BOLLON Hugo DELAPIERRE Tristan

Rapport INFO706 : SkillsList

Sommaire

Introduction

SkillsList est une application de gestion de compétences pouvant servir de "portfolio". Les utilisateurs ont la possibilité de demander la validation de diverses compétences, auto-évaluables ou non nécessitant donc la validation d'un utilisateur en ayant les droits.

Pour cette première version, nous avons mis en avant son application au sein du domaine universitaire avec donc des utilisateurs étudiants et d'autres professeurs. La gestion des compétences (création, suppression, etc.) est disponible pour les professeurs.

Le système de droits implémenté permettrait aisément d'adapter cette application à un public plus général dans un possible avenir!

Tout le code source de ce projet est disponible sur github à cette adresse :
https://github.com/hbollon/SkillsList

Architecture

Ce projet nécessitait un système d'authentification, des interactions entre utilisateurs ainsi qu'un stockage de données partiellement partagé entre utilisateurs. Nous avons donc appliqué un modèle Client/Serveur pour la mise en place de cette application.

Coté serveur

Nous avons commencé par mettre la priorité sur notre backend, nous voulions un serveur facilement déployable, compatible avec tous les systèmes d'exploitation (notamment utilisable en ssh sur un VPS distant Linux sans GUI) et particulièrement robuste. Nous avons donc opté pour un serveur Java avec le framework Spring Boot, en effet, la jvm offrait une comptabilité accrue et Spring nous à permis de mettre en place assez facilement une API REST robuste pour toutes nos interactions avec notre application mobile.

Pour simplifier la gestion des dépendances ainsi que le déploiement du serveur, nous avons utilisé Maven comme gestionnaire de projet Java.

Enfin, le stockage des données se passe dans une base de donnée MySql. La communication et les interactions avec cette dernière sont faites à l'aide de JDBC (Java Database Connectivity). Dans le cadre de cette première version, nous avons hébergé le serveur ainsi que la base de donnée sur un VPS Ovh tournant sur Ubuntu Server 18.04 totalement sécurisé au préalable avec un pare-feu IpTables ainsi que d'outils comme Fail2Ban et RKHunter. Les requêtes se font donc en http sur le port 8080. En fonction de l'action désirée la méthode http utilisée varie parmi: GET (pour la récupération de donnée), POST (pour l'insertion de donnée ou pour déclencher des fonctionnalités coté serveur), PUT (pour la modification de donnée) et DELETE (pour la suppression de donnée).

Coté client

Au niveau de l'application mobile, nous avons voulu faire une application cross-plateform (Android/IOS) et utilisant des technologies modernes. C'est pour cela que nous nous sommes tournés vers Flutter. Ce Framework, bien que jeune, couvrait tout nos besoins d'un point de vue technique notamment grâce à Dart, son language de programmation qui disposait nativement de tout le nécessaire pour les interactions http, le support du Json, etc. De plus, Flutter permet de produire facilement une interface utilisateur ergonomique, moderne et intuitive grâce aux Widgets disponibles nativement ainsi qu'à toutes les ressources open-sources existantes pour ce framework!

Fonctionnalités

- Inscription/Authentification d'utilisateurs
- Système de rôles (Étudiant/Professeur) et de permissions (Validation de compétences et création/modification/suppression de ces dernières)
- Management des SkillBlocks par les professeurs (Compétence "mère" composé de compétences)
- Management des Skills (compétences associées à un SkillBlock) par les professeurs
- Système d'auto-validation pour les compétences (par défault une demande de compétence par un étudiant doit être validée par un professeur afin d'être acquise, le statut "auto-validate" est disponible lors de la création d'un skill. Il permets d'omettre la validation du professeur pour son acquisition)
- Système d'abonnement aux Skillblock par les étudiants: permet aux utilisateurs d'afficher le Skillblock dans leur profil et de pouvoir demander certaines compétences associés
- Possibilité pour les étudiants de demander la validation de certaines compétences qui, une fois validé, apparaissent comme acquise sur son profil.
- Liste des étudiants accessible par les utilisateurs professeurs, elle permet de visualiser les compétences en attente de validation pour chacun d'eux avec la possibilité des les accepter ou de les refuser.
- Barre de progression pour chaque Skillblock en fonction du pourcentage de completion de ce dernier.
- Page de profil avec des statistiques et des médailles (en fonction du nombre de skills acquis par rapport aux Skillblock auxquels l'utilisateur est abonné)

Installation

Pour faire fonctionner cette application vous devez mettre en place à la fois le serveur (son instance sur vps étant potentiellement down) et le client.

Serveur

1. Installez Java (jdk >= 11), Maven et MySql sur votre système, sur Linux exécutez:

```
sudo apt install openjdk-11-jdk mysql-server maven
```

2. Initialisez une base de donnée "skillslist" et un utilisateur associé (ce procédé pourra être simplifié et automatisé dans l'avenir) :

```
sudo mysql
mysql> CREATE DATABASE skillslist;
mysql> CREATE USER 'testuser'@'localhost' IDENTIFIED BY 'password';
mysql> GRANT ALL PRIVILEGES ON skillslist.* TO 'testuser'@'localhost';
mysql> FLUSH PRIVILEGES;
```

Si vous avez utilisé un autre user que celui de l'exemple vous devrez changer les credentials de connexion des constantes fournies au début du fichier `./server/database/DatabaseHandler.java`. Ce procédé pourra également être améliorer avec un fichier .env dans une future version.

3. Clonez le projet dans un répertoire de votre choix:

```
git clone https://github.com/hbollon/SkillsList.git
```

4. Lancez le serveur à l'aide du script de déploiement:

```
cd SkillsList
./skillslist_server.sh
```

5. Le serveur est maintenant up! Vous pouvez accéder aux logs dans le dossier `./server/outputs/`

Client

Option 1

Voici les instructions pour faire fonctionner notre application avec votre propre serveur que vous avez lancé précédemment.

1. Vous aurez besoin de Flutter installé sur votre pc: Instructions d'installation
2. Rendez vous dans le dossier du projet précédemment cloné pour le serveur et ouvrez le fichier `./lib/main.dart`

3. Remplacez l'ip définie ligne 11 par l'ip publique de la machine hébergeant le serveur.
4. Enfin, compilez l'application en apk: `flutter build apk` et installez là sur votre téléphone ou émulateur

Vous pouvez également lancer directement l'app sur votre téléphone ou émulateur en passant par adb. Il vous suffit d'avoir adb d'installé, de brancher votre téléphone (en mode debug usb) ou lancer votre émulateur et enfin de lancer la commande `flutter run` dans le répertoire du projet.

Option 2

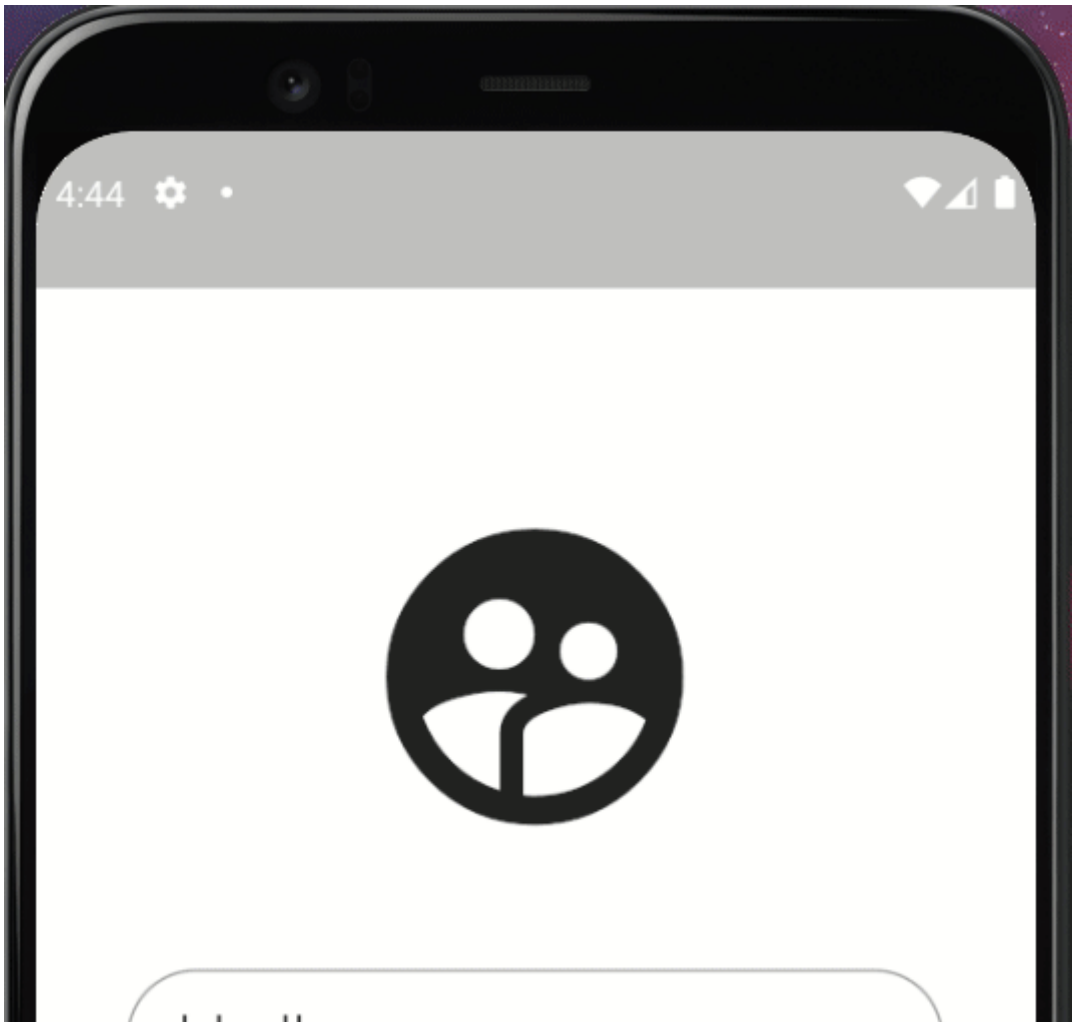
Avec cette méthode, vous aurez une version installable de notre application qui fonctionne directement avec le serveur hébergé sur mon VPS. Néanmoins, des travaux vont avoir lieu sur ce dernier, l'application ne fonctionnera donc peut-être pas.

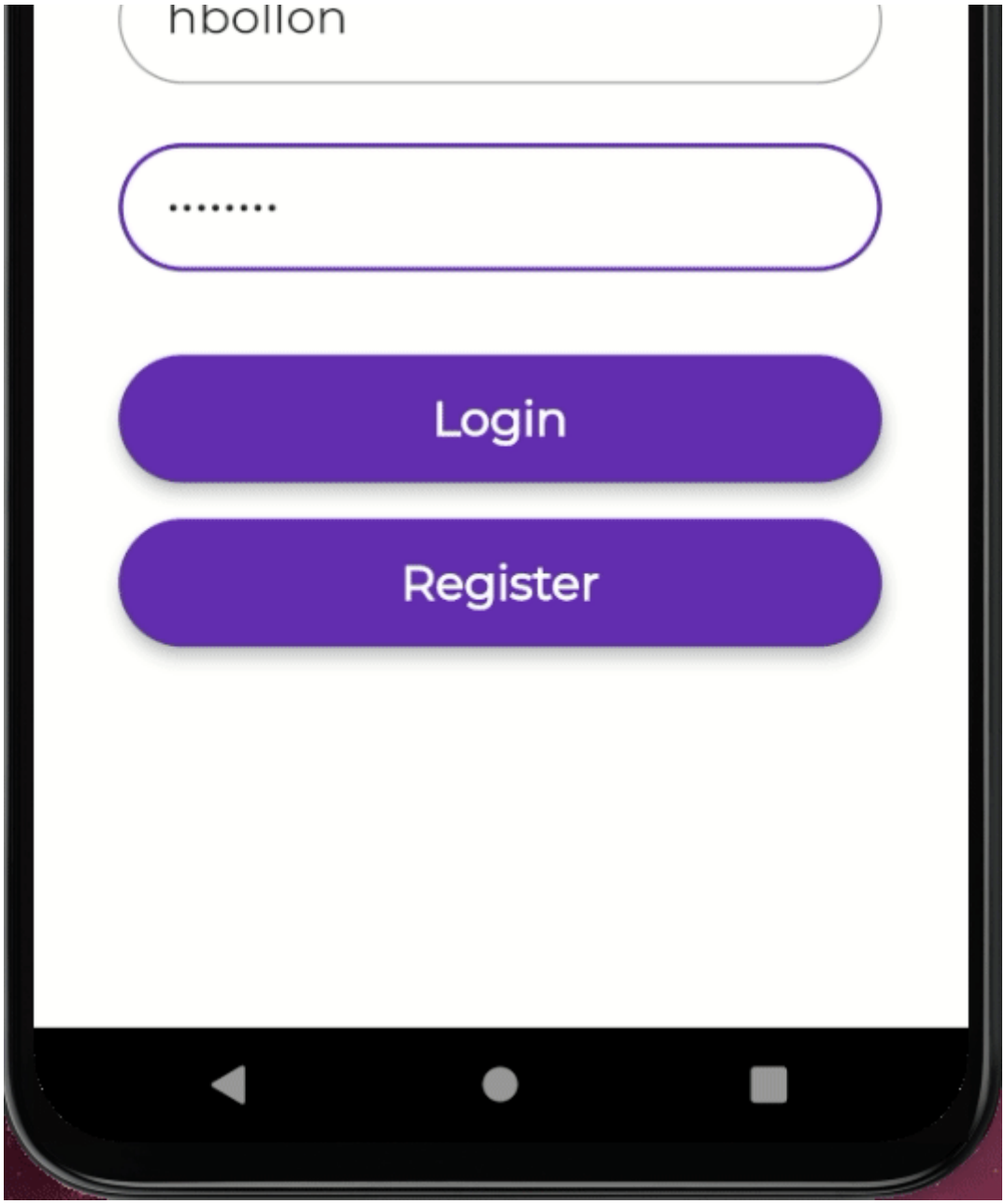
1. Rendez vous sur la page releases du repository github et téléchargez l'apk **skillslist-1.0.1.apk** situé dans la partie assets de la release **v1.0.1 APK**
2. Installez l'apk sur votre appareil

Assistance

En cas de difficultés pour deployer le client ou le serveur, ou bien si rencontrez tout autre soucis avec le projet n'hésitez pas à ouvrir une issue sur le repository Github ou à nous contacter (voir rubrique **Auteurs**).

Aperçu





Amélioration

Le projet n'en est qu'à sa première version. Le serveur à atteint le niveau que nous attendions, néanmoins, l'application aurait pu être plus développer visuellement parlant et certaines fonctionnalités manquent. Malheureusement, le front-end à été commencé trop tard.

Pour les prochaines versions il y a bon nombre de points améliorables comme:

- Au niveau de la configuration du projet, il faudrait ajouté un fichier .env à la racine pour facilité les réglages de la base de donnée et de l'ip du serveur sans avoir à toucher le code.
- L'ajout d'un système plus poussé de succès et de statistiques
- Rendre possible la visualisation de son profil même en hors ligne (à l'heure actuelle l'application ne dispose pas de fonctionnalités hors-ligne)
- Ajouter une dimension sociale (que les utilisateurs puissent consulter les profils d'autres personnes)
- Implémenter un système de notifications push
- Ajouter une fonction de "mot de passe oublié" sur l'écran de connexion
- Améliorer la charte graphique de l'application et son design

- Et bien plus encore!

Retour sur la matière

Tout d'abord, le concept de ces "mini-projets" est, de notre point de vue, l'une des meilleures façon de mener un TP dans le cadre universitaire. En effet, cela nous permet de mettre laisser libre cour à notre imagination et de nous apprendre l'autonomie et la résolution de problématiques.

Le sujet du TP était suffisamment large pour nous permettre d'ajouter une part d'originalité mais sans pour autant de partir dans tout les sens. Le coté Client/Serveur était lui aussi très intéressant!

Enfin, merci M.Carron pour votre flexibilité et votre disponibilité durant ce projet notamment pour les rendus!

Auteurs

👤 **Hugo Bollon**:

- Email: hugo.bollon@gmail.com
- Github: @hbollon
- LinkedIn: @Hugo Bollon
- Portfolio: hugobollon.me

👤 **Tristan Delapierre**

🤝 Contribuer

Ce projet est open-source et hébergé sur Github! (le projet Java pour la partie serveur est dans le sous-dossier "server"). Toutes les contributions sont les bienvenues!

N'hésitez pas à ⭐ si ce projet vous à plu!

📝 Licence