Rapport développement mobile

Julien RENAUD, M1 informatique

6 novembre 2018

Résumé

Ce rapport traite du développement de DMSN : Developpement Mobile Social Nework

1 DMSN

Pourquoi DMSN : simple petit jeu de mot et clin d'oeil à MSN qui était l'un des premiers sans steppes de messagerie instantané (donc un des premiers réseaux sociaux ?)

DMSN permettra donc à terme de partager des photos ainsi que la position de l'utilisateur avec le reste du monde (semblable à Instagram).

Dans ce rapport, j'expliquerai ma démarche lors du développement de l'application ainsi que les problèmes rencontrés et les solutions adoptés pour les résoudre et je finirai sur un récapitulatif des fonctionnalités de cette version de DMSN.

Dans un premier temps, l'application a été développé pour Android avec android studio et le langage Kotlin, pour les test, j'ai utilisé lune version émulé du Nexus 6 de google.

2 Développement de l'application

2.1 Creation de compte

Une des premières étapes afin de faire cette application était de créer un système de connexion / enregistrement de compte.

Dans un premier temps, j'ai tenté de créer une base de donnée simple couplant adresse mail et mot de passe, mais les choses se sont compliqués quand il a fallu ajouter les différentes options de sécurité et de confort tel que :

— verification de l'adresse mail (confirmation via lien mail)
— Verification de la complexité du mot de passe
— Récupération de mot de passe

Ceci fonctionnant avec une simple base de donnée MySQL.

Bien que la tâche ne soit pas impossible (et après de nombreuses heures à tenter de communiquer avec la base de donnée), j'ai découvert un framework permettant l'integration Android et IOS : Firebase.

Firebase est l'outil parfait pour le projet de réseau social car il met à disposition un système d'authentification de base de donnée, de stockage, d'hébergement ainsi que d'autres services qui peuvent s'avérer utile.

L'installation de ce Framework est assez simple bien qu'il m'ai fallu du temps pour comprendre le fonctionnement et la mise en place, un tutoriel pas à pas est disponible sur le site officiel de Firebase, aussi bien pour Android que pour IOS.

Une fois le service en place et après différents test, l'application ne reconnait aucun utilisateur enregistré dans la base de donnée disponible sur le site (alors que je m'étais connecté avec une adresse mail dédié). Le problème venait du fait que l'application nécessite un token (RequestIdToken) afin de connecter l'application avec Firebase. Il fallait utiliser le Debug token fournir par google et non le public token.

L'application se connecte maintenant sans problème et mon adresse mail apparait sur la base de donnée disponible sur ma console (sur le site Firebase).

2.2 Fil d'actualité

Deuxième étape : créer un fil d'actualité avec les photos des utilisateurs de l'application avec un tri par affinité.

Les difficultés sont les suivantes :

Fil d'actualité : En fonction du temps' En fonction des preferences de l'utilisateur' Au hasard'

Photo des utilisateurs de l'application : Necessité de pouvoir héberger les photos en ligne (ainsi que toutes les données liés tel que les commentaires et la localisation et nécessité de pouvoir les récupérer (ou en récupérer une partie)

Tri par affinité : besoin de créer un système d'amitié virtuelle (liste d'amis comme facebook ou autre)

L'hébergement de photo semble fonctionnel mais la récupération des photos en ligne n'est pas possible, la fonction a donc été mis de coté. Il sera peut nécessaire de passer par un protocole HTTP pour récupérer le maximum d'information sans passer par les permissions Android.

Cette étape a été réduit au stricte minimum : possibilité de voir les différents post fait par l'utilisateur.

Un fil d'actualité impliquait une génération dynamique des imageView dans un linearLayout ce que je n'ai pas réussi à faire dans cette version, ce système a été simplifié par la possibilité de voir les photos prises et de changer de photo à l'aide d'un double tap sur l'écran.

Le fil d'actualité devait permettre l'accès à plusieurs commandes, notamment la création d'un nouveau post, ce qui a été intégré à l'aide d'un bouton en bas de l'écran.

2.3 Prise de photo

L'appui sur ce bouton permet d'accéder à la page de création de « post ». Un post est simplement une image, un commentaire et la localisation.

Pour la prise de photo, j'ai utilisé le logiciel de capture natif du téléphone. Plusieurs problèmes ont été rencontré :

- Les permissions d'utilisation de l'appareil photo : l'ajout des <use-permission> n'a pas été suffisante dans AndroidManifest.xml, il a fallu demander la permission à l'utilisateur directement. Pour cela j'ai utilisé un petit framework qui permet de faciliter la demande de permission : « RxPermissions » qui permet de demander une permission et de réagir en fonction de la réponse de l'utilisateur et cela en une seule commande.
- La rotation de la photo : en mode portrait, la photo a une rotation de 90 degrés qui ne change pas seul en fonction de la configuration du téléphone, il est donc nécessaire de détecter le mode (paysage droite ou gauche, portrait droite ou gauche) afin de régler la rotation de la photo.
- La sauvegarde de la photo dans la galerie, malgré les différents réglages, les photos ne sont pas trouvable dans la gallérie donc finalement enregistré dans un dossier spécialisé.

,

2.4 Géolocalisation

Puis vient la localisation, un bouton permet d'accéder à la google map et doit pointer sur la localisation actuelle du téléphone, cependant ici aussi plusieurs problèmes ont été rencontrés :

- Malgré les différentes demandes de permissions, l'application crash systématiquement lorsque j'essaie de récupérer la position actuelle, la fonction a donc été retiré.
- Lors de l'utilisation du retour au parent, l'activité se relance en passant par le OnCreate ce qui fait perdre les informations de la page et relance la boite de dialogue demandant la source de la photo. Aucune solution a été trouvé.

,

3 Fonctionnalités

Récapitulatif des fonction de cette version de cette version de l'application android :
— Connexion à l'application via compte google existant
— Affichage des photos déjà prise sur l'application
— faire defiler les photos avec double tap
— Création d'un nouveau post (utilisation de la camera / prise de photo)
— Géolocalisation (sans position actuelle GPS'.Donc juste ouverture de google map dans l'application)

À terme l'application doit implémenter au minimum les fonctionnalités suivantes :
— Géolocalisation fonctionnelle
— Fil d'actualité fonctionnel en défilement vertical des publications.
— Possibilité de supprimer des post
— Possibilité de poster en ligne et de récupérer les post des autres utilisateurs
— Gestion de liste d'amis
— Possibilité d'authentification avec mail / mot de passe et autre comptes connus (facebook' Instagram')
— Possibilité de noter les photos des autres utilisateurs

4 Conclusion

Kotlin est un langage prometteur, très different de java et presque plus haut niveau il est facilement abordable mais comprend quelques differences de syntaxes qui apportent de la confusion (typage par exemple)

Même si l'utilisation d'outils cross-plateforme tend à se démocratiser (Unity, xamarin, react ?). Il est toujours intéressant pour un développer mobile de savoir comment fonctionne spécifiquement chaque environnement afin de pouvoir agir lorsqu'un problème survient sur une plateforme spécifique.

Ce projet même si il n'a pas été mené jusqu'au bout m'a permis d'en apprendre plus sur les langages qui tendent à se démocratiser (Kotlin et Swift) en laissant de coté Java et Objective-C.

Lien vers le projet Android https://github.com/jStrider/DMSN2.git