GitHub Username: omarahmed0193

Let’s Hang

Description

Let’s Hang makes the process of getting together with friends incredibly easy by giving the invited friends two simple options for a given invitation either “i’m in” or “i’m out” and with a built in group chat and easy navigation action to the selected place getting together never been simpler.

Intended User

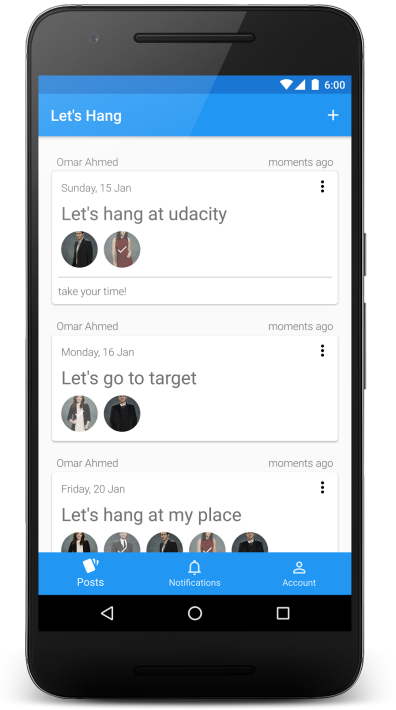
This app is for people who wants to hangout with their friends and want a simple way of communication.

Features

- Saves and syncs users data
- Built in group chat
- Suggest places near you to hangout at
- Offers social/email login and sign up
- Offer to set reminders and notify the user for a chosen post time
- Shows notifications for friends requests and new posts

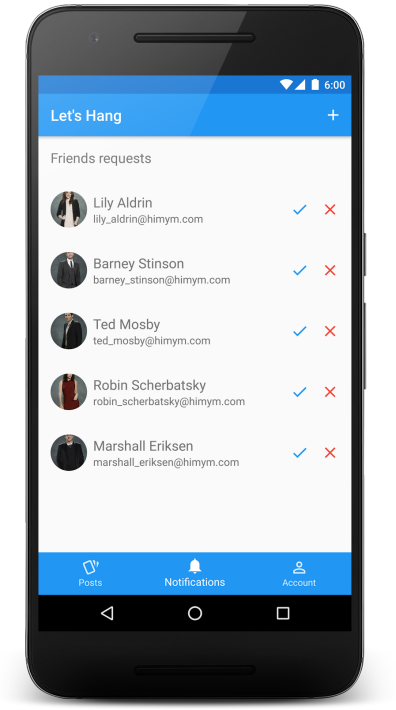
User Interface Mocks

Screen 1



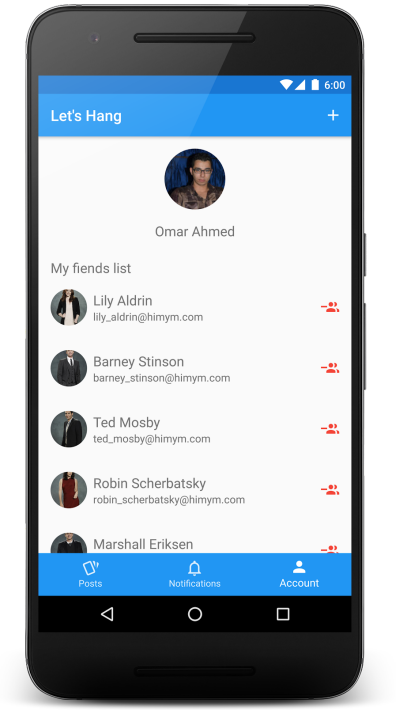
This screen has all the posts that belongs to signed user with each post providing info about the invite.

Screen 2



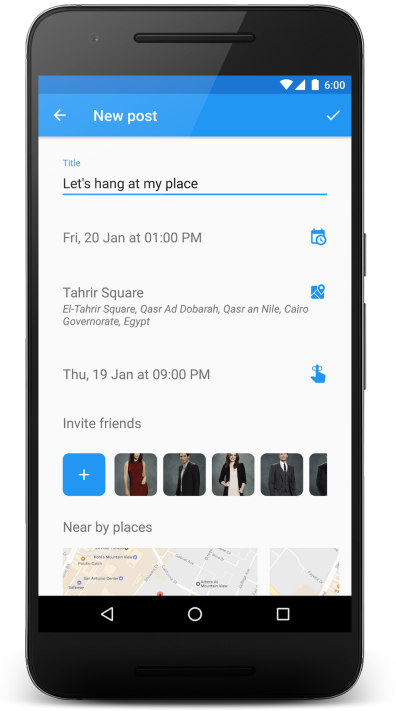
This screen has the user notifications such friends requests where the user can accept or reject the request.

Screen 3



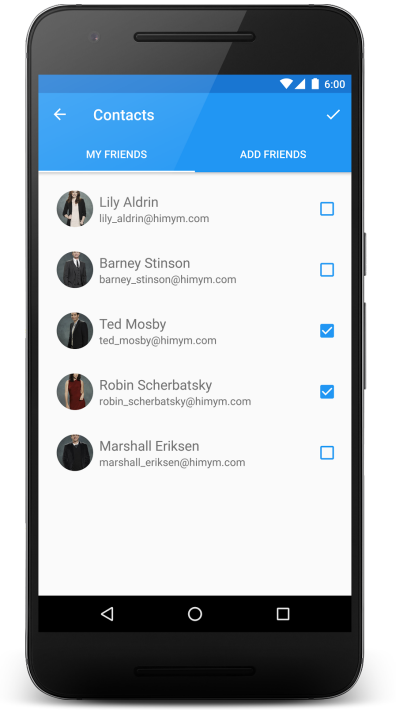
This screen has the user info in addition to his friends list where he can take actions on them such as removing them from the list.

Screen 4



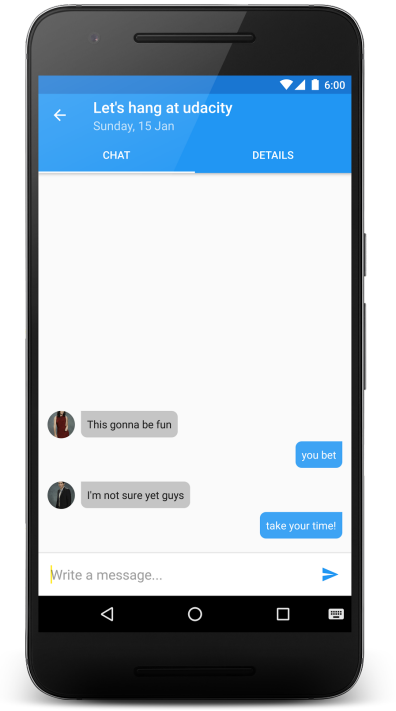
This screen lets the user add new posts with the demonstrated attributes in the screen.

Screen 5



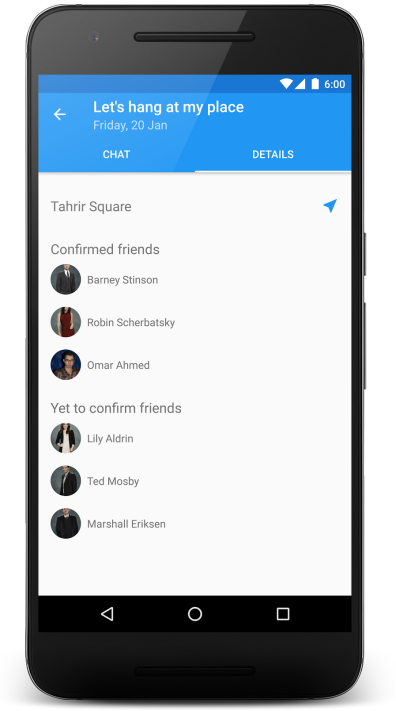
This screen lets the user invite his friends to a new post.

Screen 6



This screen has a group chat for a specific post.

Screen 7



This screen has the post details where the user also can navigate to the post place if selected.

Key Considerations

How will your app handle data persistence?

The app will use Firebase for saving and syncing the user's data with the persistence option enabled to make these data available offline, And in order to meet the project requirements a content provider will be used to save the the user's reminders for the posts created by him.

Describe any corner cases in the UX.

There is only one corner case which the actions in the notification and for this case a BroadCastReceiver will be created to handle the necessary actions, Other than that there is no corner cases in the UX.
The design is straightforward and the transitions from one screen to another is seamless.

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

- Fresco to handle the loading and caching of images.
- SublimePicker to handle date selection.
- Retrofit to handle HTTP requests.
- EasyPermissions to handle the needed permissions.
- ButterKnife to easily reference views.
- MaterialDialogs to handle dialogs creation.
- FirebaseUI to easily populate and load data into views.

Describe how you will implement Google Play Services.

The location API will be used to get the user location in order to use in congestion with the Places API to get the nearby suggested places or with a place picker to let the user select a specific location on the map.

Task 1: Project Setup

- Create project in Android Studio
- Configure min and target API level

- Configure libraries
- Create project in Firebase
- Link project with Firebase

Task 2: Implement UI for Each Activity and Fragment

- Build UI for Login activity
- Build UI for Main activity and its related fragments
- Build UI for Post's details activity and its related fragments
- Build UI for Friends activity and its related fragments

Task 3: Implement application code and logic for each UI element

Write code for the login screen:

- Create layout
- Enable social and email login firebase
- Implement each type of login
- Write code for caching the user data to firebase

Write code for main screen:

- Create layout for the activity
- Create layout for each fragment
- Implement each fragment code and logic

Write code for the new post screen:

- Create layout
- Implement location service to get user location
- Implement place service to use the place picker and get nearby places
- Implement the action to invite friends to the post

Write code for the post details screen:

- Create layout
- Implement the chat fragment
- Implement the details fragment

Task 4: Implement FCM

Write code for the Android client side:

- Create the instance id service

- Create the messaging service
- Build the notification to show to the user

Write code for the server side for FCM using Node.JS

Task 5: Clean and build up the project

Clean up the code:

- Remove literal strings and logs
- Ensure the accessibility features are implemented correctly

Build up the code:

- Create a keystore
- Build a release apk