

**IFT 717 - Applications Internet & Mobilité**  
**Projet de session partie 2 : Rapport de projet.**

Colin Troisemaine (matricule 20 088 209)  
Guillaume Bouchard (matricule 20 100 748)  
Alexandre Turpin (matricule 20 088 156)  
Quentin Levieux (matricule 20 102 087)  
Adrien Verdier (matricule 20 088 959)

# Sommaire

<b>Introduction</b>	<b>3</b>
<b>Architecture générale du projet</b>	<b>4</b>
<b>Le serveur backend Express</b>	<b>5</b>
<b>Le site web React</b>	<b>5</b>
Les fonctionnalités principales	5
Le système de connexion avec OAuth	6
Les différentes technologies utilisées	8
SweetAlert	8
Spotify & Geonames	9
Google Calendar	9
<b>L'application Android</b>	<b>10</b>
Structure de l'application	10
Login	12
Konote	13
Kourse	14
Kotemps	15
Kochat	16
Kognotte	17
Kusique	18
Koulette	19

# 1. Introduction

Dans le cadre du cours d'IFT 717 - Application Internet et Mobilités, nous avons effectué un projet de session visant à mettre en œuvre les différentes notions vues en cours en mettant en place une application mobile et web fonctionnelle.

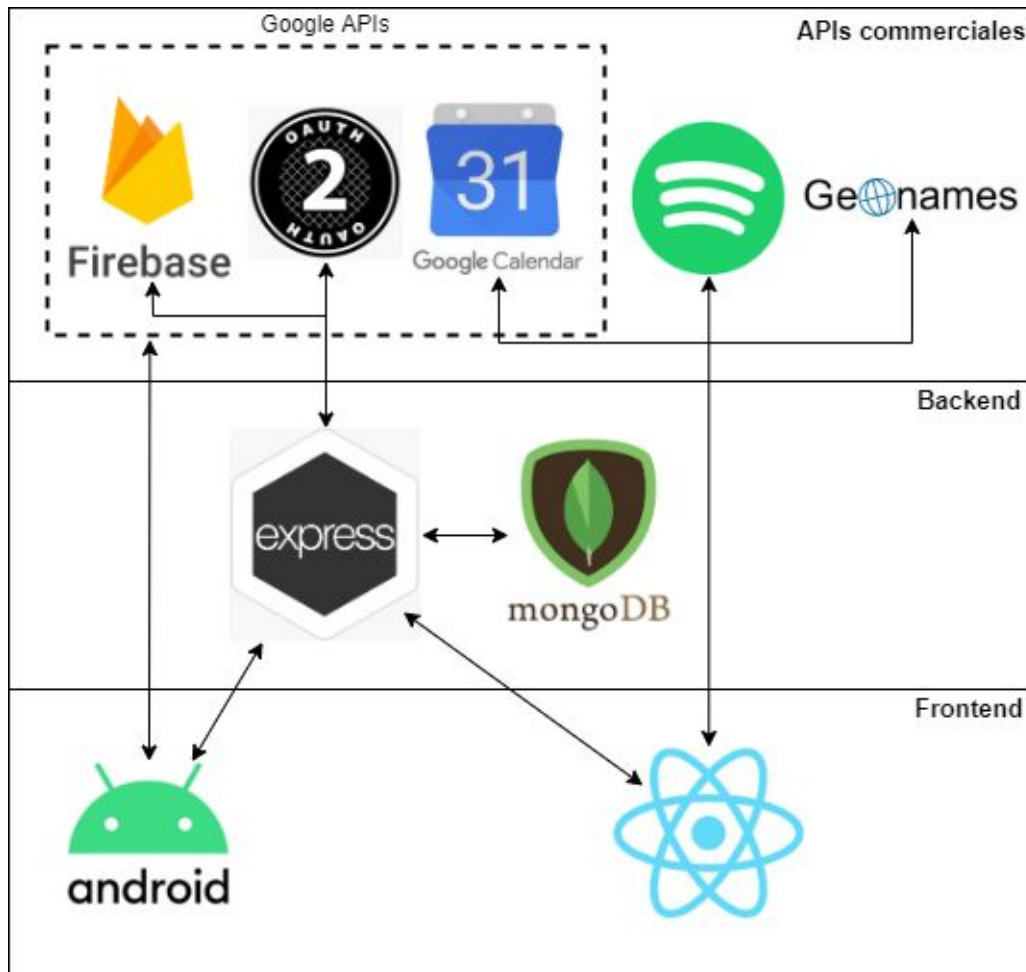
Ayant pour objectif de nous installer en colocation tous les 5, nous avons vu dans ce projet de session l'opportunité de mettre en place une application pour gérer la colocation de la meilleure manière possible. Suite à une recherche sur le web ainsi que sur les différents stores mobiles, nous n'avons pas trouvé de produit équivalent à notre idée, nous avons donc trouvé particulièrement intéressant de développer ce produit.

L'objectif est donc de créer un "Dashboard" partagé qui va permettre de faciliter les tâches communes et la communication au sein d'une colocation. Il a fallu mettre en place un serveur REST, une application web ainsi qu'une application mobile Android. Cette architecture doit permettre aux membres du groupe de se connecter simplement à l'application et de pouvoir accéder efficacement à toutes les fonctionnalités nécessaires à la vie en communauté. Plus précisément : avoir accès à un espace permettant d'écrire des notes lisible par l'ensemble des membres, un espace pour gérer les différentes listes de courses, un calendrier personnel où on va pouvoir ajouter des événements pour les membres du groupe, un chat partagé entre tous les membres, une cagnotte commune où l'on va pouvoir suivre les comptes de tous les membres, une roulette permettant de décider qui devra effectuer les différentes tâches ménagères et enfin un lecteur de musique pour que tout le monde puisse écouter de la musique via l'application.

Dans ce rapport, nous allons donc avoir l'opportunité de vous présenter notre application : KoBoard !

## 2. Architecture générale du projet

Dans le premier rapport de ce projet, nous avons réalisé une ébauche de l'architecture que notre projet adopterait. Au cours de la réalisation de celui-ci, cette architecture a naturellement connu des modifications et des enrichissements :



*Schématisation de l'architecture du projet*

Si on la compare à notre première version, on remarque que deux nouvelles APIs commerciales ont été rajoutées : Geonames et Google Calendar. En effet, comme nous l'expliquerons plus en détail par la suite, Geonames est utilisé en parallèle avec Spotify afin de déterminer la meilleure playlist à proposer à l'utilisateur en fonction de sa localisation.

Nous avons également pu préciser le type de base de données nous utiliserons : Notre choix s'est tourné vers une base MongoDB que hébergeons actuellement sur un VPS (Virtual Private Server) afin que les données et les schémas soient partagés entre tous les développeurs, ce qui a grandement facilité le développement de notre projet.

### 3. Le serveur backend Express

L'élément le plus central de notre projet est certainement notre serveur backend Express. Il permet la liaison entre les données de la base MongoDB et l'application Android et le serveur web React.

```
├─ serveur_express          <- Le serveur express backend (notre API)
│   └─ README.md
│   └─ public
│   └─ src
│       └─ controllers      <- Implémentation de la logique des routes
│       └─ models           <- Modèle des objets de la BDD
│       └─ routes           <- Définition des routes
│       └─ services         <- Méthodes de gestion de la BDD
│       └─ utils            <- Fichiers de configuration et méthodes utilitaires
│       └─ app.js
```

*Structure du serveur express*

Voici ci-dessus la structure du serveur express backend. Nous avons notamment fait attention à bien séparer la définition des routes et leur implémentation logique dans deux sous-dossiers différents afin de faciliter la lecture du code. On peut également noter que nous avons fait une grande utilisation des promesses Javascript au sein du code car la plupart de nos méthodes sont asynchrones (accès à la BDD et communication avec les APIs commerciales). Cela nous a permis d'écrire du code plus propre et d'éviter des erreurs d'implémentation liées aux callbacks, tout en gérant mieux les exceptions.

### 4. Le site web React

#### 4.1. Les fonctionnalités principales

- **Accueil :**

C'est sur cette page que nous arrivons une fois que nous sommes connectés à l'application. On peut y retrouver le lecteur web spotify intégré dont nous parlerons plus en détail par la suite.

- **Konotes :**

Cette page contient notre fonctionnalité de partage de notes. Elle permet de transmettre des messages à travers la colocation de manière asynchrone. Similairement à l'application Android, écrire un caractère "@" lancera le système de tag et proposera une complétion automatique à partir des noms des membres enregistrés de l'application.

- **Kourses :**

Cette page contient les listes de courses partagées de notre colocation. En effet, les courses sont un élément important d'une colocation et cette fonctionnalité permet à tout le monde de la remplir selon ses besoins et de suivre facilement les achats réalisés.

- **Kotemps :**

Cette page contient un Google agenda avec les emplois du temps de la personne authentifiée. Les événements seront ajoutés au calendrier "principal" de l'utilisateur connecté. Cette page permet donc d'ajouter, modifier, supprimer et déplacer les différents événements du calendrier.

- **Kochat :**

Cette page contient une application de chat synchrone via socket.io. Cela permet aux membres de notre colocation de communiquer instantanément sans pour autant passer par des applications externes, comme Whatsapp ou Messenger.

- **Kognotte :**

Un autre aspect important d'une colocation est la gestion du budget. Il existe de nombreuses applications offrant ces fonctionnalités mais le but de notre application est de réunir toutes ces fonctionnalités en un unique endroit. Nous avons donc développé Kognotte, notre propre système de gestion de budget pour les membres d'une colocation.

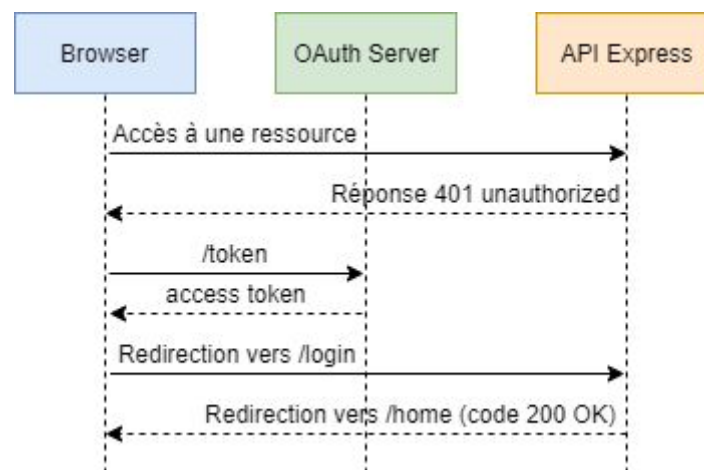
- **Koulette :**

La koulette est une fonctionnalité que nous avons imaginée pendant le développement de notre site. Cette roulette permet de déterminer d'une façon ludique qui, par exemple, fera les tâches ménagères de la semaine.

Toutes ces fonctionnalités sont disponibles sur notre site React de façon totalement responsive, ce qui permet à nos utilisateurs d'y accéder depuis n'importe quel appareil.

## 4.2. Le système de connexion avec OAuth

Une partie qui a été particulièrement technique à mettre en place est l'authentification avec OAuth. Le cheminement de la connexion que nous avons finalement mis en place est le suivant :



*Cheminement simplifié de la connexion*

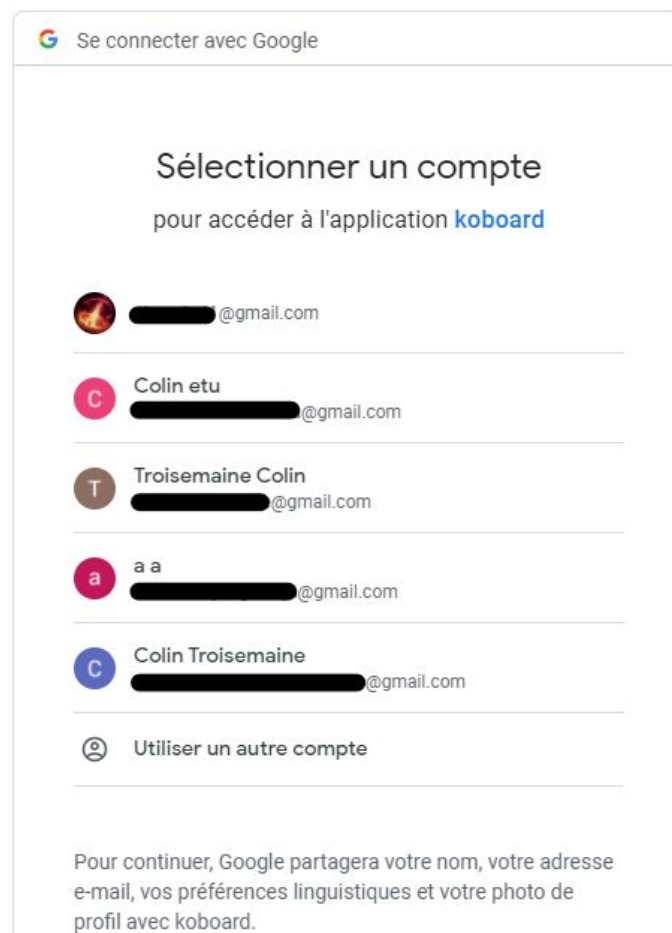
À chaque fois qu'une requête est envoyée à notre API Express, un middleware que nous avons développé vérifie que la requête possède des identifiants de connexion corrects

avant de traiter ou non la requête. On vérifie ainsi si le header possède un *Bearer Token* qui n'a pas encore expiré en appelant l'API OAuth2 de Google. S'il n'a pas expiré, on transmet la requête à son controller pour qu'elle soit traitée, mais si ce n'est pas le cas, on génère une nouvelle URL de connexion Google et on la renvoie au serveur Web React :



*La page de connexion*

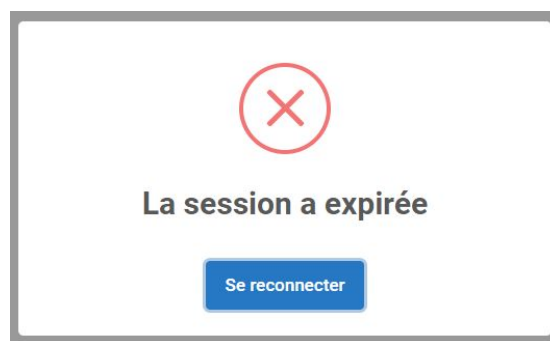
Si le serveur Web React reçoit un code d'erreur 401, il va nous rediriger vers la page de connexion ci-dessus. Le bouton "Se connecter avec Google" nous redirige donc vers le lien que l'API Express aura envoyée en même temps que le code d'erreur.



*La page de redirection Google*

Ce bouton nous renvoie donc vers la page de connexion ci-dessus où l'on nous invite à sélectionner le compte Google avant de poursuivre. Lorsque c'est fait, Google nous redirige vers l'URL que nous lui avons spécifiée. Il s'agit cette fois-ci d'une redirection vers l'API Express directement par la route <http://localhost:5000/login> où notre contrôleur de routes va récupérer le token que OAuth vient de générer. Notre API Express va alors récupérer l'email de l'utilisateur et mettre à jour (ou créer s'il n'existait pas encore) l'utilisateur correspondant dans la base de données, avant de rediriger le navigateur Web de l'utilisateur vers la page d'accueil du serveur Web React tout en lui transmettant son token.

Si au cours de l'utilisation du site web, notre token expire, le middleware de notre API Express va détecter que la requête qu'il reçoit ne dispose plus d'un token valide et va donc renvoyer un code d'erreur 401 avec un nouveau lien de connexion. Le serveur web React va alors afficher à l'utilisateur un popup lui demandant de se reconnecter :

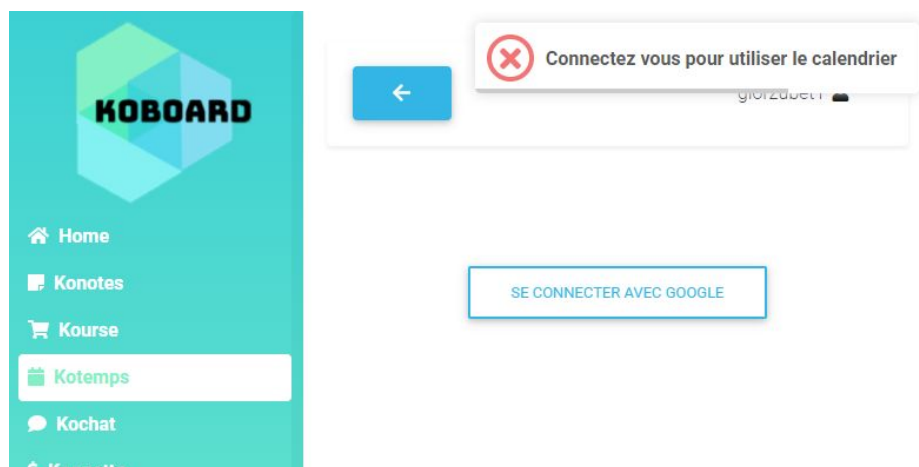


*Popup de reconnexion SweetAlert*

## 4.3. Les différentes technologies utilisées

### 4.3.1. SweetAlert

Dans le but d'améliorer l'expérience utilisateur, nous avons choisi de nous servir du package Node Js "SweetAlert2". Il s'agit d'un remplacement des popups de base de JavaScript par des boîtes très customisables, animées et responsives. Nous nous en servons notamment pour afficher des confirmations que les actions de l'utilisateur ont bien été prises en compte ou pour afficher des messages d'erreur.

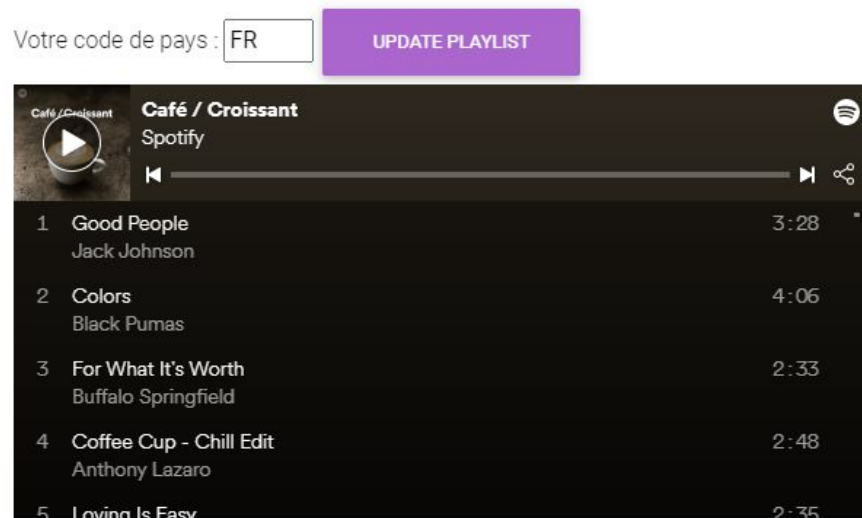


*Exemple de notification SweetAlert*



### 4.3.2. Spotify & Geonames

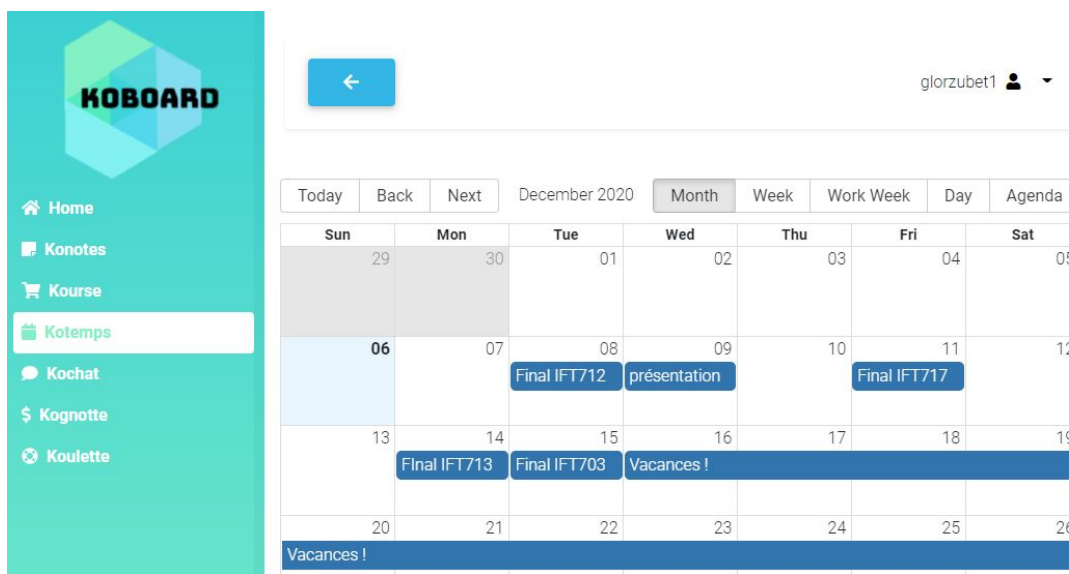
Afin d'intégrer l'aspect géolocalisation à notre projet, nous avons ajouté à la page d'accueil de notre site Web l'affichage d'une playlist Spotify. Cette playlist correspond à la top playlist écoutée sur Spotify dans le pays choisi. On récupère donc les coordonnées GPS du navigateur de l'utilisateur via les fonctions JS basiques, puis on envoie ces coordonnées à l'API gratuite de Geonames qui va nous permettre de déterminer le code de pays que l'on transmet ensuite à Spotify pour récupérer la top playlist du pays.



*La playlist spotify de la page d'accueil*

### 4.3.3. Google Calendar

Comme nous l'avons expliqué précédemment, notre site propose l'utilisation d'un calendrier Google. Afin de l'intégrer à notre site React, nous nous sommes servis de deux packages différents : D'abord [react-big-calendar](#) nous a permis de facilement afficher un calendrier complet, élégant et responsive. Puis [react-google-calendar-api](#) qui est là pour faire la liaison entre les événements affichés sur le calendrier et ceux stockés actuellement sur le calendrier de l'utilisateur.

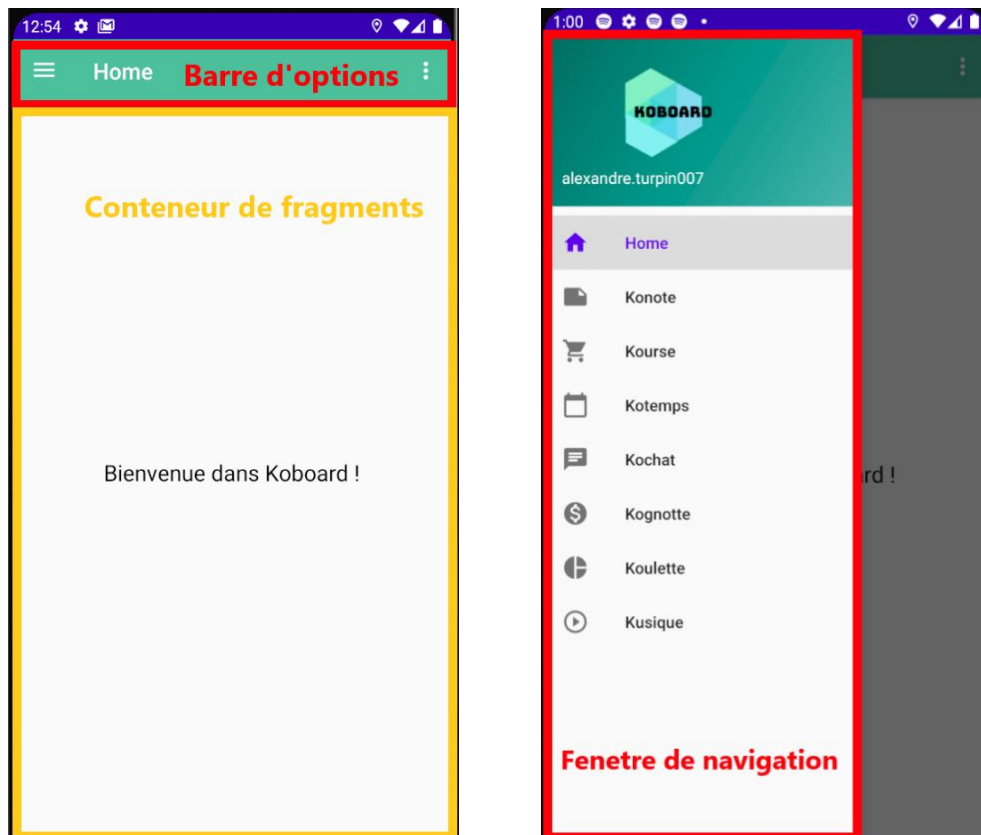


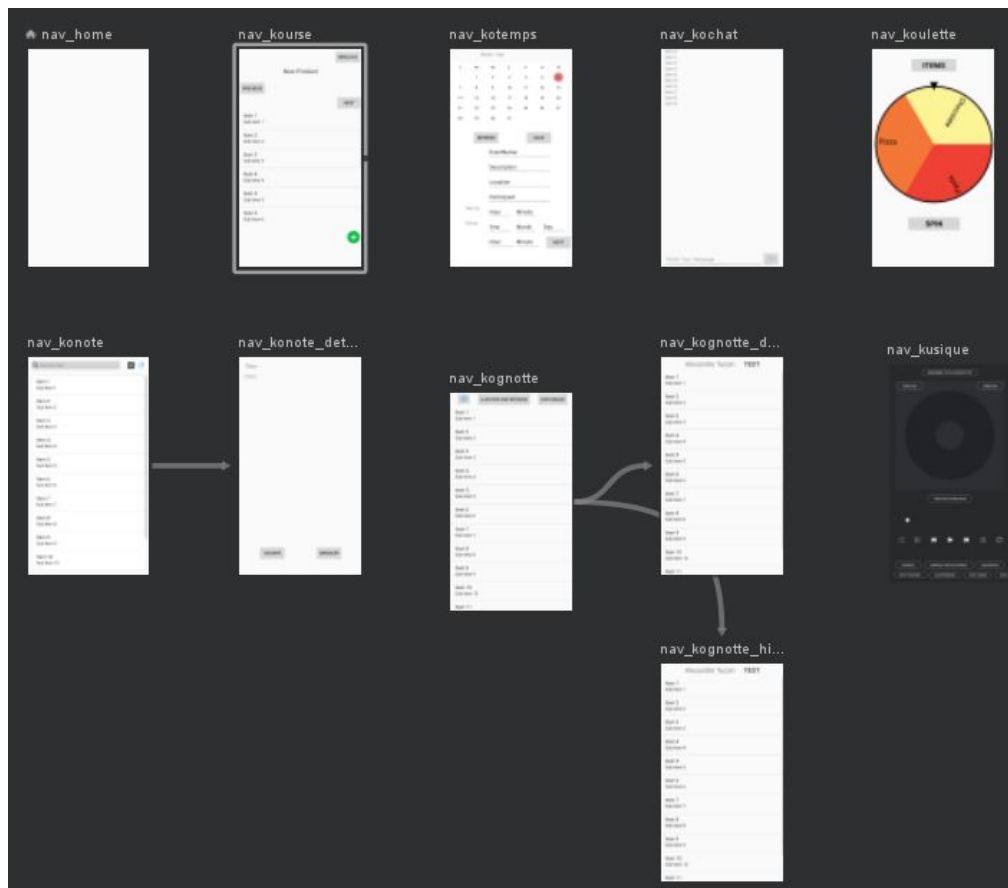
*Le calendrier*

## 5. L'application Android

### 5.1. Structure de l'application

L'application est découpée en deux activités. Une activité pour la connexion et une autre contenant le reste de l'application. La deuxième activité est une "Navigation Drawer". Elle est donc composée d'une barre d'option en haut d'un conteneur à fragment au milieu et d'une fenêtre de navigation qui s'ouvre sur la partie gauche de l'activité. C'est cette fenêtre de navigation qui va nous permettre de naviguer dans notre application entre les différents fragments.

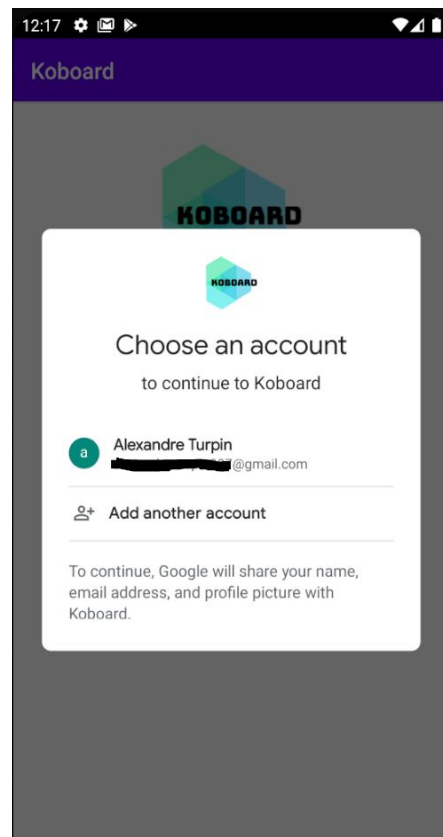
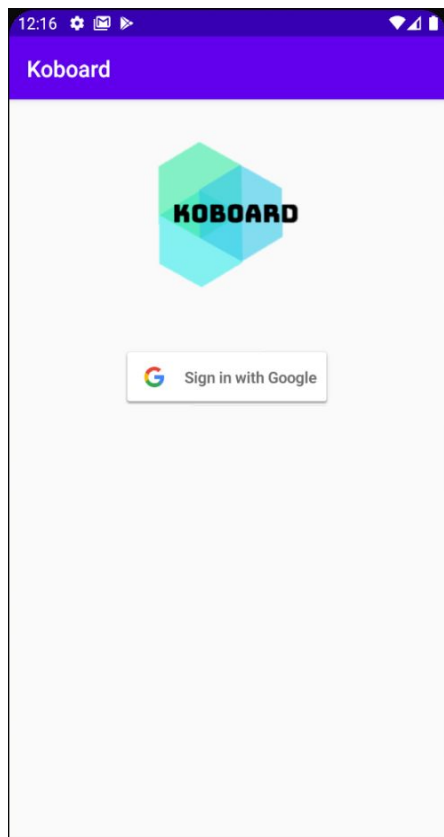




*Liste des différentes navigations possibles au sein de l'application*

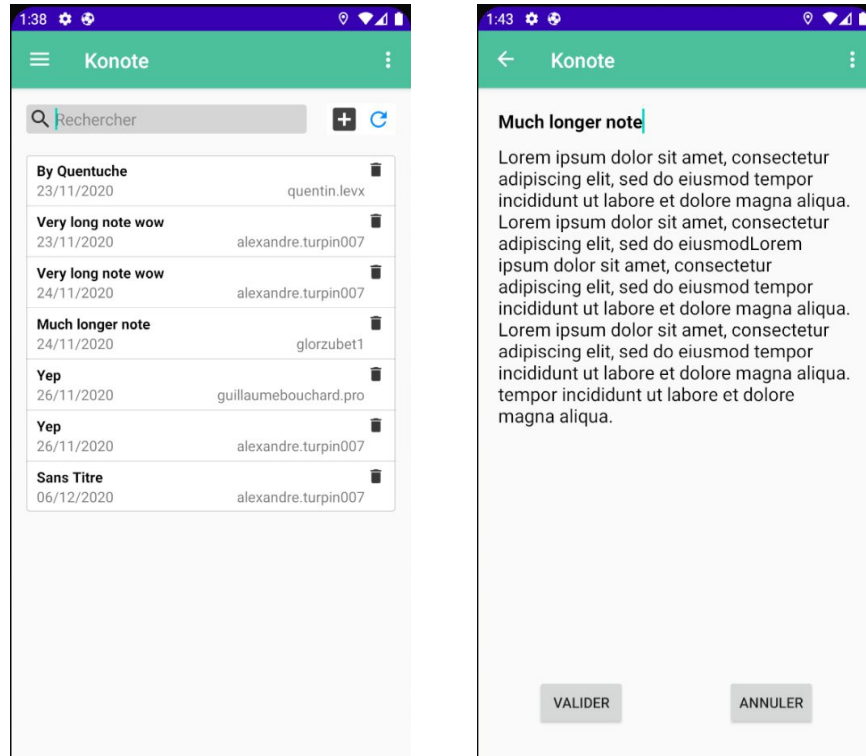
## 5.2. Login

La connexion au serveur se fait via google. C'est-à-dire que pour se connecter à l'application Koboard, il faut utiliser un compte google. Pour implémenter cela, nous avons utilisé l'API de google et les implémentations qu'il proposait. Une fois après avoir renseigné l'application dans nos projets de la console d'API de google nous avons pu afficher le bouton pour se connecter dans notre activité Login. Une fois que l'utilisateur s'est connecté, nous récupérons son code d'identification au serveur de Google. Nous envoyons ensuite ce code à notre serveur express qui va vérifier s'il est valide. Si c'est le cas, alors il nous renvoie un token permettant de communiquer avec le serveur express pour une durée limitée.

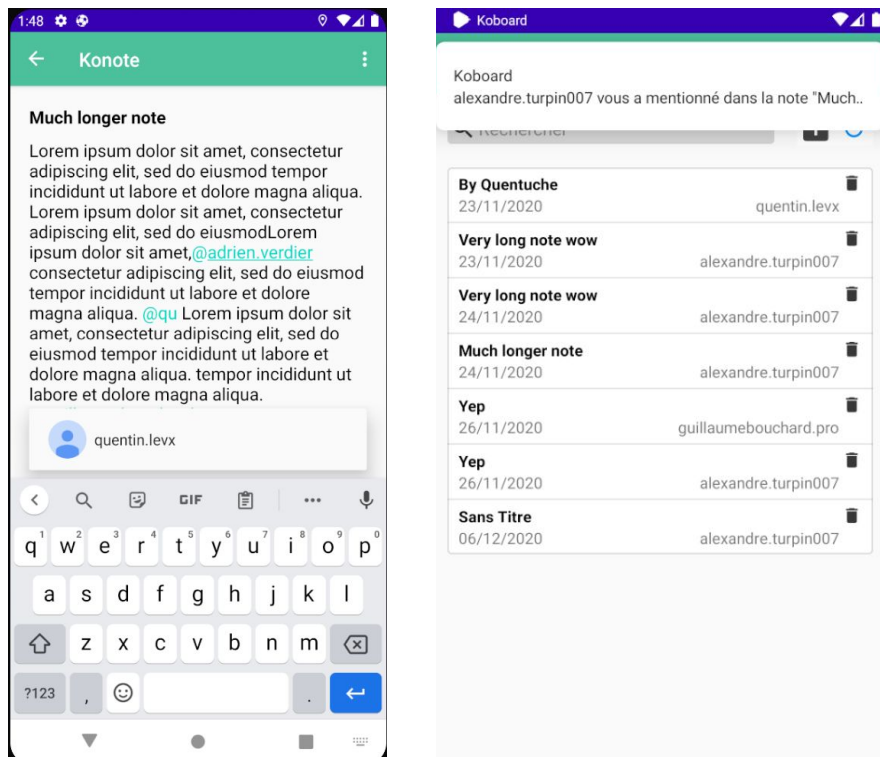


### 5.3. Konote

Ce fragment va nous permettre de partager des notes avec d'autres utilisateurs de Koboard. Plusieurs possibilités s'offrent à nous. Nous pouvons ajouter, supprimer et modifier une note.



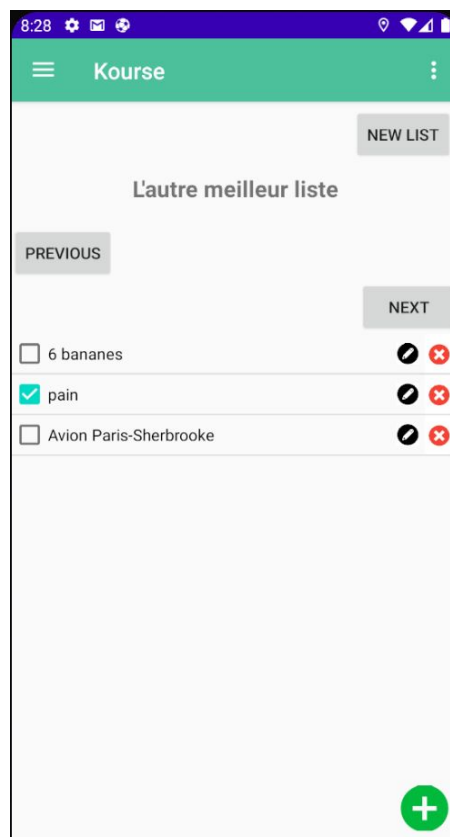
De plus, nous avons la possibilité de mentionner d'autres utilisateurs en mettant le caractère '@' devant leurs noms. Cela aura pour effet d'envoyer une notification à l'autre utilisateur via Firebase.



Le rafraîchissement des données se fait de deux manières. Soit automatique grâce au service qui tourne en arrière plan. Celui-ci est créé quand on arrive dans le fragment Konote et est supprimé quand on part de celui-ci. Sinon, il est possible de rafraîchir les données grâce au bouton en haut à gauche de la page. Vous avez aussi la possibilité de rechercher une note avec la barre de recherche.

## 5.4. Kourse

Kourse est un outil de gestion de listes de courses. Sur cette page, vous pouvez créer une liste de courses, modifier une liste de courses et les consulter.



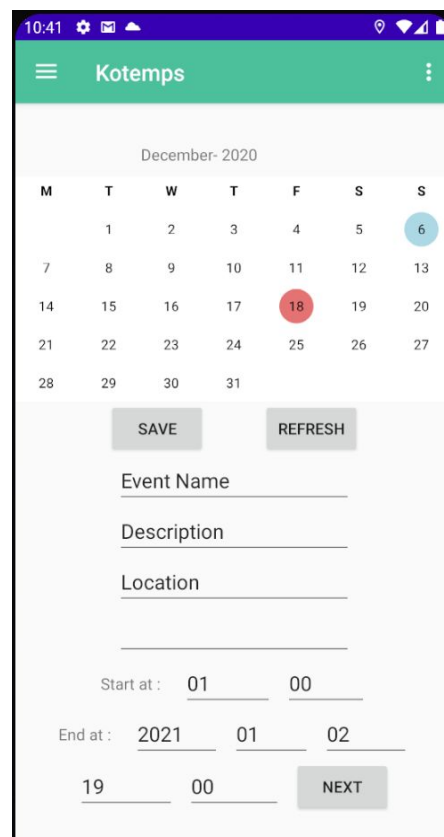
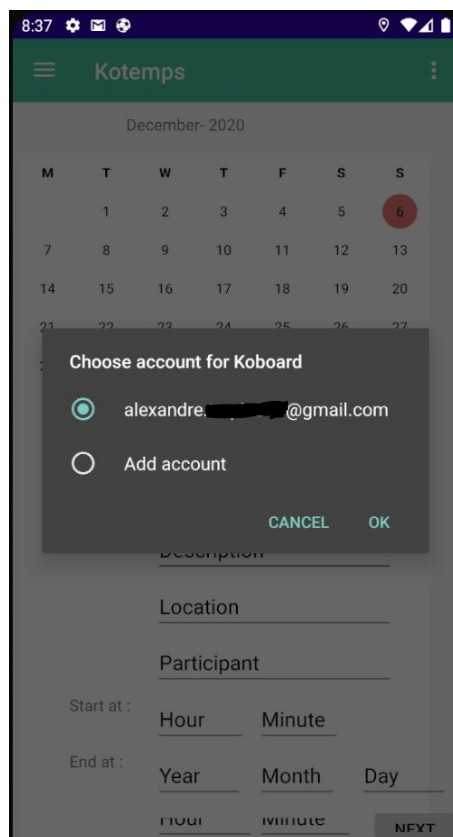
En haut à droite de ce fragment, on va pouvoir créer une nouvelle liste en appuyant sur le bouton "New list". Le titre de la liste qui est actuellement affiché se trouve centré en dessous de celui-ci. On est en mesure de naviguer entre les différentes listes via les boutons Previous et Next. Ces boutons vont avoir pour effet de modifier la liste de produit juste en dessous.

Sur cette liste, on peut supprimer ou modifier les produits en cliquant sur les deux petits boutons à droite de notre description. On peut aussi cocher les éléments une fois qu'ils ont été achetés. Et enfin, lorsqu'on appuie sur le bouton plus en bas à droite, une fenêtre va s'ouvrir pour nous proposer d'ajouter de nouveaux éléments dans la liste de courses.

## 5.5. Kotemps

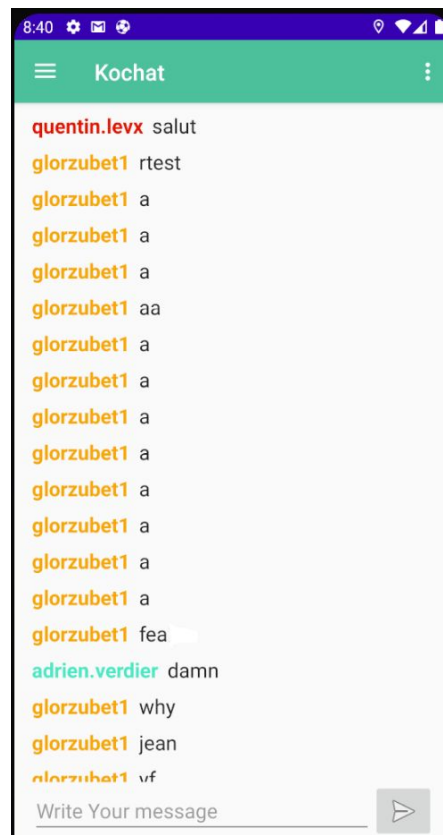
Ce fragment va nous permettre d'accéder à l'emploi du temps de l'application. On va d'abord sélectionner le compte google avec lequel on va se connecter et les différents évènements de l'utilisateur vont venir remplir le calendrier. Lorsqu'un événement est prévu sur la journée sélectionnée, toutes les informations concernant cet événement vont s'inscrire sous ce même calendrier. On va pouvoir naviguer entre tous les événements en appuyant sur la touche Next. En passant tous les événements, ou s'il n'y a pas d'événements ce jour ce sera la première option, renseigner les informations pour ajouter un nouvel événement. Un bouton pour enregistrer les actions ainsi qu'un bouton pour rafraîchir sont présents entre le calendrier et ces informations.

Ce calendrier est relié au compte Google de la personne connectée actuellement. C'est le calendrier personnel qui va s'afficher et les modifications vont être enregistrées par Google pour que l'on puisse y accéder ultérieurement.



## 5.6. Kochat

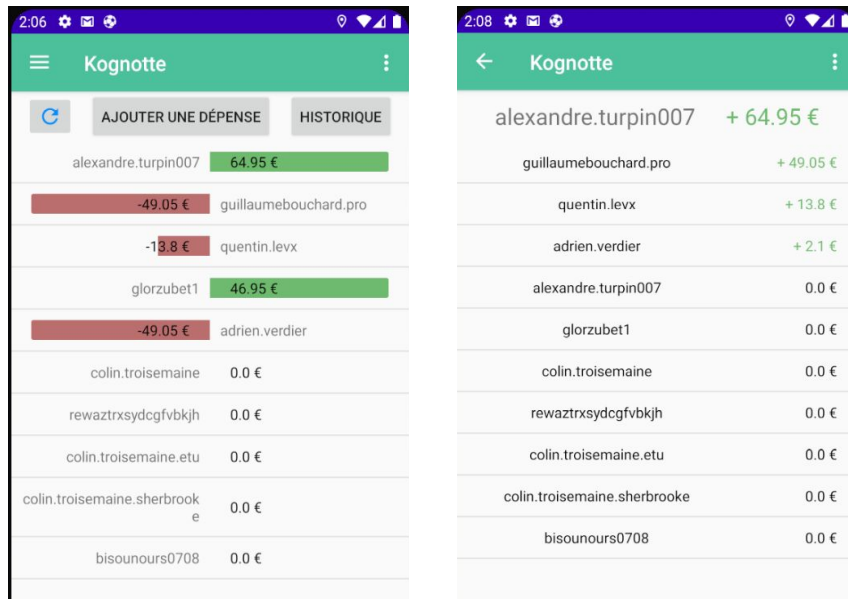
Kochat est un chat partagé entre tous les utilisateurs. Il permet d'envoyer des messages en direct. Tous les messages s'affiche dans la partie supérieure du fragment en indiquant le nom de la personne. Les messages vont venir s'afficher en direct lorsqu'une personne en envoie un. Si jamais on veut en envoyer, il suffit d'inscrire notre message en bas et de cliquer sur le bouton en forme d'enveloppe pour l'envoyer.





## 5.7. Kognotte

Kognotte est un outil de gestion de budget entre colocataires. En arrivant dans cet onglet, vous aurez dans un premier temps la liste de tous les participants avec ce que chacun doit à la colocation. Si vous appuyez sur un des participants, vous allez avoir le détail de ce qu'il doit faire pour équilibrer le budget. Il y aura notamment à qui il doit de l'argent et qui lui doit de l'argent.

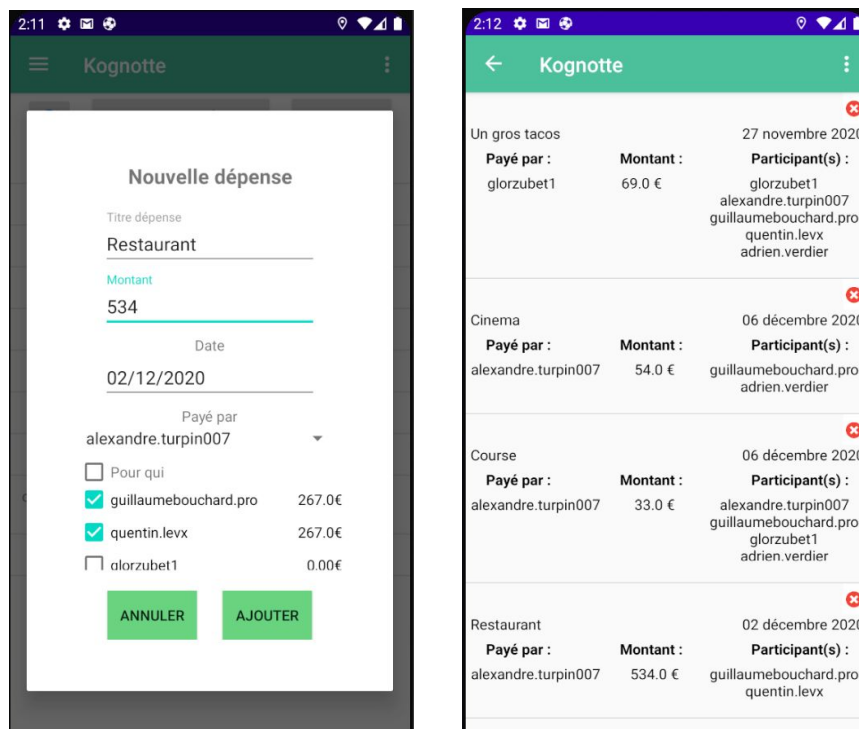


Participant	Montant
alexandre.turpin007	64.95 €
guillaumebouchard.pro	-49.05 €
quentin.levx	-13.8 €
glorzubet1	46.95 €
adrien.verdier	-49.05 €
colin.troisemaine	0.0 €
rewaztrxsydcgfvbkjh	0.0 €
colin.troisemaine.etu	0.0 €
colin.troisemaine.sherbrooke	0.0 €
bisounours0708	0.0 €

Participant	Montant
alexandre.turpin007	+ 64.95 €
guillaumebouchard.pro	+ 49.05 €
quentin.levx	+ 13.8 €
adrien.verdier	+ 2.1 €
alexandre.turpin007	0.0 €
glorzubet1	0.0 €
colin.troisemaine	0.0 €
rewaztrxsydcgfvbkjh	0.0 €
colin.troisemaine.etu	0.0 €
colin.troisemaine.sherbrooke	0.0 €
bisounours0708	0.0 €

Ensuite vous avez la possibilité d'ajouter une dépense et de voir l'historique des dépenses. Dans cet historique vous pouvez notamment supprimer les dépenses que vous voulez.



**Nouvelle dépense**

Titre dépense  
Restaurant

Montant  
534

Date  
02/12/2020

Payé par  
alexandre.turpin007

☐ Pour qui

☒ guillaumebouchard.pro 267.0€

☒ quentin.levx 267.0€

☐ glorzubet1 0.00€

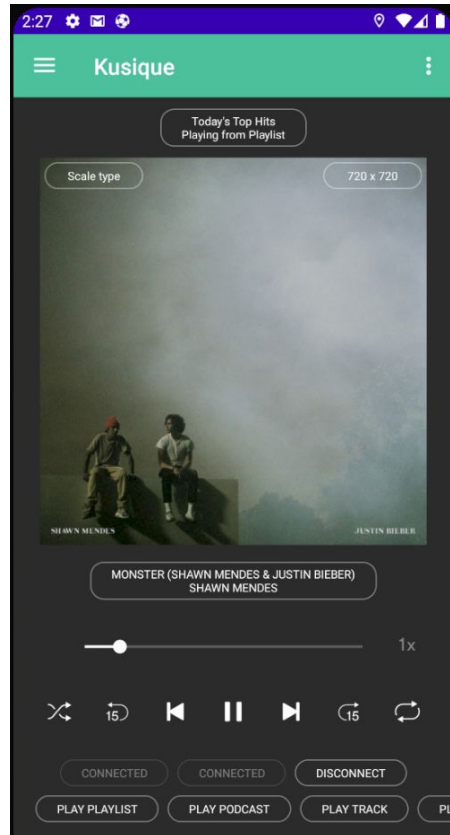
ANNULER AJOUTER

Description	Date	Payé par	Montant	Participant(s)
Un gros tacos	27 novembre 2020	glorzubet1	69.0 €	glorzubet1 alexandre.turpin007 guillaumebouchard.pro quentin.levx adrien.verdier
Cinema	06 décembre 2020	alexandre.turpin007	54.0 €	guillaumebouchard.pro adrien.verdier
Course	06 décembre 2020	alexandre.turpin007	33.0 €	alexandre.turpin007 guillaumebouchard.pro glorzubet1 adrien.verdier
Restaurant	02 décembre 2020	alexandre.turpin007	534.0 €	guillaumebouchard.pro quentin.levx

## 5.8. Kusique

Kusique est un fragment qui va nous permettre d'interagir avec l'application de Spotify. Il est donc impératif d'avoir Spotify installé sur le smartphone pour faire fonctionner correctement ce fragment.



Dans cette interface, nous pouvons donc passer à la musique suivante, mettre en pause votre musique et changer d'autres paramètres de lecture. Ensuite nous avons plusieurs boutons : des boutons pour connecter et déconnecter notre application à l'application Spotify, jouer une musique, playlist, album, ... Le bouton "Play playlist" est différent. En effet, il permet de jouer la top playlist du pays où vous vous trouvez actuellement. Pour ce faire, l'application va utiliser votre position GPS. Une fois connecté via votre compte google un service va être lancé qui va récupérer le pays où vous vous trouvez et va interroger l'api de Spotify pour trouver l'URI de la playlist correspondant à votre pays. Vous pouvez alors écouter cette playlist en appuyant sur le bouton "Play playlist".

## 5.9. Koulette

Koulette est un petit jeu de chance. Son fonctionnement est exactement le même que pour le site React : Il va nous permettre de tirer aléatoirement un nom parmi une liste qu'on lui fournit.

