GitHub Username: rafaelanastacioalves

Speko

Description

Speko is a 100% language exchange App aimed at practicing conversation between students through audio and video calls. Speko allows the language student to achieve fluency in record time by talking to students already fluent in the language anywhere in the world.

Intended User

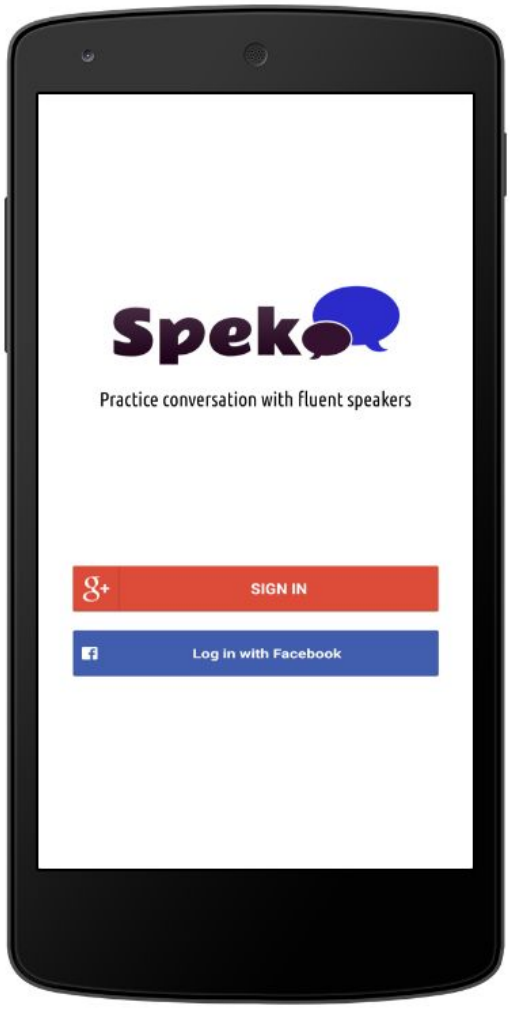
Our main audience are young people and adults interested in practicing and learning new languages.

Features

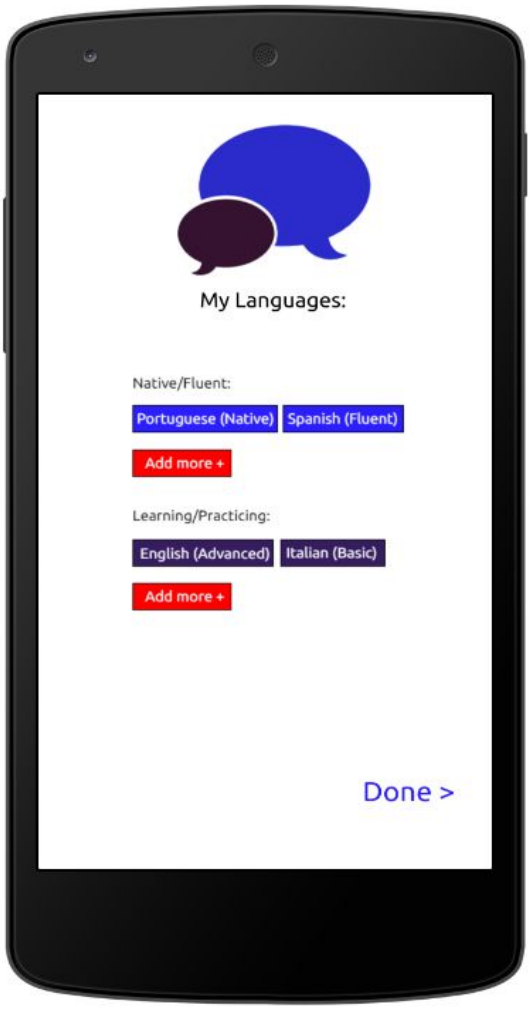
- Add Users;
- Video Call.

User Interface Mocks

Screen 1 (Login)



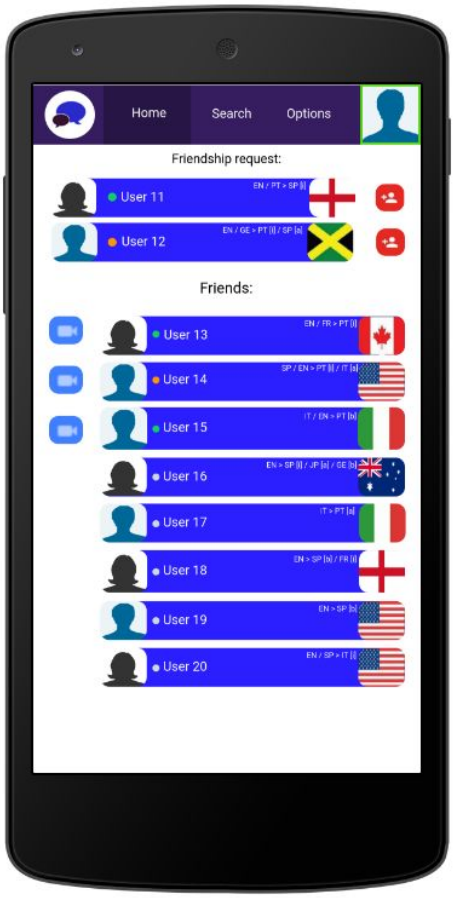
Screen 2 (Languages Selection)



*The **Screen 1** is the Login Screen, the user will choose one of the options of signing (google or facebook) that he will use in Speko, the data of this account will be used in the automatic creation of his profile in the App.*

*In the next screen (**Screen 2**), the user will complete his registration by selecting the languages he speaks natively/fluently and the languages he wishes to practice/learn using Speko.*

Screen 3 (Home)



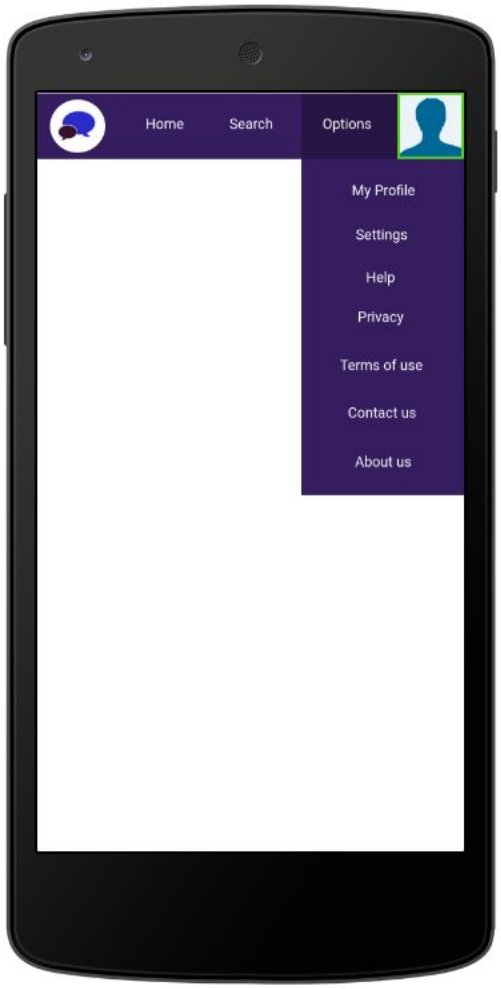
Home screen: Displays the Friends list, with available contacts on top of the others.

- *On the left side of the Friends list names (Just ONLINE) there is the button to send a call request - if the user of the list that got called accepts the request, the call will be initiated;*
- *At the top of the screen appear the "friendship requests" - to accept, just click the red button next;*
- *In the names of users on the lists there is also the information of the native and fluent language(s) of the user and the language(s) that he is practicing.*

Screen 4 (Search)

Search Screen: Shows all users currently available. At the right side of each list name is the "add user" button (red), which sends a friend request to the user.

Screen 5 (Options Guide)



Options: Clicking on the Options' button in the top menu will open a "Guide" with some options that will take you to their respective screens.

Screen 6 (Widgets)

3x4 and 4x4 widgets in order to show the user's friends

Key Considerations

How will your app handle data persistence?

It'll combine AbstractThreadedSyncAdapter class with a Service class in order to sync information from the web service.
All information will be stored in locally by using our own Content Provider associated with SQLite.

Describe any corner cases in the UX.

The user is already logged in but he is offline: The user will be able to see his friend list with the information from when he was last logged.Additionally, he'll see a permanent notification on the top of the screen he'll not be able to see any friends details or to search for new friends.

Describe any libraries you'll be using and share your reasoning for including them.

I'll be using:
Picasso for showing the user's image.
Libjingle (io.pristine:libjingle:11139), which uses WebRTC for video conference.
Socket.io for signalling protocol messages (com.github.nkzawa:socket.io-client:0.4.2)

Describe how you will implement Google Play Services.

I'll use "Google Sign In" (and "Facebook Log In" as well) framework in order to allow user to sign up and sign in in the app. I'll integrate this with our backend in order to manage user's information and logged status.
Additionally, I'll use Firebase for managing user's account info.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

Task 1: Project Setup

Create a new Android Studio project.

- Configure a Google API Console project and set up Android Studio project.
- Configure Facebook Login.
- Setup a backend service for persisting user information remotely.
- Setup a backend service for signalling protocol messages.

Task 2: Implement Signing Process

- Build UI for Signing.
- Integrate Google Sign In;
- Integrate Facebook Log In;
- Build UI for completing Signup process.
- Implement completing Singnup process.

Task 3: Implement Listing User's Friends

- Build UI For List of User's Friends
- Implementing retrieving list of User's Friends

Task 4: Implement Adding Users

- Build UI For Adding user's friends
- Implement adding user's friends

Task 5: Implement Deleting Users Friends

- Build UI For deleting user's friends
- Implement deleting user's friends

Task 6: Implement Calling Users for Video Chatting

- Build UI for video call between users
- Implement video call between users

Task 7: See app user's info

- Build UI for visualizing own user's info
- Implement visualizing own user's info

Task 8: See friend's info

- Build UI for visualizing friend's info
- Implement visualizing friend's info