Project Report
Principles of Software Development

Rafael Ramires (120004951B) 30-05-2021 Professor: Tegawendé Bissvande

App - ChatClose

Concept Introduction

This application was developed in the context of the *Principles of Software Development* course and has the objective of testing some concepts learned during the classes.
For that it was asked that we develop a messenger style application that allow the users to communicate with each other when they are close by using a peer-to-peer connection without a database sending the messages directly to other nearby users instead of having a server responsible for sending and saving the messages.

App Foundation

To develop this application, I used an already an API developed by google called e ***Google Nearby Connections***, [1] that is a peer-to-peer networking API that allows apps to easily discover, connect to, and exchange data with nearby devices in real-time, regardless of network connectivity. The API is located in the **com.google.android.gms.nearby.connection** package. Nearby Connections enables advertising, discovery, and connections between nearby devices in a fully offline peer-to-peer manner. The app is based on Bluetooth and Wi-fi connections and for convenience these features are enabled automatically as they are required and restored to its prior state once the app is done.

App Structure

After the decision of using *GNC API* the best option to do the app was in a *Host-Client* format, where we have 2 different profiles of user. The *Hosts* that are able to create new chat rooms for other users to join, and the *Clients* that are able to search for chats already created by previous *Hosts*. When a *Client* tries to join a new chat it's created a new Endpoint between them to allow the messages as well the information about the others user to pass throw a safe channel. The structure of the app is the following:

1. **Home Screen:** The user can insert his username and submit it.
2. **Option Screen**: The user can decide if it is going to be a *Host* or a *Client*.
3. **Host Screen:** If the user decided to become a *Host* he comes to this screen where he will be able to see a list (RecyclerView) of the *Clients* that joined his room and have a button to start texting them.
4. **Client Screen:** If the user decided to become a *Client* he comes to this screen where he will be able to see a list (RecyclerView) of chats already created by other *Hosts* as well as the number of unread texts they have in each of the chats and can decide to open one of them to start texting.
5. **Chat Screen**: The users can write messages in this screen as well as seeing a RecyclerView of messages that other users send with their respective name and time of receival.

A demo video of the app working can be found here: https://youtu.be/SMUgSAio0Mw

Implemented concepts

This application is mainly based on several concepts learned in the classes such as [2]:

1. **Activities:** Put together is a fundamental part of the platform's application model ,the whole app uses several activities (ex: MainActivity, HostActivity…).
2. **Views:** Implemented in xml files several elements were used (ex: Buttons, TextViews, RecyclerViews…).
3. **Manifest:** The manifest file describes essential information about your app to the Android build tools and the Android operating system.
4. **Intents:** An abstract description of an operation to be performed.
5. **Layouts:** Defines the structure for a user interface in your app, such as in an activity.
6. **Design Patterns:** Adapter, Composite, Observer…
7. **Shared Preferences:** Object points to a file containing key-value pairs and provides simple methods to read and write them.

References

- [1] - https://developers.google.com/nearby/connections/overview
- [2] - https://developer.android.com/guide (several subpages)