App Demo https://pr.to/AGRBAG/

GitHub Username: ijzepeda

FriendsKnow

Description

A game that will show cards with events that players need to vote for which of their friend is most likely to be like that description.
The game can be played offline or online. Online mode is intended to be played on different times. Not necessarily play the whole session at once.

Intended User

People that have open mind, great sense of humor and willing to spend good time with friends. For mature minds. Or grownups with immature mind are the best users.

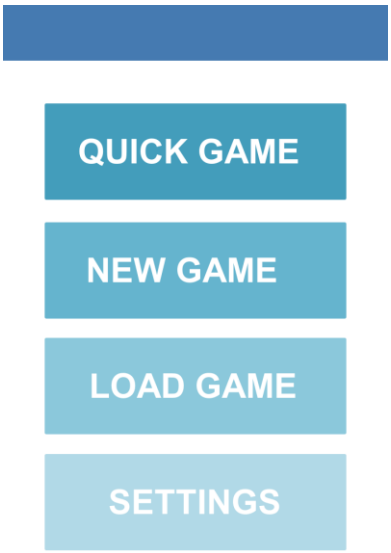
Features

- Access Internet
- View Contacts
- Use Microphone
- Online synchronization with backend API.

User Interface Mocks

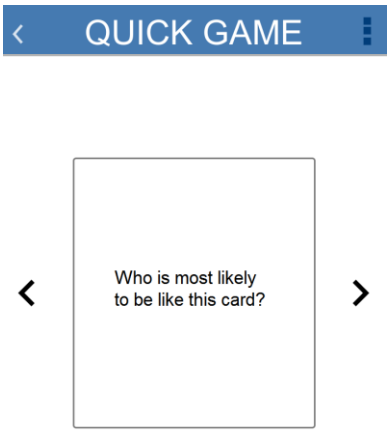
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

Screen 1: MainScreen



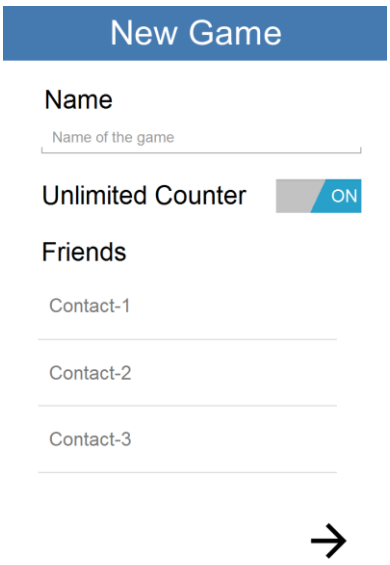
MainScreen will show different type of games: Quickgame for Offline mode, and NewGame and Load Game to have an online game.
Settings to modify the overall app settings.

Screen 2: QuickGame



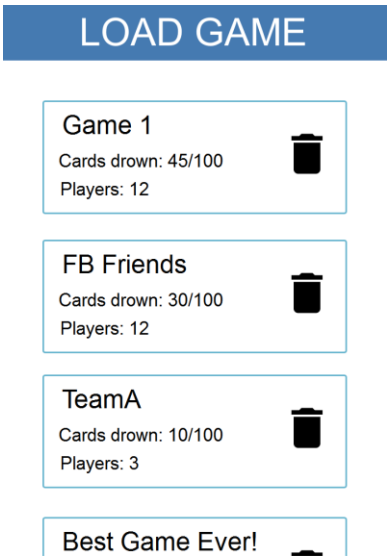
Quick Game: is an Activity that will show a card from a deck. Able to go back and forward for all cards on the deck.

Screen 3: NewGame



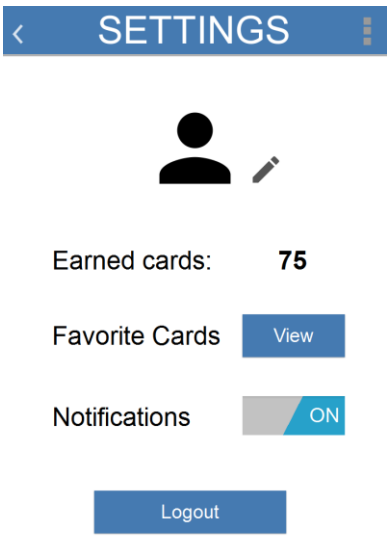
New Game: create a game to play with friends

Screen 4: Load Game



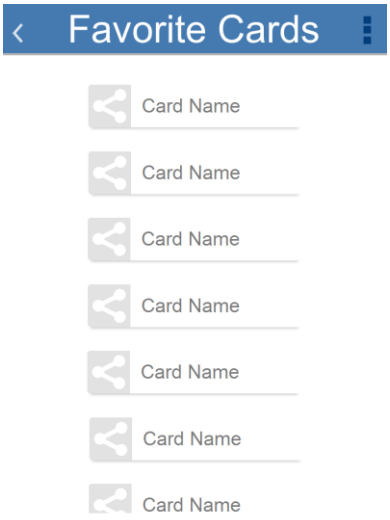
Load Game: Will show all your games where are currently active. Choose from the list to continue playing.

Screen 5: Settings



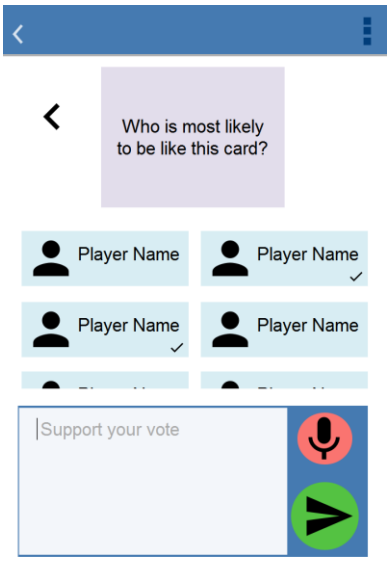
Settings: Is an Activity that displays basic information of the user. View Cards, Favorites, and activate notifications.

Screen 6: View Favorite Cards



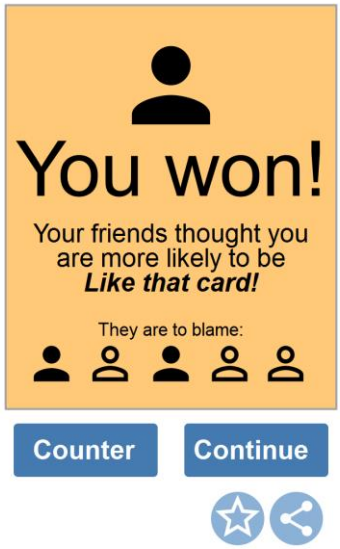
Favorite Cards: is an Activity that will have a recyclerview with all the saved cards, and can be shared over different apps or social networks with personalizes texts, an image to the card and a link to get the app.

Screen 7: Main Game: Vote



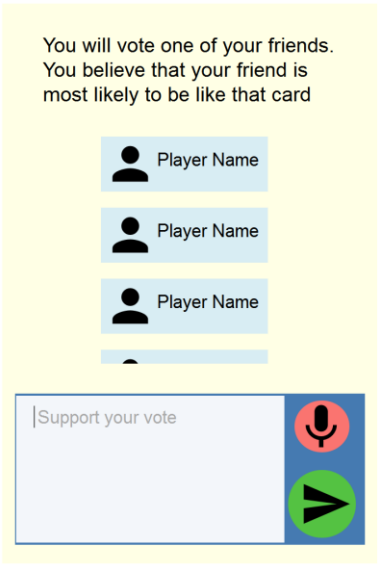
MainGame: is an Activity that will have a card from the deck. Just current and older cards are visible. After reading the card, the user Selects one of their friends that think that is the most likely to be like the card. If the User wants a comment or audio note can be ‘attached’ to the vote. Only when all users have voted the screen will switch to the results.
Players can leave the screen and be notified by a Notification when the result is ready.

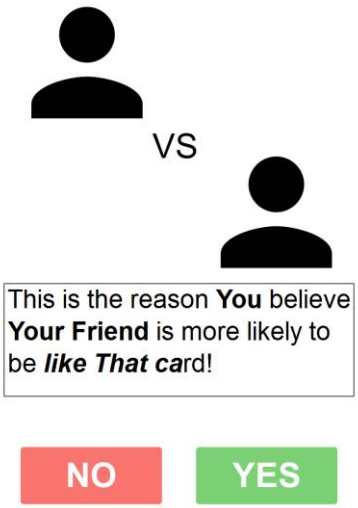
Screen 8: Results

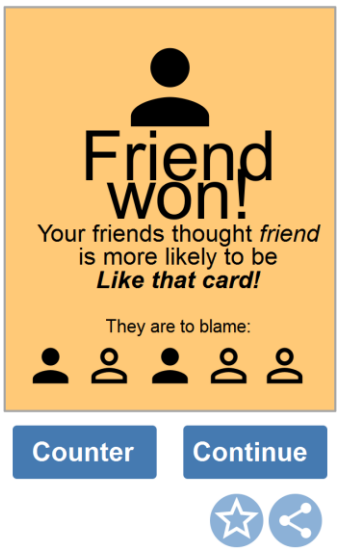


Results: This Activity will show the winner (Or sometimes loser). A list of players that voted for the winner and (if available) the comments. Also two more buttons allow the user to save the card (Only for the winner) and Share, for all players.

Optional Features: Counter







Counter Feature: The winner can select another player that might be a better match for the card. The argument is shared with the rest of the players and if majority votes positive to it, the selected player becomes the winner (loser). The counter function can be unlimited unless deactivated in settings.

Key Considerations

How will your app handle data persistence?

SQLite, Content Provider, Network, SharedPreferences

Describe any corner cases in the UX.

For QuickGames, User only be able to access to It from main menu, same for New Game.
Load a game, will open the selected game. While tapping a notification will directly open the selected game.
In the App Lifecycle will exist an ActionBar that displays the current Activity and a back button.

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

SwipeDeck, Picasso, OkHttp.

Describe how you will implement Google Play Services.

Admob: create different flavors with and without ads.
Analytics: keep track of its usage with users.
GCM: for Notifications, and PushNotifications*
FireBase: Backend API

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

- Configure libraries: Swipe-Deck, OKHTTP
- Create the Activities for each screen.
- Create an Utils class to manage methods for connections, Swipe Deck, Settings, and UserProfile.
- Add all dependencies
- Modify manifest for required permissions

Task 2: Implement UI for Each Activity and Fragment

- Build UI for each Activity
- Create Custom Views and use appropriate Views for SwipeDeck
- Build UI for Widget

Task 3: User Profile Creation

At the beginning of the app, when it first run, it will prompt to the users for some details:

- Phone number
- Name
- Email
- Password

Task 4: Notification

- Receive Push Notifications from Server
- On tap open the app on selected game
- Create a server to fetch status of games every few hours, in case push notification failed to received the user.

Task 5: Widget

- Create layout
- Show status of games: Last game played, or latest notification.

Task 6: Activities

- Work with libraries to display the game
- Add appropriate listeners for each action
- Use AsyncTask or OKHttp to keep game synched.
- If using microphone, ask for permisions and convert the audio file to bas64 (Doing so, the values sent are just strings or primitives)

Task 7: Backend

Backend will manage the databases, Users, Games and Notifications

Task 8: Add Resources and Accesibility options

- String to Res Folders
- LTR, RTL
- Accessibility Options

Task 6: Gradle

- Add all dependencies required
- Create flavors
- Add API KEY for Google Services

CONSIDERATION
As this app have multiple functions, I may not be able to get all done by the deadline. Therefore I substracted some features from it, such as: counter (when a player ask for a rematch) of add friends using Facebook.

App Demo: https://pr.to/AGRBAG/