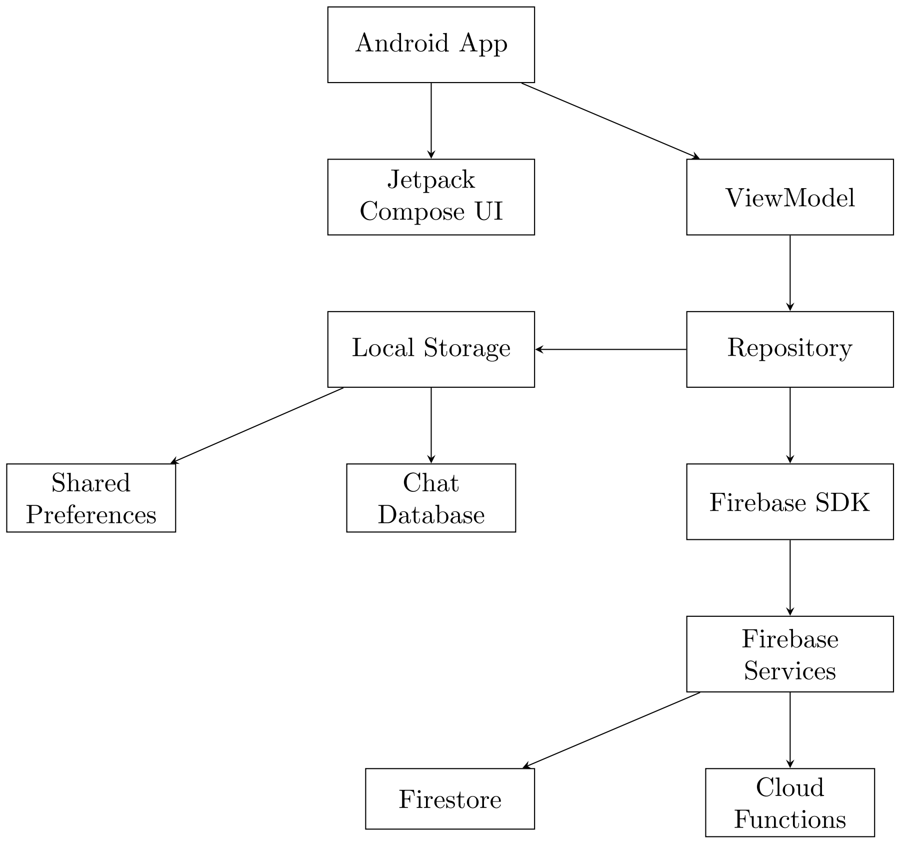
Assignment 3
'Whisper'
System Design

Joakim Tollefsen Johannesen
joakimtj@hiof.no

Niklas Berby
niklab@hiof.no

2024

1 Technical Structure



- Data storage and management:
  - Firebase Firestore to store chats, messages and user data.
  - Firestore will handle chat and message synchronization.
  - E.g., room exipration.
- User management:
  - There is no user authentication planned due to the anonymized nature of the app.
  - Most relevant user data (preferences as such) will be stored locally.

2 Navigation

Our app is comprised of four main screens.

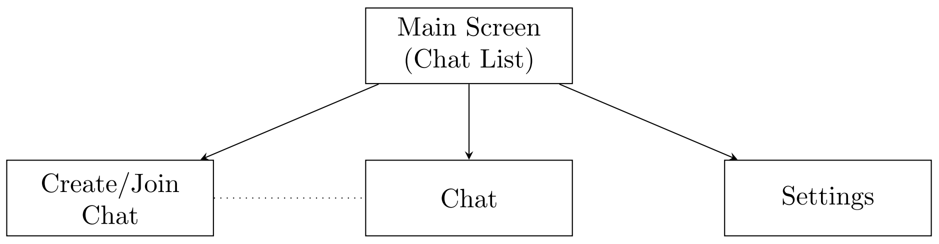
- Main Screen
- Chat
- Settings/Profile
- Create / Join

The main screen will list your currently active chats – as well as when they are set to expire. A floating action button is present that will allow you to create a chat or join an existing one.

The settings/profile screen is where you will select your display name, your tripcode, your chat bubble customization and your picture. (optional)

The create / join screen allows you to enter a code, if that code is in use (e.g., a chat is actively using that code), you will join it, if not, you will create a new chat. You can also join existing chats using a QR code.

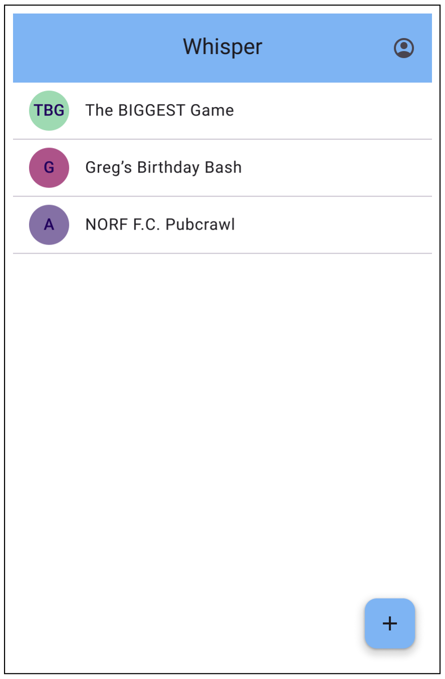
Below is a graph representation of the app's navigation. The screens are unlikely to change much from the project proposal outside of changes made to comport with Material Design principles.

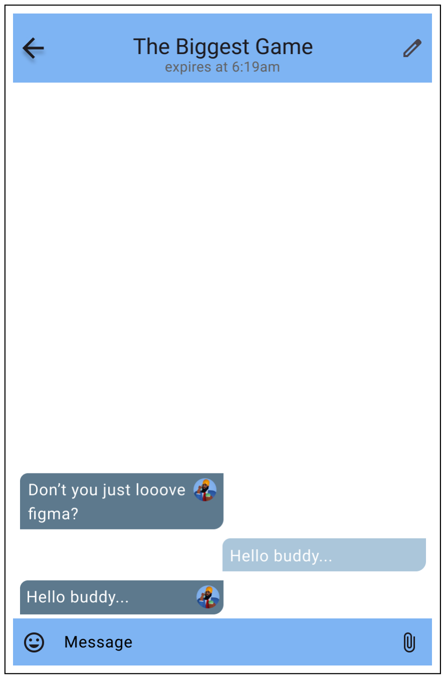


The dotted lines represent navigation to the chat that has been joined/created.

After some consideration, we've decided that we will join the Create and Join screens together.

The primary screens we have planned are shown in the images below.

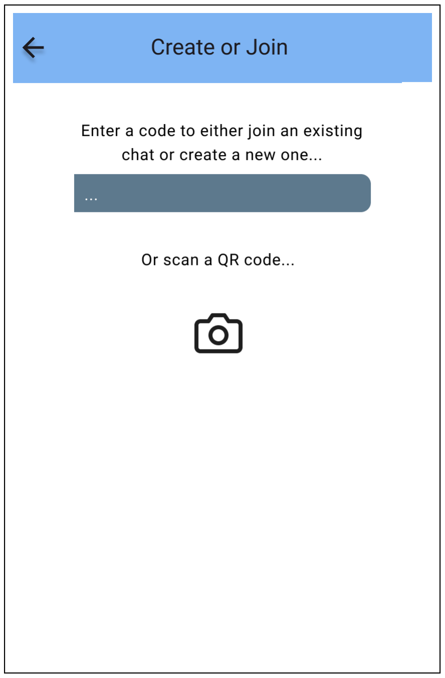




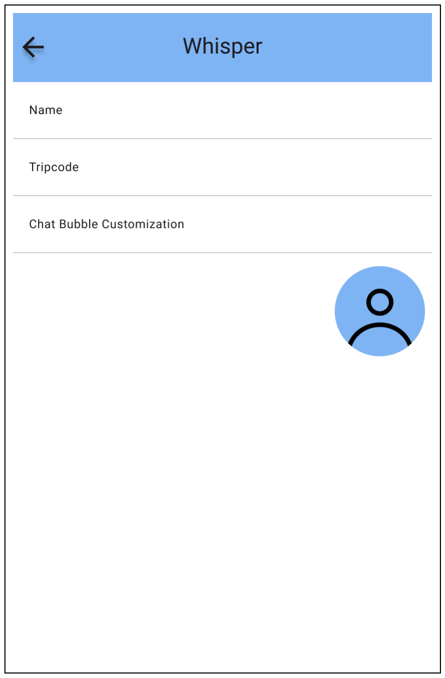
Don't you just looove figma?

Hello buddy...

Hello buddy...



Enter a code to either join an existing chat or create a new one...



3 Resources

We plan to source most art assets externally if we deviate from this we will, of course, document it. It's a little difficult to make this decision right now. But we do know that the following will be sourced externally:

1. Material Design symbols for common actions.
2. Sound effects (if implemented). E.g. push notifications.

4 Classes

We intend to implement these classes.

```
MainActivity: Entry point of the app.

WelcomeViewModel: Handles logic for joining/creating rooms.

ChatViewModel: Manages chat messages and room data.

ChatRepository: Interacts with Firebase services.

User: Data class for user information (nickname, tripcode).

ChatRoom: Data class for room information.

Message: Data class for chat messages.

TripcodeGenerator: Utility class for generating tripcodes.

DateTimeUtils: Utility class for handling date/time operations.

FirebaseManager: Singleton for initializing and accessing Firebase services.
```

This is not an exhaustive list and there may be more that we are missing but those will be documented as we add them.

5 External libraries/dependencies

Here are the external libraries that we intend to use. We may find better alternatives as we develop the app.

- Firebase SDK
- Jetpack Compose
- Jetpack Navigation
- Kotlin Coroutines
- Hilt
- Coil

6 Help

The Firebase integration and real-time synchroniation between server and device is probably what we might need the most help with.