Rapport projet

# Description de l’application :

Notre application sert de moyen pour les gens de se rencontrer pour former des amitiés ou même des relations amoureuses.

Pour commencer, nous supportons un système d’authentification/login. À chaque fois que l’on ouvre l’application, l’utilisateur est demande soit de se créer un compte, ou de se connecter à un compte existant avec email et password.

Le compte utilisateur contient les données récolté lors de la création du compte, soit un nom, une photo, une description et le sexe.

L’application collecte automatiquement la position géographique et la température ambiante.

Lorsque l’utilisateur est en ligne, on ouvre une carte ou tous les utilisateurs sont affichés. Si l’utilisateur appuie sur un marcheur, ses informations seront affichées. Cela inclus les informations du compte mentionné plus haut en plus de la température du cellulaire. Lorsque l’on appuie sur cette fenêtre. Cela ouvre la fenêtre de chat. On ne peut pas ouvrir un chat en cliquant sur la fenêtre de sont propre compte.

Une fois la fenêtre de chat ouverte, on peut envoyer des messages à l’autre en écrivant le message dans la boite de texte puis en appuyant sur le bouton Envoyer. L’application peut aussi faire des suggestions de messages à envoyer pour aider à commencer une conversation, ou pour rire. Lorsque l’on appuie sur le bouton Suggestion, une phrase au hasard apparaitra dans la boite de texte. Enfin, le dernier bouton dans la fenêtre de chat est le bouton Trouver. Si l’on appuie ce bouton, l’application retourne à la carte et affiche le chemin pour se rendre à la position de la personne à partir de se propre position.

# Critères du projet

Voici comment nous avons répondu aux critères du projet :

1. Connexion d'équipements mobiles par réseaux sans fils (Wi-Fi, Wi-Fi P2P, Bluetooth, …) ou partage de données par NFC (transfert de fichier avec l'API Android Beam) ou autre technologie
   1. Partage de donné avec firebase
   2. Connexion à l’api de Google Maps pour trouver le chemin entre deux points.
2. Utilisation d’au moins deux capteurs (accéléromètre, gyromètre, champ magnétique, température, pression, orientation, proximité, lumière, caméra …)
   1. Caméras pour la photo du compte usagé
   2. Capteur de température ambiante pour les suggestions de messages.
3. Application de géolocalisation et utilisation de carte interactive (Google Maps API). Vous pouvez de plus vous appuyer sur d’autres services Google comme l’API Places.
   1. Carte qui affiche les utilisateur et leurs positions.
4. Stockage de données grâce à une base de données SQLite et utilisation de bases de données ou de services (par exemple de traitement) en ligne (Firebase, Cloud, …)
   1. Firebase
5. Utilisation des informations sur la batterie, notamment l’évolution du niveau de celle-ci pendant l’utilisation de l’application. (exemple : calculer et afficher la consommation d’énergie depuis le lancement de l’application)
   1. On diminue la fréquence d’envoi des données lorsque la batterie est faible.

# Décision techniques

## Packages

* Activities : Contient les différentes fenêtres inclue dans l’appliaciton (fenêtre de login, de carte, etc.)
* fireBaseHelper : Contient les classes faisant la connection à la base de données firebase. (UserDbHelper, ChatDbHelper)
* Model : Contient les classe qui représente les structures de données. (User, Chat, etc.)
* Services : Contient les classe gérant les services.
* Utils : Contient des utilitaires qui ne retraient dans aucune des catégories précédentes (ChatUtils, Path, UserInfoWindow, etc.)

## Base de données

Nous utilisons la base de données Firebase et son système NoSQL. Notre structure de données est divisé en deux catégories : la liste des users et la liste des conversation.

Les users ont comme données la température, leur coordonées, leur description, leur sexe et leur nom. Leur id est leur id d’identification. Celui-ci est lié à un email et un mot de passe à l’aide de l’authentification de firebase.

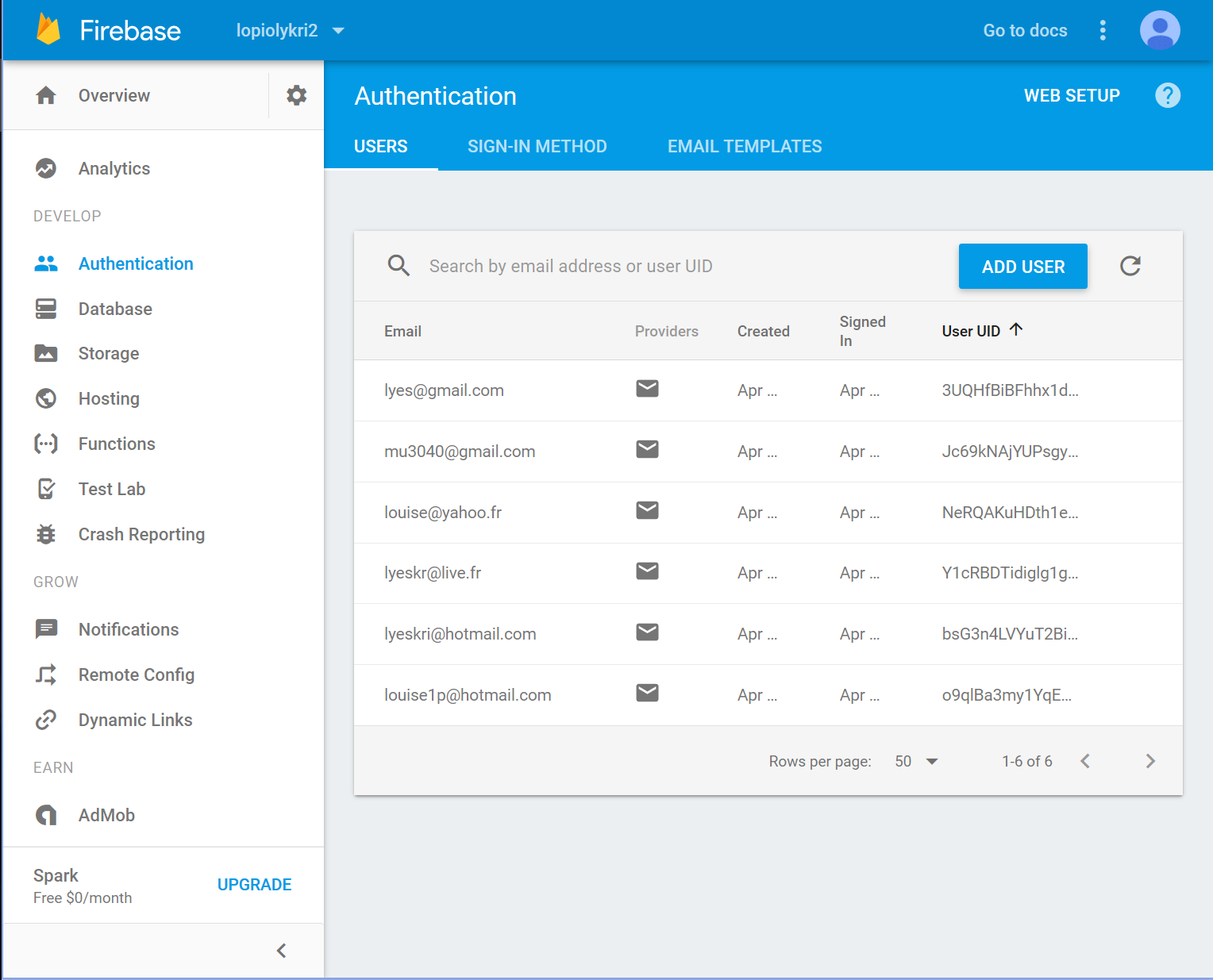


Figure 1 : Authentification user firebase

## Markers

Comme au TP2, les markers ne permettent plusieurs configurations différentes. Pour attaquer ce problème, nous avons utilisé une stratégie différente que celle du TP2.

Dans le TP2, nous avions mis de l’information à parser dans le snipet pour savoir quoi afficher. Dans ce projet par contre, nous avons simplement envoyer le ID du user dans le titre. À partir du ID, on trouve le user et on change l’affichage selon qui il est. (On n’affiche pas le bouton qui ouvre le chat dans la fenêtre d’information de l’utilisateur authentifié (curent user) Cette stratégie marche car tous les structure de données sont identique et à la même place, alors que dans TP2, nous avions trois structure de données différentes (utilisateur, lieu et évènement).

La même stratégie est utiliser dans le onClickListener. Ce listener est appliquer à tous les markers, alors on bloque la fonctionnalité de chat si l’utilisateur est celui login sur le cellulaire. Nous allons chercher l’information de la même façon car nous avons accès au marker (et le id dans sont titre) dans le listener.

# Difficultés

Comme nous étions plus à l’aise avec le développement Android, nous avons eu moins de difficulté lors du projet. Les seules vraies difficultés que nous avons rencontré sont :

* L’authentification sur Firebase
* Diviser les conversation selon les users