GitHub Username: ayman

When I’m There

Description

Have ever been thinking while waiting at the cashier line trying to remember the things you needed two days ago at home? Have you stood in front of the pharmacy shelf wondering about the things missing at home? Are you tired of calling home every time to ask if they need anything from the bakery?
With this app you can add items when you need or remember them, and open the app when you’re at the market to get those stuff. You can then mark them as done just like a todo list. You can share your list with other people either to help you gather the items or put items that they want you to bring.
With this app you can safely call home and tell them: “I’m not calling everyday anymore. Put the stuff you need from the market in the app and I’ll will get it When I’m There”.

Intended User

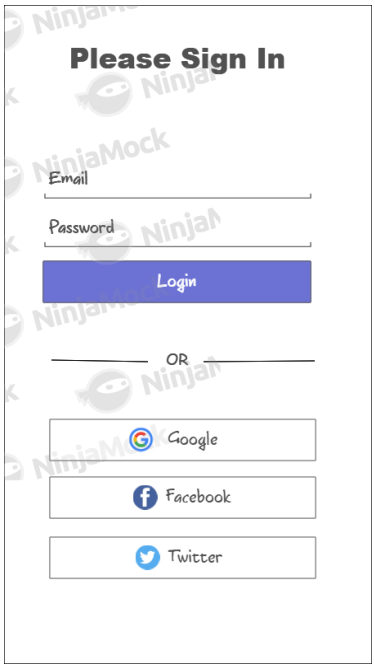
Anyone who shares the place you live in, work at or hangout around. if you’re a lonewolf you can benefit too.

Features

- Saves information.
- Organizes Tasks.

User Interface Mocks

Screen 1



First launch Sign In Menu

Screen 2

The user is in their own list. Specifically viewing the items that they need to get.

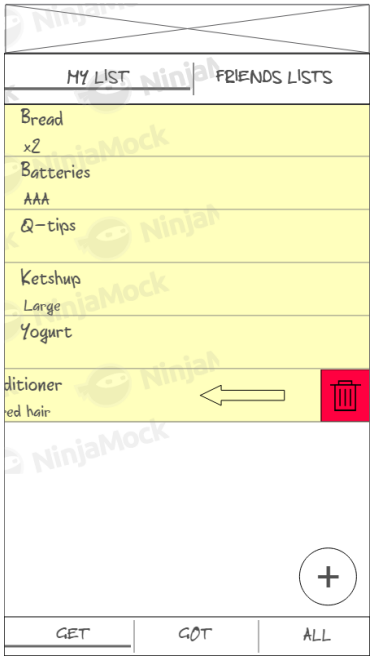
Screen 3

Items can be added by clicking the Floating Action Button and they will be presented with a dialog to enter the item details.

Screen 4

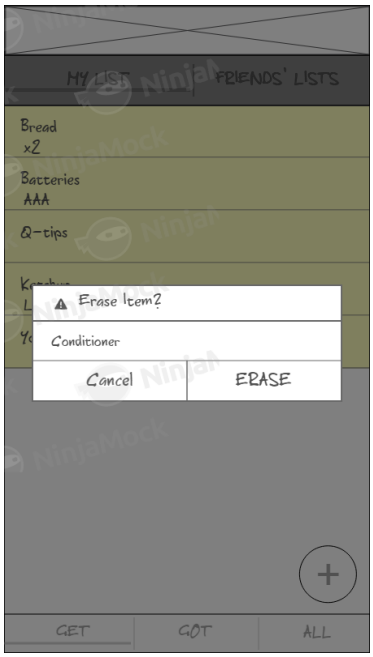
Clicking on an Item will show the user a dialog to edit the item details.

Screen 5



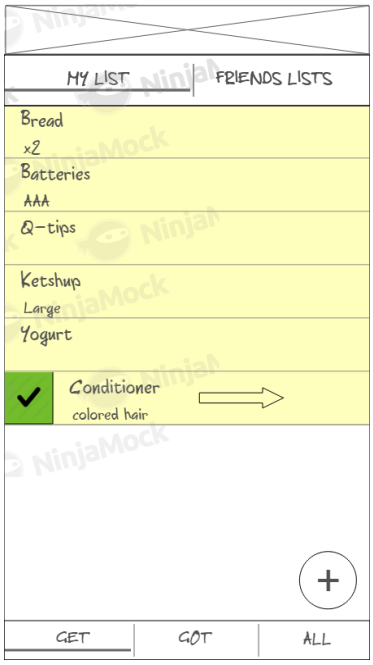
Swiping left on an item will delete the item.

Screen 6



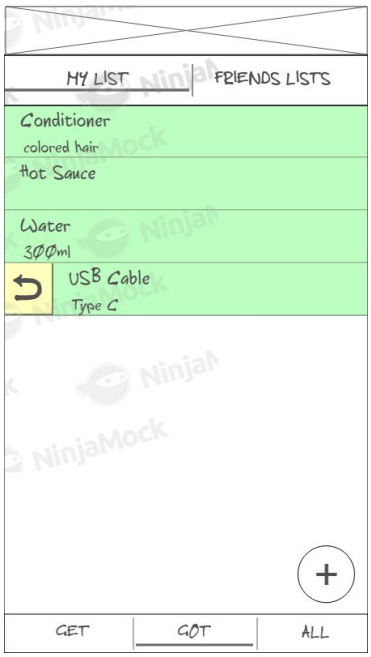
A prompt dialog will appear to confirm deletion.

Screen 7



Swiping right on an item will mark it done and move it to the Got list shown in Screen 8.

Screen 8

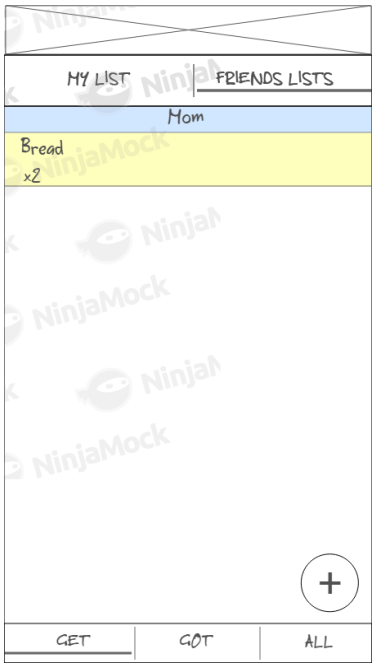


In the Got list, swiping right will put the item back in the Get list. Swiping left will delete the item.

Screen 9

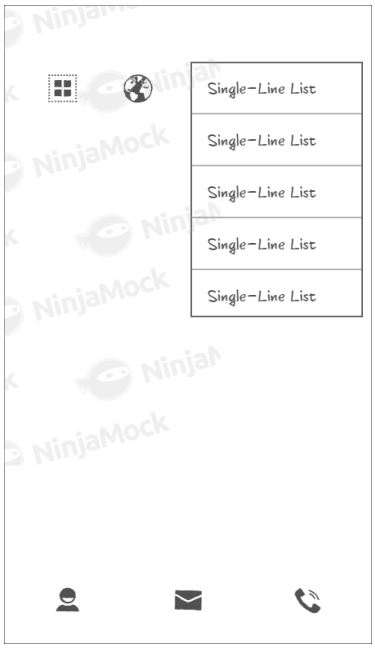
The friends list menu. Clicking on a friend's name will open their list shown in Screen 10.

Screen 10



A friend’s list acts and looks exactly like the user’s own list except for the blue bar at the top showing the friend's name.
Notice: the top area crossed is for the ad.

Screen 11



On the home screen the widget will look something like shown. The list shown is the list shown in the Get menu which are the items you need to get. It is a scroll list so the user can scroll up and down to see more items. Swiping right or left for deleting and marking Got will be tested. If it doesn’t interfere with the regular home screen swiping in terms of UX, it will be implemented.

Key Considerations

How will your app handle data persistence?

Firebase Realtime Database.

Describe any edge or corner cases in the UX.

Pressing back while on a friend's list will take the user to the Friends List menu.

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

- ButterKnife to inject and map views. It will be used to avoid boilerplate code of creating views like: findViewById(R.id.text_view).
- Gson to handle parsing json strings in the development phase.
- RecyclerView to display lists.

Describe how you will implement Google Play Services or other external services.

AdMob will be used to display ads.

Next Steps: Required Tasks

Task 1: Project Setup

- Create a new project and configure version settings like targeted and min SDK.
    - Android Studio 3.4.2
    - Android Gradle Build: 3.4.2
    - Gradle version: 5.1.1
- Configure libraries that will be used in the project.
    - ButterKnife: com.jakewharton:butterknife:10.1.0
    - Gson: com.google.code.gson:gson:2.8.5
    - Firebase: com.google.firebase:firebase-core:17.0.1
    - RecyclerView: com.android.support:recyclerview-v7:28.0.0
- Add internet and network permissions.
- Set Up a firebase account and database.

Task 2: Implement UI for Each Activity and Fragment

- Build UI for the login activity.
- Build UI for MainActivity.
- Build UI for the list fragment.
- Choose a color theme and create styles.xml and colors.xml files.

For accessibility support all elements in the UI will have the attribute ContentDescription filled with the name of the Item and the font and buttons can have bigger sizes.
Colors will have high contrast and queues other than color will also be used.
Accessibility Scanner will be used to investigate further features.

Task 3: Implement the main functionalities of the application.

Write the java code that connect the activities, fragments and layouts along with the database source.

- Use an AsyncTask to check for internet connectivity while viewing an indicator like a spinner.
- Connect the MainActivity to Firebase or prepare a test Json string.
- Create the Implementation of displaying a list of items.
- Create the Implementation of adding, deleting and editing items.
- Create the Implementation of changing the state of an item.

Task 4: Implement the main features of the application.

Allow the possibility for list sharing and connecting with friends.

- Adjust the design of the database to handle a list of friends for each user.
- Adjust the code to function and view list properly and correctly.

Task 5: Create the widget implementation.

- Create layout of the widget.
- Create the implementation of displaying items to get in the widget provider activity.

Task 6: Implement AdMob and setup signing configuration.

- Add a view for AdMob, request and load the ad.
- Sign and build the APK.