**Cahier des charges**

**PFE : Application ‘Wifi share app’**

****

**Table des matières**

[Résumé 1](#_Toc96267956)

[Etapes de développement 1](#_Toc96267957)

[Fonctionnalités 1](#_Toc96267958)

[Pré-requis 1](#_Toc96267959)

[Première activité : SendActivity – Envois de données 3](#_Toc96267960)

[Boutons 3](#_Toc96267961)

[Menu 4](#_Toc96267962)

[Seconde activité : AnalyseLAN – Analyse du réseau LAN 4](#_Toc96267963)

[Boutons 5](#_Toc96267964)

[Menu 6](#_Toc96267965)

[Protocole de transfert 6](#_Toc96267966)

# Résumé

Dans le cadre du projet de fin d’études, j’ai développé une application mobile sous Android qui permet d’échanger des fichiers entre utilisateurs de la même application.

## Etapes de développement

Il y a cependant 3 étapes de développement à prendre en compte avant de lire la suite du document.

La première étape consiste à développer l’application où la seule fonctionnalité de base est de pouvoir envoyer un message à une adresse IP demandé à l’utilisateur.

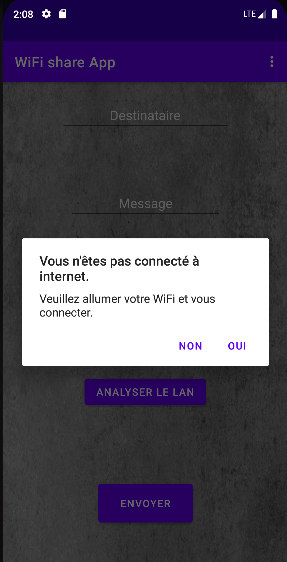
La seconde étape est de développer la fonctionnalité de recherche d’appareils sur le réseau LAN via le protocole Wifi.

La dernière étape consiste à remplacer l’envoi de message par des fichiers.

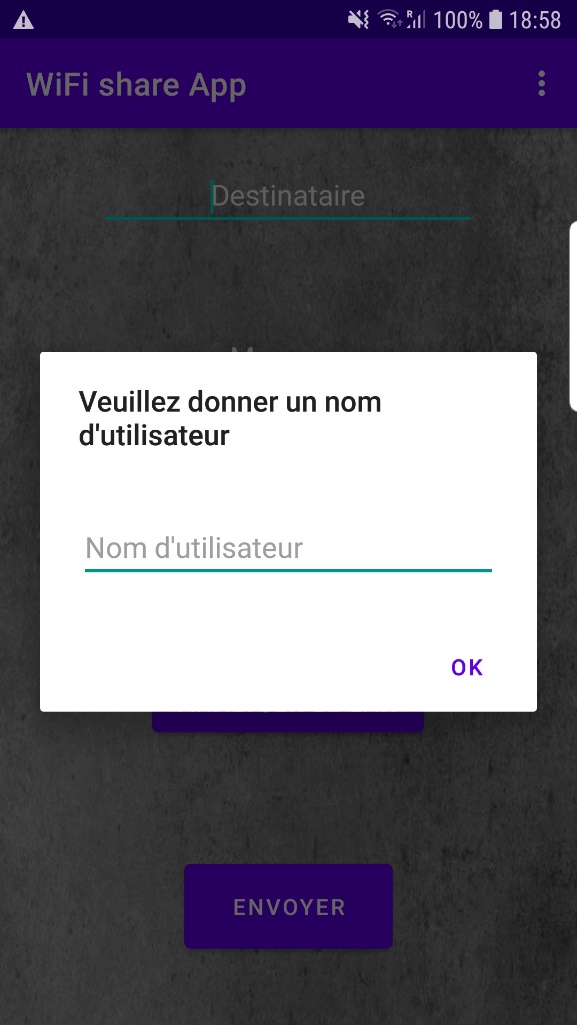
# Fonctionnalités

## Pré-requis

L’utilisateur aura comme première tâche, dès le lancement de l’application, d’allumer son Wifi sur son smartphone.

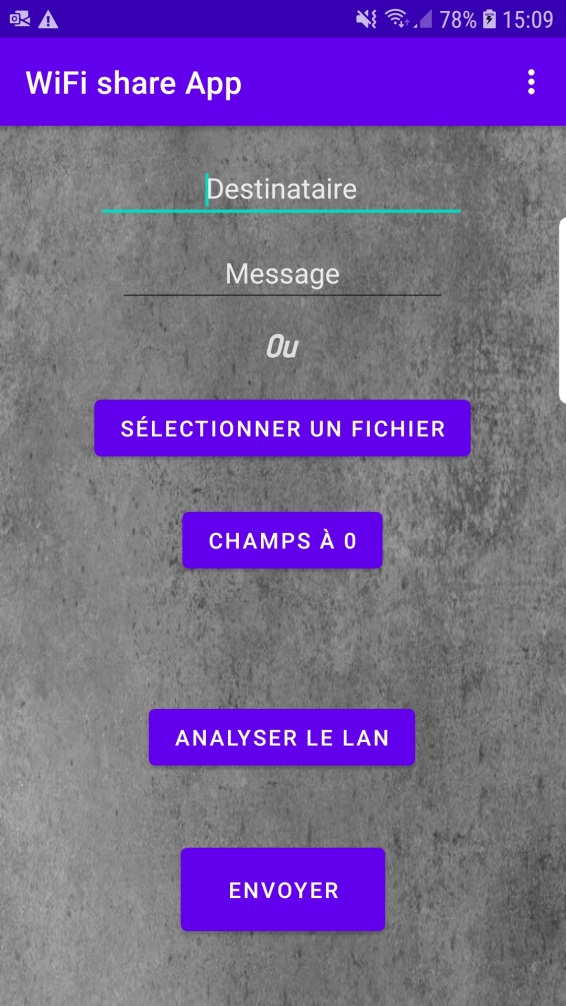


Dès qu’il est allumé, on lui demandera un nom d’utilisateur uniquement lors du premier lancement de l’application. Ceci permettant la reconnaissance de l’appareil via un nom au lieu d’une adresse IP.



## Première activité : SendActivity – Envois de données

Après ces 2 premières étapes, l’utilisateur peut enfin commencer à naviguer dans l’application avec comme première activité : *SendActivity*.



### Boutons

- 'Sélectionner un fichier' Ce bouton permet d'ouvrir la galerie du smartphone de l'utilisateur pour séléctionner un fichier à envoyer

* ‘Champs à 0’

Ce bouton permet de réinitialiser les champs ‘Destinataire’ et ‘Message’.

* ‘Analyser le LAN’

Ce bouton permet de passer à l’activité *AnalyseLAN*.

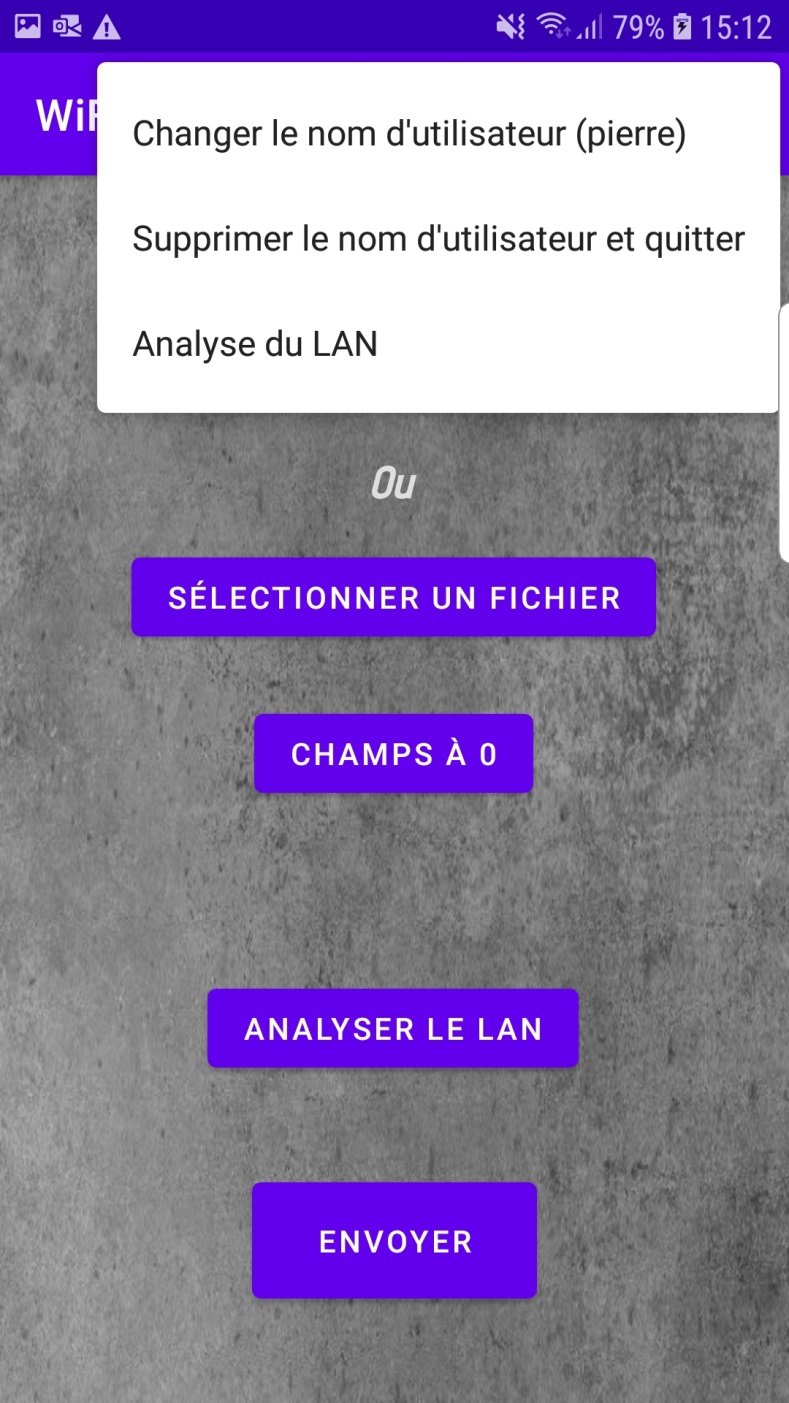
* ‘Envoyer’

Ce bouton permet d’envoyer le ‘message’ ou le fichier sélectionné au ‘destinataire’.

Attention : Une vérification des champs ‘Destinataire’ et ‘Message’ existe pour vérifier si :

L’adresse du destinataire est bien valide et que le message ne soit pas vide.

### Menu



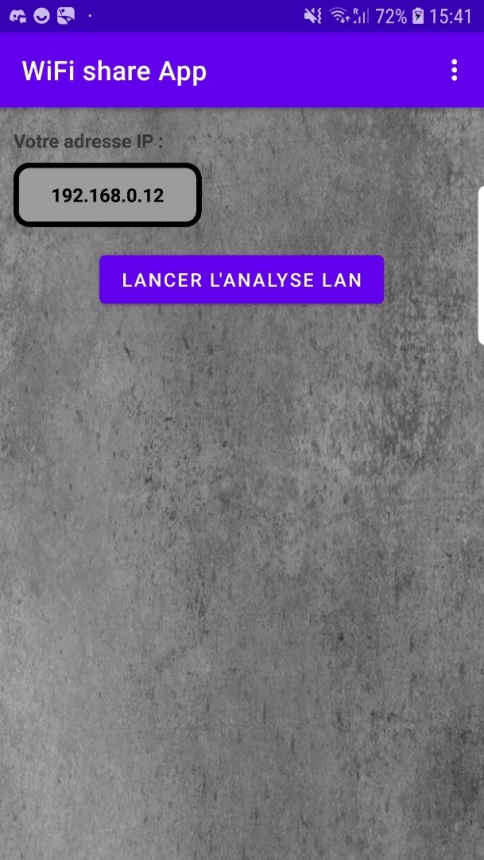
La première option permet de changer le nom d'utilisateur (ici 'pierre').

La seconde option permet de réinitialiser l'application avec les paramètres d'usine.

La troisième option permet de passer à l’activité *AnalyseLAN* en cliquant sur l’option ‘Analyse du LAN’.

## Seconde activité : AnalyseLAN – Analyse du réseau LAN

Comme précisé dans le résumé, la seconde étape est de développer la fonctionnalité d’analyse du réseau à la recherche d’autre appareils Android qui ont leur application *Wifi share app* en fonctionnement.



*Pour information à l’utilisateur,*

*son adresse IP est visible dès l’activité lancée.*

### Boutons

Le seul bouton de l’activité est ‘Lancer l’analyse LAN’. Ce qui va rechercher les autres appareils Android qui ont l’application *Wifi share app* en cours d’exécution.

Après l’analyse, une liste des appareils disponibles sont affichés. Avec en plus une bannière indiquant la date du moment où l’analyse a été effectuée.



*Ici, on retrouve les informations comme le nom d'utilisateur,*

*son adresse IP ainsi que le statut du port pour l'émission et réception de données*

Il est possible alors de cliquer sur un élément de la liste pour ensuite envoyer à l’utilisateur sélectionné un message ou un fichier. Cette action permet donc de retourner à l’activité *SendActivity*.

### Menu



Dans le menu déroulant de l’activité *AnalyseLAN*, il est possible de changer le nom d'utilisateur en cliquant sur l'option 'Changer le nom d'utilisateur'.

Il est aussi possible de retourner à l’activité *SendActivity* en cliquant sur l’option ‘Envoyer des données’.

## Protocole de transfert

Pour l’envois de messages/fichiers il existe 2 protocoles connus, TCP et UDP.

TCP permet de vérifier à chaque parquet envoyé d’attendre un accusé de réception de la part du destinataire. Ceci permet donc de faire le transfert en toute sécurité.

UDP permet d’envoyer les paquets sans l'utilisation de timeout par rapport au protocole TCP. En effet, celle-ci ne n’attend pas de la part du destinataire un accusé de réception.

L’application utilisera ces deux protocoles dans des situations différentes. Pour la recherche d’appareils Android avec l’application *Wifi share app*, le protocole TCP est utilisé. En effet, pour l’analyse, il n’est pas nécessaire d’envoyer des gros paquets de données.

Pour le transfert de messages/fichiers, le protocole UDP est utilisé. La vitesse sera donc un atout pour cette fonctionnalité.

Si jamais le destinataire se déconnecte du réseau alors que le transfert n’est pas terminé, celle-ci sera coupée. Cette vérification se fait par le moyen d’une connexion TCP de manière périodique.

***Mise à jour***

UDP n'est plus à utiliser pour l'échange de fichiers.

Voir le lien <https://stackoverflow.com/questions/3396813/message-too-long-for-udp-socket-after-setting-sendbuffersize>

De *user207421* :

*"When UDP is fragmented, it loses the datagram if a fragment is lost, because there is no retransmission."*

En effet, comme le destinataire a besoin de recevoir toutes les données, on ne peut pas permettre qu'un paquet n'ait pas été transmis. De plus, le protocole UDP ne retransmet pas les paquets perdus.

TCP est alors la meilleure option.

Dans le repository gitlab, il y a 2 versions :

- une version qui utilise le protocole TCP pour la recherche et UDP pour l'envoi de petits fichiers et messages.

- une version finale qui utilise le protocole TCP pour la recherche et l'envoi de messages et fichiers.

Le repository contient aussi un Serveur.java qui simule la réception de messages et de fichiers ainsi que la recherche.