

Département de génie informatique et de génie logiciel

**INF8405 - Informatique Mobile**

**Projet final : Application sociale pour Android**

**Sacha Licatese-Roussel, 1635849**

**Yann Dago, 1635682**

**Golnoush Rahimzadeh, 1759612**

**Travail soumis à Aurel Josias Randolph**

**Remis en date du**

**18 avril 2016**

# **Sommaire**

[Sommaire 2](#_Toc446438668)

[Introduction 3](#_Toc446438669)

[Présentation technique du travail 3](#_Toc446438670)

[Aspect de présentation de l’application 4](#_Toc446438671)

[Gestion de la base de données d’utilisateurs 5](#_Toc446438672)

[Proposition de lieux et dates 6](#_Toc446438673)

[Affichage des positions des membres sur une carte 6](#_Toc446438674)

[Difficultés rencontrées 6](#_Toc446438675)

[Critiques et suggestions 7](#_Toc446438676)

[Conclusion 7](#_Toc446438677)

# Introduction

**Dans le cadre de ce projet, soit une application sociale pour Android, il a fallu réaliser une application permettant aux utilisateurs d’avoir une liste de séries préférées, ajouter et supprimer des amis, avoir la capacité de leur recommander des séries et aussi voir les personnes dans un rayon de 100m utilisant l’application ainsi que les séries que ceux-ci regardent et celles que vous avez en commun. L’utilisation d’une base de données de série, d’une base de données pour les données utilisateurs, de *Google Maps* et de capteurs du téléphone ont servis à fournir une expérience utilisateur particulièrement intéressante et innovatrice.**

# Présentation technique du travail

Plusieurs contraintes techniques ont été imposées pour ce projet, ainsi pour chacune d’entre elles, les solutions choisies sont les suivantes :

* Connexion d’équipements mobiles par réseaux sans-fil :
  + L’application utilise la connectivité Wi-Fi ainsi que le réseau de données du téléphone pour accéder à internet.
* Utilisation d’au moins deux capteurs :
  + L’application utilise le GPS intégré afin de détecter la position de l’utilisateur et d’utiliser la carte pour afficher les utilisateurs à proximité
  + L’application utilise l’accéléromètre afin de détecter que le téléphone est secoué

Pour ensuite proposer à l’utilisateur le choix d’ajouter une nouvelle série qui pourrait l’intéresser.

* + L’application utilise le gyromètre pour détecter les changements d’orientation du téléphone afin d’automatiquement changer de vue, vers la carte de *Google Maps* lorsque l’orientation est en mode paysage.
* Application de géolocalisation et utilisation de carte interactive :
  + L’application utilise *Google Maps* afin d’afficher les positions de l’utilisateur et des utilisateurs aux alentours.
  + L’application utilise la géolocalisation de Google afin d’avoir des coordonnées de localisation précises de l’appareil en cours d’utilisation.
  + L’application utilise *GeoFire* de *Firebase* afin de faire la recherche des personnes autour d’une certaine coordonnée de localisation.
* Stockage de données :
  + Le stockage de données est fait grâce à Firebase qui fournit une base de données de style NoSQL ainsi qu’une fonctionnalité d’authentification d’utilisateurs très bien conçue.

## Aspect de présentation de l’application

L’application présente plusieurs pages qui sont les suivantes :

|  |  |  |
| --- | --- | --- |
| Figure 1 : Page principale | Figure 2 : Page de complétion de compte | C:\Users\yannd\AppData\Local\Microsoft\Windows\INetCache\Content.Word\12980968_10209095463729870_1970054986_o.jpg  Figure 3 : Menu principal |
| C:\Users\yannd\AppData\Local\Microsoft\Windows\INetCache\Content.Word\13010095_10209095462689844_2075926157_o.jpg  Figure 4 : Page de gestion des amis | C:\Users\yannd\AppData\Local\Microsoft\Windows\INetCache\Content.Word\12980822_10209095463929875_1993954963_o.jpg  Figure 5 : Page de gestion des séries | C:\Users\yannd\AppData\Local\Microsoft\Windows\INetCache\Content.Word\12970522_10209095462849848_1362419542_o.jpg  Figure 6 : Page de gestion des suggestions de séries reçues |
| C:\Users\yannd\AppData\Local\Microsoft\Windows\INetCache\Content.Word\13010200_10209095463409862_2038170848_o.jpg  Figure 7 : Page de de visualisation des utilisateurs proches | C:\Users\yannd\AppData\Local\Microsoft\Windows\INetCache\Content.Word\12999722_10209095463209857_718760108_o.jpg  Figure 8 : Page de de visualisation des séries d’un utilisateur proche | C:\Users\yannd\AppData\Local\Microsoft\Windows\INetCache\Content.Word\12959368_10209095463489864_1866256577_o.jpg  Figure 9 : Page de de changement de paramètres |

* La page principale :

C’est la page d’entrée à l’application, elle permet de se connecter ou de créer un compte. Après avoir entré son courriel, si le compte n’existe pas, il est créé puis l’utilisateur est amené à la page de complétion de compte pour choisir un nom d’utilisateur.

Des rétroactions sont faites à l’utilisateur pour l’informer que ses informations sont valides ou pas.

* Le menu principal :

Ce menu permet de naviguer à travers toutes les pages de l’application. Il apparait en glissant son doigt à partir du côté gauche du téléphone vers la droite.

* La page de gestion d’amis :

Cette page permet à l’utilisateur de voir sa liste d’amis, la liste de requêtes d’amis reçues. Il peut voir les informations lui permettant d’identifier ces personnes telles que leur nom et leur photo de profil. Il a la possibilité de supprimer des amis, accepter ou refuser des requêtes. Il a aussi la possibilité de faire une recherche d’amis, grâce à un champ doté d’auto complétion pour les noms d’utilisateur disponibles.

* Page de gestion des séries :

Dans cette page, l’utilisateur peut voir les séries qu’il suit, et peut faire une recherche de série, qu’il peut ensuite ajouter dans sa liste de séries. Recommander des séries de sa liste à des amis et supprimer des séries.

* Page de gestion des suggestions de séries reçues :

L’utilisateur voie les suggestions reçues et peut soit suivre ces séries ou les refuser.

* Page de visualisation des utilisateurs proches :

L’utilisateur peut voir les autres utilisateurs proches qui apparaissent en rouge s’ils n’ont aucune série en commun, en vert s’ils en ont au moins une en commun et en bleu s’il s’agit de l’utilisateur même.

* Page de visualisation des séries d’utilisateur proche :

Après avoir cliqué sur un utilisateur de la carte, une fenêtre est ouverte montrant les séries qu’il regarde, et les séries communes seront surlignées en vert.

* Page de changement de paramètres :

L’utilisateur peut décider d’être visible sur la carte pour les autres ou non, changer la couleur du fond d’écran et changer sa photo de profil.

## Gestion de la base de données d’utilisateurs

Le service ***Firebase* a été utilisé afin de de gérer la banque de groupes et d’utilisateurs, ainsi la synchronisation entre le téléphone et la base de données est gérée par ce cadriciel. L’authentification de l’utilisateur faite à l’aide d’une adresse courriel et un mot de passe est aussi gérée par ce cadriciel. Ainsi, dans la base de données est sauvegardée une table de noms d’utilisateurs dont la clé est un ID propre à chaque utilisateur et la valeur est le nom d’utilisateur. Evidemment, des vérifications sont faites pour éviter des noms d’utilisateurs dupliqués. Puis il y a une table d’utilisateurs, qui contient pour chaque utilisateur ses données, soit sa liste d’amis, de séries, ses préférences, les suggestions de séries reçues, et les demandes d’amitiés reçues. Afin de faciliter la recherche d’utilisateurs aux alentours, *Geofire* a été utilisé, ce cadriciel quant à lui, sauvegarde les positions des utilisateurs dans une table à part avec le nom d’utilisateur pour clé.**

## Gestion des amitiés

Chaque profil utilisateur contient à liste d’amitiés et de demandes reçues ; ainsi pour envoyer une requête d’amitié, le nom de l’utilisateur demandant est inscrit dans liste de requêtes reçue du destinataire de la demande. Pour accepter une amitié, il suffit de mettre les noms des utilisateurs en questions dans leurs listes d’amis respectives. Pour retirer un ami, il suffit d’enlever les noms des deux utilisateurs concernés de leurs listes d’amis respectives.

## Gestion des séries

Les séries sont retrouvées sur OMDB, une base de donnée Open-Source qui contient toutes les séries contenues sur IMDB, leurs ID sur le site IMDB et plusieurs informations importantes telles une le nom de la série, une photo du poster, une description, etc. Lorsque l’utilisateur fait une recherche, une requête HTTP est faite sur le serveur d’OMDB afin de retrouver une série qui contient l’objet de recherche, puis une liste de séries possible est retournée au format JSON. Seule l’ID des séries est sauvegardé sur la base de données *Firebase*, et les données des séries sont sauvegardées localement tout au long de l’utilisation de l’application, ainsi à chaque connexion ou chaque fois que les données d’une série spécifique sont requises, une requête HTTP est faite au serveur afin de récupérer les données, puis une autre est faite afin de récupérer le poster de la série, au besoin.

Chaque profil utilisateur contient à liste de séries et de suggestions de séries reçues ; ainsi pour suivre une série, il suffit d’ajouter l’ID de la série dans la liste de séries. Pour envoyer une recommandation, il suffit d’ajouter l’ID de la série dans la liste de recommandations reçues du destinataire. Pour suivre une recommandation reçue, il suffit de retirer son ID des suggestions reçues et l’ajouter dans la liste de séries suivies. Pour arrêter de suivre une série, il suffit de retirer son ID de la liste de séries.

## Affichage des positions des utilisateurs sur une carte

Afin d’afficher la position des membres sur une carte, *Google Maps* a été utilisé. En utilisant les services de localisation du téléphone, la position du membre est sauvegardée dans son profile sur la base de données *Firebase* grâce à *Geofire*. L’utilisateur a le choix de partager ou non sa position ; s’il refuse, il n’aura pas accès à cette fonctionnalité et ne verra qu’une carte vide sinon il apparaitra avec un marqueur bleu sur la carte, tandis que les autres utilisateurs auront des marqueurs rouges, s’ils n’ont aucune série en commun avec l’utilisateur courant et des marqueurs verts s’ils en ont au moins une. Suite à un clic sur l’un des marqueurs rouge ou verts, la liste des séries de l’utilisateur sélectionné est affichée, avec les séries en commun mises en évidence.

## Utilisation des capteurs

Pour ce qui est de l’utilisateur de l’accéléromètre, nous l’avons utilisé afin de détecter lorsque l’utilisateur secoue son appareil de manière frénétique durant deux ou trois secondes environs. Ceci nous permet donc de répondre à cet évènement de manière à aller sélectionner aléatoirement, parmi les séries d’un des amis de l’utilisateur, une série qui est inconnue à l’utilisateur courant et de lui proposer de l’ajouter à sa bibliothèque, lui permettant ainsi de commencer à suivre cette nouvelle série télévisée. Évidemment, l’utilisateur doit avoir des amis dans son profil afin d’avoir accès à cette fonctionnalité.

Pour ce qui est du second senseur, nous avons utilisé la détection de l’orientation de l’appareil (soit portrait ou bien paysage) afin de modifier la vue active de l’application, dans le but de rendre l’expérience utilisateur la plus interactive possible. En effet, lorsque l’appareil est positionné en mode portrait, l’ensemble des fonctionnalités sont offertes via le menu principal. Cependant, lorsque le téléphone ou la tablette est retourné en mode paysage, on change de vue afin d’afficher automatiquement la carte des utilisateurs à proximité et ce, en plein écran afin d’offrir une expérience immersive. Pour le type d’application que l’on souhaitait développé, nous considérons ce type de comportement comme des éléments intéressant à intégrer au niveau des senseurs présents sur le téléphone.

## Affichage de l’utilisation de la batterie

Passons maintenant à la partie de l’évaluation et de l’affichage de l’utilisation de la batterie. Pour accomplir cette tâche, nous nous sommes défini une classe *Singleton* nommée *RessourceManager*. Celle-ci possède donc un ensemble restreint de méthodes permettant, entre autre, de noter la valeur actuelle du niveau de la batterie et de calculer le delta dans un second appel de méthode. Ceci est donc utilisé à divers endroits dans l’application afin de déterminer la quantité d’énergie utilisé par l’appareil pour la réalisation de certaines tâches réalisées par notre application. Par exemple, si on ouvre la carte, on débute une lecture et lorsque l’on revient à l’activité principale, l’utilisation de batterie durant l’utilisation de la carte est affichée à l’écran durant un court instant. Le même principe est utilisé pour les tâches comme la recherche de série sur OMDB, la connexion à l’application, l’ajout d’ami(s), etc. Finalement, lorsqu’on ferme l’application en entier, nous avons une dernière information qui s’affiche à l’écran, il s’agit de l’utilisation globale de la batterie durant la dernière session d’utilisation de l’application en entier. Ces différentes mesures permettent donc d’obtenir une meilleure idée de la consommation de la batterie aux différentes stages d’utilisation de l’application ainsi que d’un point de vue un peu plus global.

# Difficultés rencontrées

Les principales difficultés furent au niveau de l’architecture logicielle, puisqu’il fallait utiliser plus qu’un cadriciel et que ceux-ci devait communiquer ensemble de manière efficace et transparente. Une autre difficulté aura été la gestion de la mémoire dans les différentes vues de notre application, puisqu’aller chercher toutes les données d’un utilisateur peut être lourd du point de vue des communication réseau, on souhaite donc optimiser le plus possible ces appels et utiliser des composantes visuelles réutilisables et recyclables. Un autre problème a été de choisir les bonnes vues pour afficher les données ; il y a eu des problèmes avec l’utilisation de *ListView*, puisque celles-ci se redessinait au complet au moindre changement, donc avec des requêtes asynchrones sur un grand volume de données, il y avait trop de rafraichissements inutiles et cela entrainait une trop grande utilisation de la mémoire et du processeur de l’appareil, l’utilisation de *RecyclerView* a donc été grandement bénéfique puisqu’elle remédiait à ce problème avec brio. Un autre problème était la synchronisation des requêtes asynchrones sur un même modèle de données, par exemple, pour une série, il fallait chercher de façon asynchrone ses données, puis à partir de ses données, chercher une photo. Ici, l’architecture a joué un grand rôle afin d’offrir une expérience utilisateur agréable, puisque les données sont affichées au fur et à mesure qu’elles sont reçues pour ne pas que l’utilisateur n’attende trop lorsqu’il y a un grand volume de données et pour évite une consommation de données trop élevé.

# Critiques et suggestions

Pour ce qui est des critiques et suggestions de ce dernier travail, nous avouons que celles-ci sont plutôt restreintes à quelques éléments uniquement. En effet, puisque les connaissances acquises lors des deux premiers travaux pratiques nous auront permis de nous équiper afin de réaliser ce travail final de manière beaucoup efficace.

Tout d’abord, la critique principale s’attaque au fait de devoir absolument incorporer un nombre précis de senseurs et éléments de *back-end* à notre application. Selon nous, une plus grande flexibilité au niveau du choix des composantes à intégrer serait intéressante, puisque ceci offrirait probablement une plus grande diversité au niveau des applications produites par les différentes équipes. Et donc, ceci rendrait les multiples présentations orales probablement beaucoup plus diversifiées et intéressantes, par-rapport aux différents aspects techniques choisis par les équipes, plutôt que de limiter le choix à quelques senseurs et une carte *Google Maps*.

De plus, il aurait été intéressant selon nous, de faire de ce projet un projet sur toute la session avec plusieurs petites remises en guise de travaux pratiques, afin qu’il soit au final plus étoffé en fonctionnalités et nous permettant ainsi d’améliorer le produit final tout au long de nos apprentissages.

# Conclusion

En conclusion, ce travail pratique a permis de mettre en commun des apprentissages faits aux derniers travaux pratique tout en permettant d’ajouter une touche de créativité et un défi supplémentaire quant aux fonctionnalités afin d’offrir un produit final satisfaisant qui respecte les contraintes tout en ayant sa touche d’originalité. Nous sommes très satisfaits du résultat obtenu et attendons impatiemment notre date de présentation orale afin de pouvoir vous partager cette application captivante.