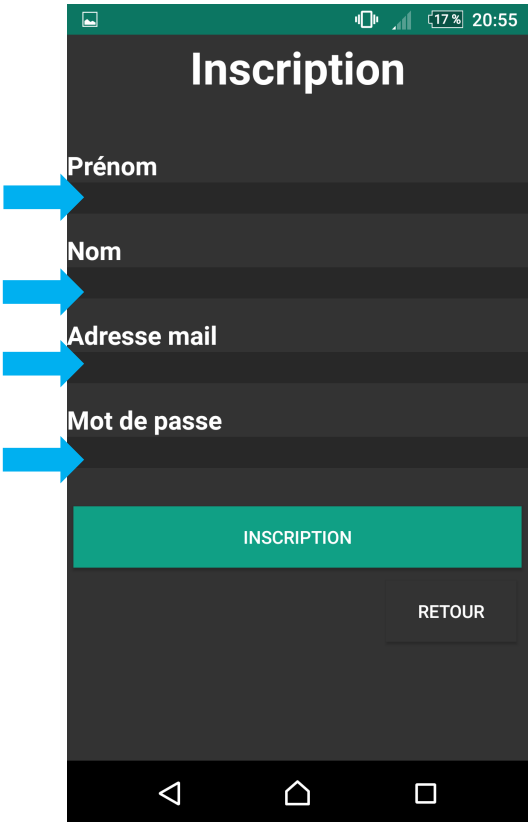
Ensuite, le premier bouton "Se connecter" permet d'aller vers la vue des applications, à condition que l'adresse mail et le mot de passe soient en accord et soient validés. Dans le cas inverse, l'utilisateur est renvoyé au menu et il faut vérifier les identifiants.

Et si vous n'avez pas de compte, vous pouvez vous inscrire via le deuxième bouton qui mènera vers la vue "Inscription".

Le bouton "Retour" ramène au menu.



Inscription



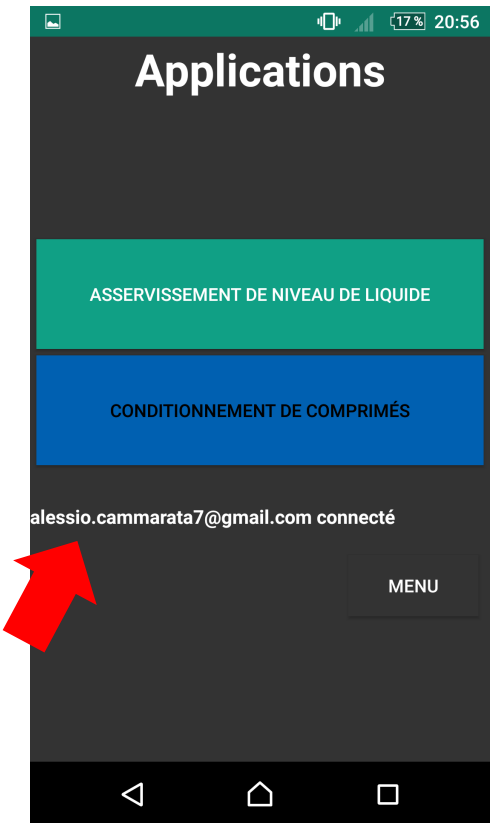
Si l'utilisateur n'a malheureusement pas de compte (et c'est bien dommage), il peut s'inscrire ici dans la vue "Inscription". L'utilisateur est invité à remplir obligatoirement ces 4 champs afin de pouvoir s'inscrire.

Le prénom et le nom doivent faire au minimum 3 caractères, l'adresse mail minimum 10 caractères et le mot de passe minimum 4 caractères.

Si un des champs est mal rempli ou ne respecte pas les conditions, un *Toast* (petite information brève sur l'écran) demande à l'utilisateur de corriger l'erreur.

Mais si tout est respecté, un *Toast* mentionne à l'utilisateur qu'il est connecté, revient automatiquement à la vue précédente "Connexion" et l'utilisateur peut entrer ses identifiants.

Choix des applications



Une fois les bons identifiants entrés et que l'utilisateur a appuyé sur le bouton "Se connecter", il arrive sur cette vue qui lui permet de choisir l'application souhaitée.

Le premier bouton mène à l'asservissement de niveau de liquide, et le second permet d'aller au conditionnement de comprimés.

Un TextView indique également l'adresse de l'utilisateur connecté.

Le dernier bouton nous renvoie au menu.

Asservissement de niveau de liquide

Une fois l'application concernant l'asservissement de niveau de liquide choisie, l'interface sobre mais simple permet de rapidement comprendre le fonctionnement. Les 3 champs à remplir permettent de compléter respectivement **l'adresse IP** à laquelle l'utilisateur souhaite se connecter, **le rack** adéquat et **le slot** souhaité (il faut au préalable que le smartphone soit connecté au même réseau Wi-Fi que l'automate).

Une fois ces champs remplis, l'utilisateur peut appuyer sur le bouton "Se connecter". Si tous les champs sont bien complétés, les informations de connexion apparaissent au bas de l'écran, en vert.

Une fois connecté, les informations à l'écran sont remplacées en direct par les informations de l'automate. Le bouton "Se connecter" devient "Se déconnecter", pour se déconnecter de l'automate.

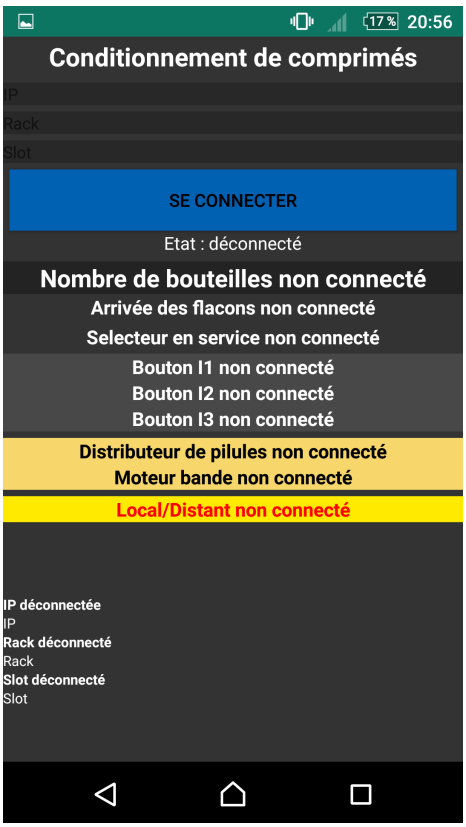
Conditionnement des comprimés

D'un point de vue visuel mais également au niveau des fonctionnalités, les deux applications sont très similaires.

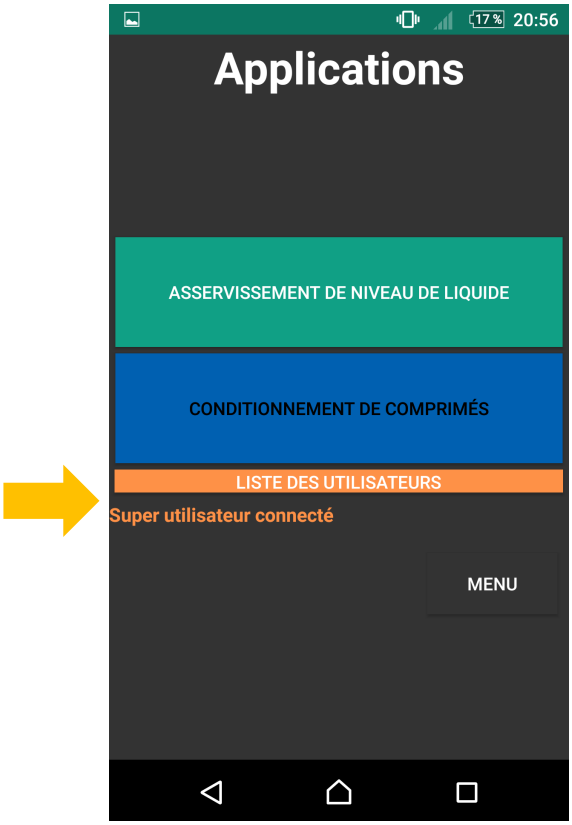
Dans cette vue dédiée au conditionnement des comprimés, les champs à remplir pour l'adresse IP, le rack et le slot sont toujours présents.

Le bouton permet toujours de se connecter (ou se déconner), et une fois la connexion effectuée, les informations sont remplacées en direct par celles de l'automate.

L'utilité de 2 vues distinctes est une question de confort visuel. En effet, l'utilisateur pourra directement choisir son application et avoir une vue dédiée à celle-ci.

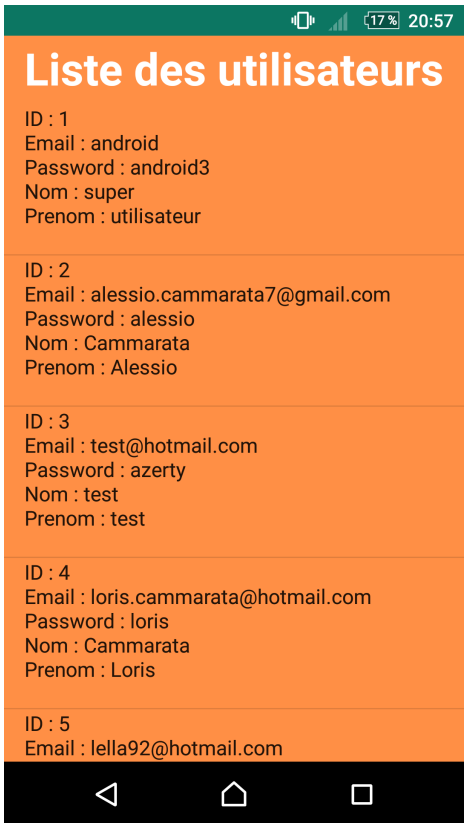


Le super utilisateur



Un cas particulier dans l'application est le cas du **super utilisateur**. Dans la vue "Connexion", si l'adresse est "**android**" et le mot de passe "**android3**", la vue "Applications" change quelque peu. L'indicateur de connexion mentionne que le super utilisateur est connecté, et un nouveau bouton "Liste des utilisateurs" apparait.

Le super utilisateur a donc la possibilité de voir, via ce bouton qui mène à une autre vue, les noms et prénoms de chaque utilisateur inscrit ainsi que leurs adresses et mots de passe.

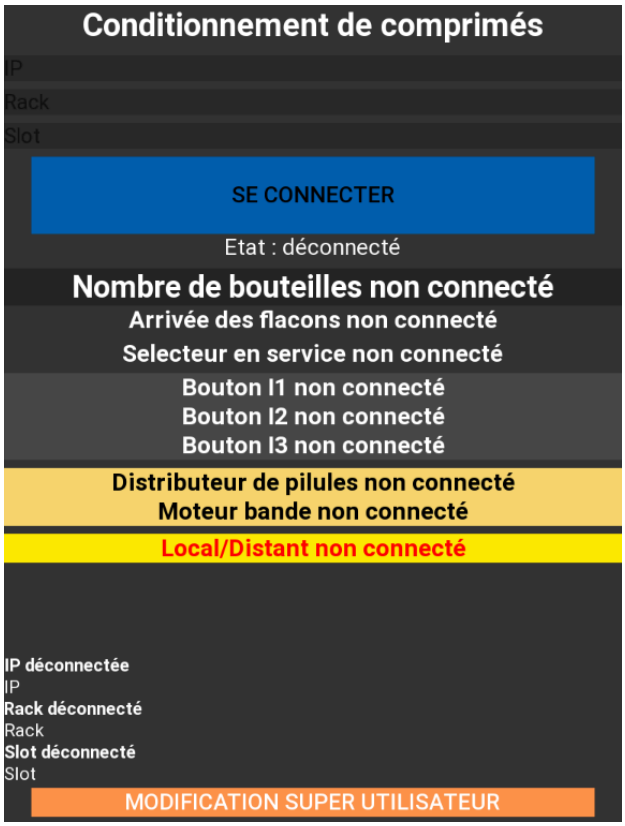


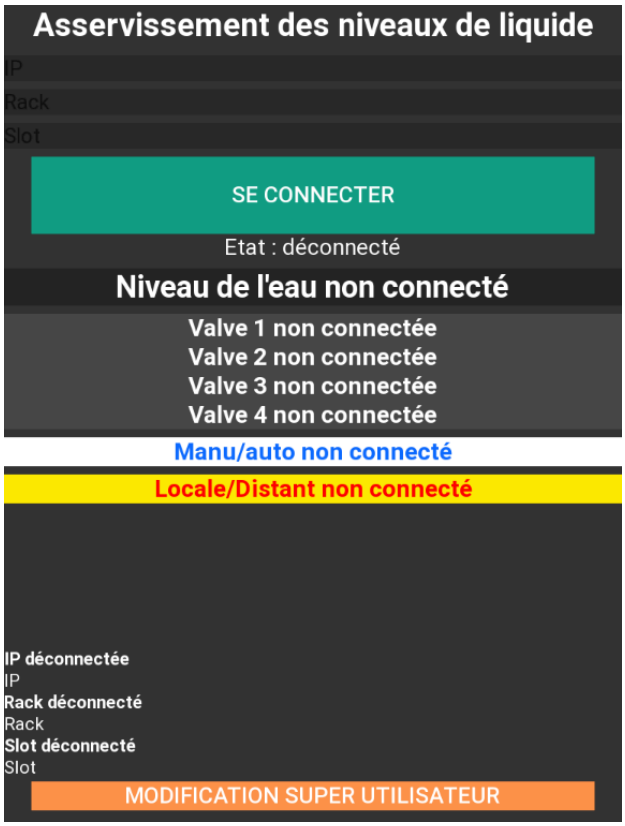
<u>Pourquoi afficher la liste des utilisateurs ainsi que leurs informations ?</u>

Imaginons que cette application soit utilisée dans une société et que chacun utilise cette application de manière centralisée.

Un employé ne se souvient plus de ses identifiants et ne peut donc plus se connecter, rendant l'application inutilisable.

Plutôt que de se refaire un nouveau compte, il peut demander à l'administrateur d'entrer ses identifiants de super utilisateur, et pourra ainsi récupérer ses identifiants. Cela permet à l'utilisateur de récupérer ses identifiants via son administrateur, et permet à ce dernier de garder une copie des identifiants de ses employés.





Si on est connecté en mode super utilisateur, un nouveau bouton apparait dans les deux applications

Il n'est cependant pas, à l'heure actuelle, mis au point. Il est présent dans le code de l'application mais n'a aucune fonction. Il permettra, une fois initialisé, de modifier les indications à l'écran et en même temps de modifier celles de l'automate.

Il a pour but la fonction d'écriture et permettrait donc de gérer l'automate à distance.

Analyses

Vérification d'inscription

Lorsque l'utilisateur s'inscrit pour la première fois, il doit entrer son prénom, son nom, son adresse mail et son mot de passe. Grâce aux vérifications mises en place, l'inscription n'est pas valide si le prénom fait moins de 3 caractères, pareil pour le nom.
L'adresse mail ne doit pas faire moins de 10 caractères (et ne pas être déjà utilisée), le mot de passe quant à lui doit faire minimum 4 caractères. Tous les champs doivent être remplis via ces règles.

```
if(prenom.getText().toString().length() < 3)//Si prénom vide ou plus petit de 3 caractères, refusé
{
    Toast.makeText(this, "Veuillez entrer un prénom correct", Toast.LENGTH_SHORT).show();
    flag = false;
}

else if(nom.getText().toString().length() < 3)//Si nom vide ou plus petit de 3 caractères, refusé
{
    Toast.makeText(this, "Veuillez entrer un nom correct", Toast.LENGTH_SHORT).show();
    flag = false;
}

else if (count == 1)//Mail entré existant, donc refusé
{
    Toast.makeText(this, "Cette adresse mail est déjà utilisée", Toast.LENGTH_SHORT).show();
    flag = false;
}

else if(email.getText().toString().length() < 10)//Si mail vide ou plus petit que 10 caractères, refusé
{
    Toast.makeText(this, "Veuillez entrer un mail correct", Toast.LENGTH_SHORT).show();
    flag = false;
}

else if(password.getText().toString().length() < 4)//Si MDP vide ou plus petit de 4 caractères, refusé
{
    Toast.makeText(this, "Veuillez entrer un mot de passe correct", Toast.LENGTH_SHORT).show();
    flag = false;
}

else if(prenom.getText().toString().isEmpty() || nom.getText().toString().isEmpty() || email.getText().toString().isEmpty() || password.getText().toString().isEmpty())
{
    Toast.makeText(this, "Veuillez remplir tous les champs", Toast.LENGTH_SHORT).show();
    flag = false;
}

else if(flag) //verrou de controle
{
    User user1 = new User(email.getText().toString(), password.getText().toString(), nom.getText().toString(), prenom.getText().toString());
    //ajout de la ligne de l'utilisateur dans la bdd
    userDB.openForWrite();
    userDB.insertUser(user1);
    userDB.Close();
    Toast.makeText(this, "Vous êtes inscrit(e)", Toast.LENGTH_SHORT).show();
    finish();

    Intent connexion = new Intent(this, ConnexionActivity.class);
    finish();
    startActivity(connexion);
}
break;
```

Le flag

Le flag est un bit initialement positionné sur "true" ou "false" (selon notre convenance) qui permet de faire pas mal de choses. Dans le cas de ce code, il est initialement placé sur "true". Cependant, dans l'exemple de l'inscription, il faut qu'il reste placé sur "true" pour que l'inscription soit validée.

Ici, je mentionne que si une condition (ou plus) n'est pas respectée (par exemple mot de passe trop court), le flag passe en "false", et donc l'inscription est impossible.

Ce principe est utilisé de la même manière pour l'étape de la connexion.

Super utilisateur

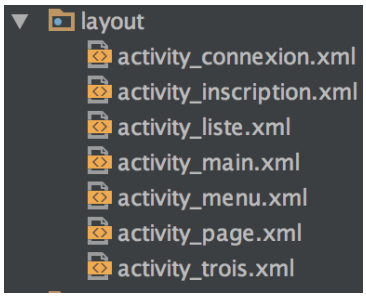
Le super utilisateur est présent dans la base de données, comme toutes les autres informations des utilisateurs.
Cependant, j'ai décidé de l'implémenter directement dans le code.

```
@Override
public void onCreate(SQLiteDatabase db) {
    db.execSQL(CREATE_DB);
    db.execSQL("INSERT INTO table_user (EMAIL, NOM, PRENOM, PASSWORD) VALUES ('android', 'super', 'utilisateur', 'android3')");
}
```

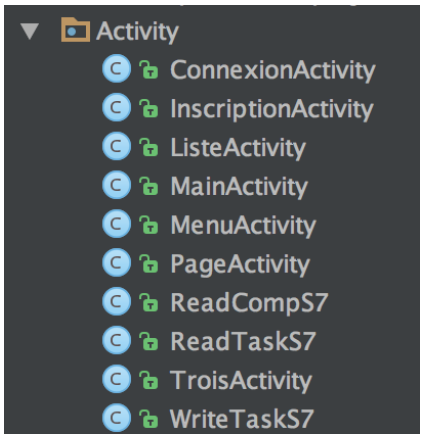
Etant donné que cet utilisateur doit être présent pour des utilisations spécifiques dès le démarrage de l'application, il me semblait logique qu'il soit présent directement.

Classes

Parmi les classes utilisées, j'ai regroupé certaines d'entre-elles afin qu'une modification future par moi ou une autre personne soit plus facile.

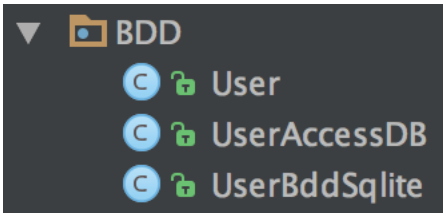


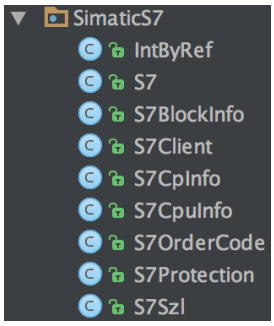
Les **activités** à interfaces visuelles (ainsi que les classes dédiées) sont classées respectivement dans "layout" et dans un dossier créé "Activity". Par ordre alphabétique, elles permettent de modifier l'interface des vues ainsi que les codes liés à celles-ci.



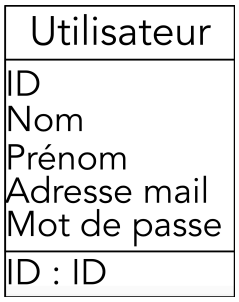
Certaines classes sont également dans ce dossier, notamment les classes qui gèrent les tâches de fond et les actions lors de l'appui sur certains boutons. Ces classes permettent également de récupérer l'adresse IP, le numéro du rack ainsi que celui du slot pour que ces informations puissent être utilisées.

Tout ce qui concerne la base de données se trouve dans le dossier "BDD". Que ce soit la gestion des utilisateurs, la génération de la base de données ou encore les éléments essentiels de cette dernière (ID, nom, prénom…).





Enfin, concernant les éléments indispensables pour la connexion à l'automate, il y a toutes les classes dans le dossier "SimaticS7". Elles vont de la liaison par le Wi-Fi jusqu'à l'identification de l'automate via certaines variables.



Voici un schéma conceptuel de la base de données.
Très simple, cependant toutes les informations nécessaires et utiles sont présentes. Un ID est associé aux informations de l'utilisateur (nom, prénom, adresse mail, mot de passe) de sorte à être retrouvé plus facilement. Ensuite des vérifications sont effectuées pour qu'une adresse mail ne soit pas utilisée 2 fois.

Les points importants

Points forts

D'un point de vue purement personnel, cette application m'a aidée. Le langage Android étant proche du langage Java, j'avais quelques appréhensions avant de commencer ce travail car j'ai du mal avec le langage Java.
Cependant, travailler sur le langage Android a été instructif. Cela m'a d'abord aidé à mieux comprendre le Java mais j'avais surtout une certaine motivation d'étape en étape : réussir à faire interagir des éléments entre eux, ensuite dialoguer avec l'automate, etc. Chaque chose à faire était pour moi compliquée et représentait une sorte de défi, mais je suis quand même parvenu à réaliser certaines choses qui avant me semblaient impossibles.

Enfin, je pense avoir fait en sorte que mon application soit accessible et lisible pour tous visuellement. J'ai utilisé en intégralité des **Linear Layout** pour faire les interfaces. Comme ça, chaque élément est placé l'un en-dessous de l'autre et permet une meilleure lisibilité.

Surtout, si l'application doit être amenée sur un autre périphérique (smartphone plus grand/plus petit ou même une tablette), il n'y aura pas de souci de lisibilité : étant donné que tous les éléments sont placés les uns à la suite des autres et que leur longueur est réglée de manière automatique, aucun élément n'ira mal se placer et ne sera pas caché.
Tous les éléments sont placés de manière réfléchie mais également pour rester dans une interface sobre.

J'ai également joué avec les couleurs pour signaler quand un élément est fonctionnel ou non.

Problèmes rencontrés

La gestion des utilisateurs avec la base de données n'a pas été chose simple, et la gestion des inscriptions et connexion non plus. Cependant, les problèmes majeurs qui m'empêchaient d'avancer ont été réglés.
L'application ne permettait pas au début de pouvoir se connecter à l'adresse IP souhaitée, je ne trouvais pas de quelle manière la modifier. Cependant, ce souci a été résolu.

Cahier de charge demandé

Lorsque le projet a été demandé, un cahier de charge nous a été imposé pour que l'application puisse effectuer certaines commandes bien précises.

Demandes respectées

- Avoir une interface soignée afin d'afficher les informations d'entrées et de sorties "machine" de façon claire
- Modifier certaines infos au départ de zone de texte (adresse IP, rack, slot)
- Gestion des utilisateurs
- Un utilisateur est représenté par son nom, prénom, adresse mail et mot de passe
- Un utilisateur se connecte via adresse mail et mot de passe
- Le mode "super utilisateur" est possible via les identifiants "android" et "android3"
- L'application se connecte aux automates

Demandes non respectées

- Le super utilisateur peut voir les identifiants de tout le monde, mais n'a pas de possibilités d'écriture
- Le choix des privilèges n'est pas accessible
- Le mot de passe "android3" n'est pas modifiable via l'application

Conclusion

L'application à réaliser n'était pas chose simple. Ne connaissant pas vraiment le langage Android et ayant du mal avec le langage Java, j'avais peur de ce qui pouvait m'attendre.

Cependant, étant donné l'enseignement que j'ai reçu cette année pour ce cours et les ressources quasi illimitées trouvables facilement, j'ai tout de même mené à bien ce travail (en majeure partie en tout cas). Cette application m'a montré à quel point Android est riche et personnalisable. Je pense que je n'ai vu que la surface de ce que propose le langage Android. Je tiens d'ailleurs à préciser qu'Android Studio est un langage très convenable est très bien conçu, et que je m'en servirais sans doute plus tard dans mon travail en tant que développeur.

Concernant l'application elle-même, elle sera parfaitement complète lorsque l'utilisateur pourra modifier les informations de l'automate directement via le smartphone. Et pourquoi pas, cela sera très utile de pouvoir mettre soi-même des éléments dans l'application et via l'application afin de pouvoir faire une application personnalisée sans devoir tout refaire au complet.
Je laisse donc un maximum d'indications dans le code de mon application, afin de pouvoir la modifier plus facilement plus tard (moi ou quelqu'un d'autre).

Logo de l'application : conçu par mes soins.