[Description](#_sm4ra97uwo11)

[Intended User](#_aws88pzfmqca)

[Features](#_zheq5430xrpq)

[User Interface Mocks](#_r1e9j2sbgaxv)

[Phone - Main](#_fxxaekdggsie)

[Phone - Main - Add Note](#_rfzvqk1nodbs)

[Phone - Navigation Drawer](#_v2lzbjm2i1uj)

[Phone - Saved Lists](#_cvua8j7sof4)

[Tablet - Main - Add Note](#_n1p2ip2crsjd)

[Tablet - Home Screen Collection Widget](#_y6ti8khj9jyv)

[The collection widget allows a user to readily view items that have not been packed yet (checked off) along with their quantities. Tapping the widget will open the app and display that specific packing list.](#_2l6hcl1fcy7m)

[Key Considerations](#_gvcvmae8jn8u)

[How will your app handle data persistence?](#_v8my7nhtvz0m)

[Describe any corner cases in the UX.](#_gw69vjn1ico0)

[Describe any libraries you’ll be using and share your reasoning for including them.](#_6yqqubmw5bs)

[Describe how you will implement Google Play Services.](#_qrxg682nywe6)

[Next Steps: Required Tasks](#_v518bncmggeg)

[Task 1: Project Setup](#_8oe8zpk3qsmp)

[Task 2: Implement UI for Each Activity and Fragment](#_rzllsk6uqztx)

[Task 3: Implement Firebase](#_fdmohs7hes)

[Task 4: Integrate Google Calendar API](#_rfe44vxqcj6g)

[Task 5: Home Screen Collection Widget](#_umfwsvmx7tpn)

[Task 6: Testing](#_1vi0vqeilppr)

**GitHub Username**: alardizabal

Pack Overflow

# Description

Never again waste time trying to remember what to pack for your trip! Whether you’re going on a beach vacation, a business trip, or a weekend getaway, Pack Overflow lets you save and recall packing lists for every occasion. Do you run down a mental list of items each time you pack but always forget your underwear? Going on a trip with friends? Share your list with them and eliminate the hassle and stress of preparing for your trip. As you pack items on your list, tap each item to check them off so there’s no need to double or triple check your baggage.

# Intended User

This app is for anyone who has spent time trying to recall mental packing lists. Even written lists are difficult to edit and are easily lost. Pack Overflow can help college students pack for school or trips home, businesspeople pack for regular trips, and vacationers who are packing for a fun time!

# Features

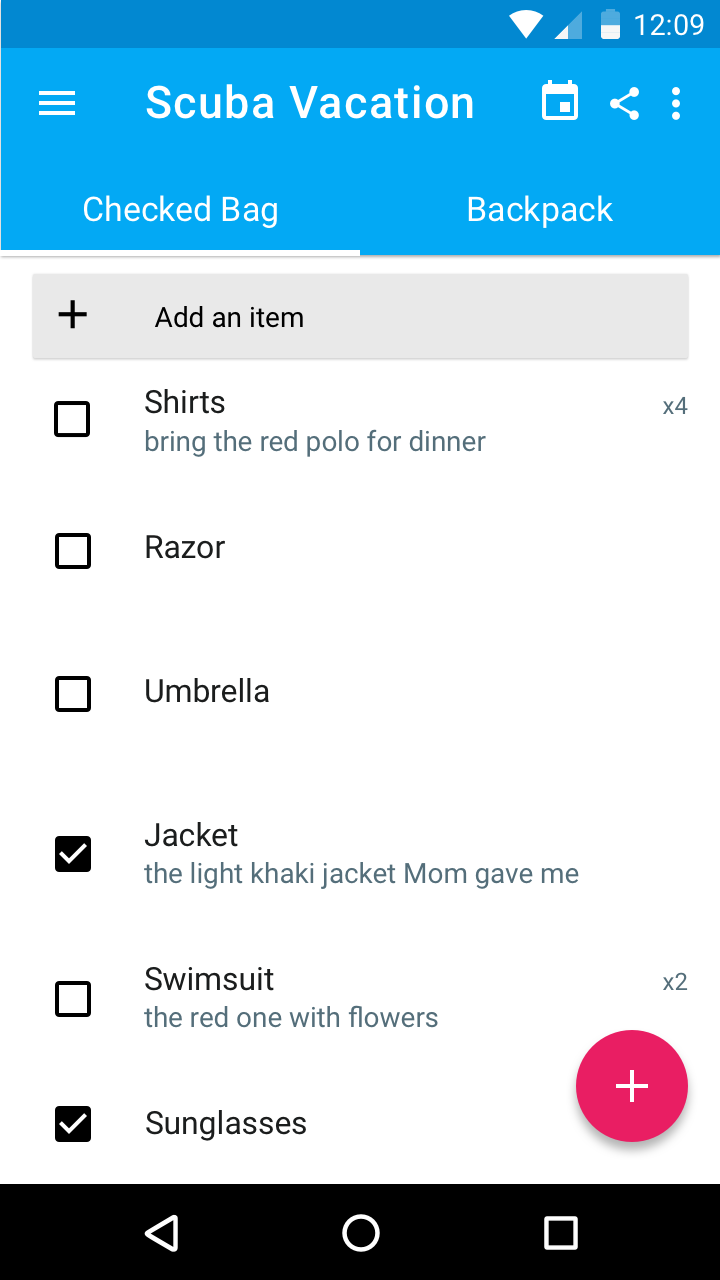
* Create packing lists with custom names based on your trip type
* Add a quantity for each item
* Add a widget to your home screen displaying items you haven’t packed yet
* Share lists with friends
* Sync data across multiple devices using Firebase’s real-time database
* Swipe to delete packing list items in the app’s main interface.

# 

# 

# User Interface Mocks

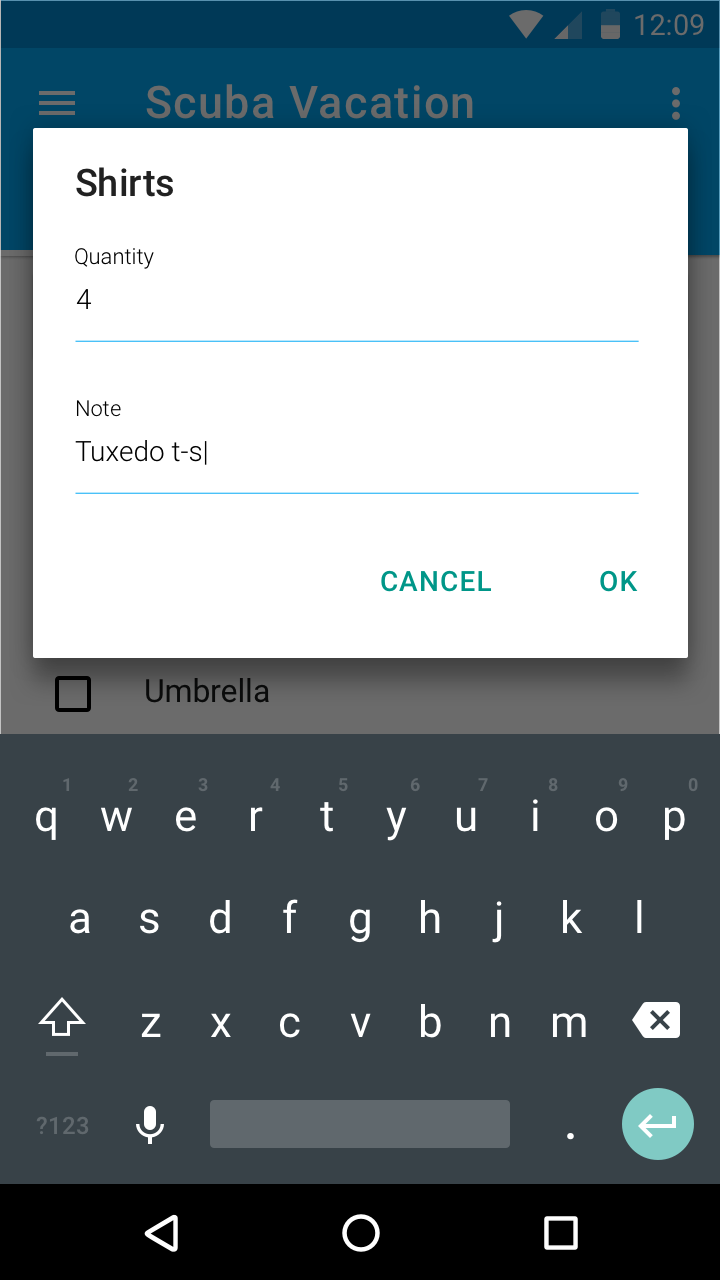
## Phone - Main



This is the main screen for phones. This screen will display the user’s selected packing list. There will be a text field that will allow users to easily add items without having to transition to another screen. As users pack the selected items, they can be checked off. Tapping an item in the ListView will display a dialog allowing a user to edit quantity and add a note.

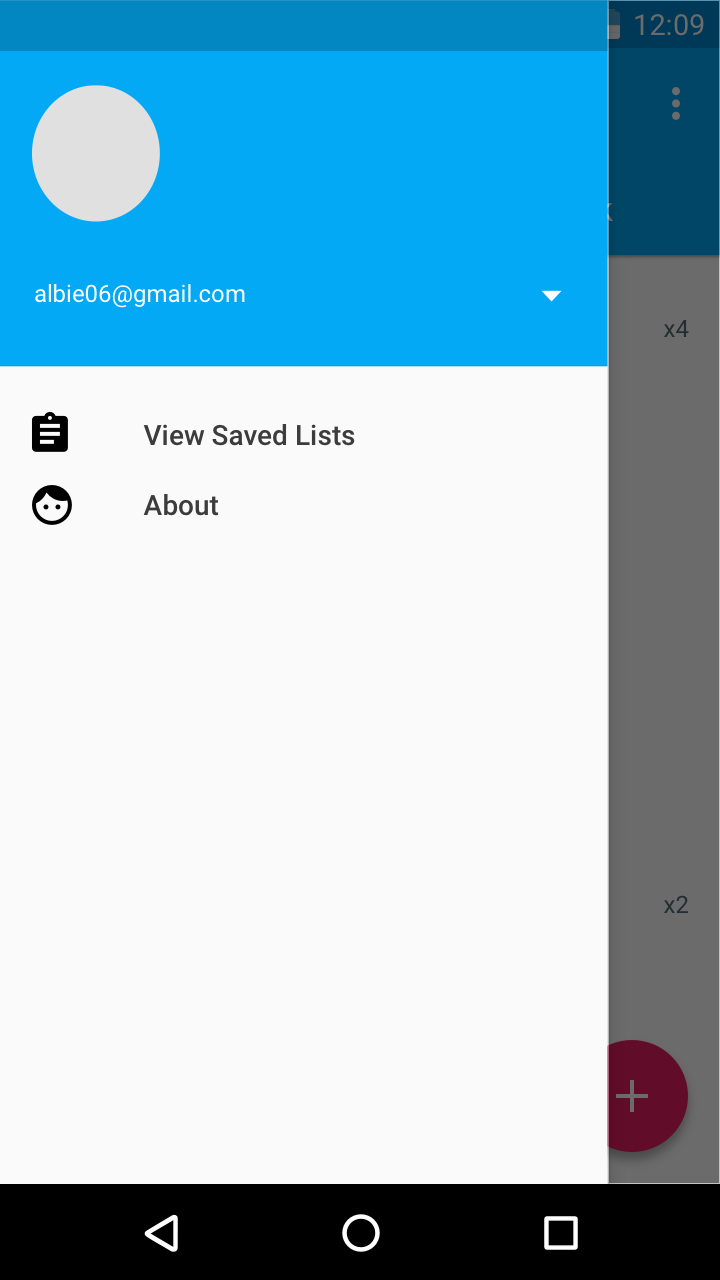
Tapping the calendar icon will open a dialog allowing users to save the date of their trip to their calendar using the Google Calendar API. Users will be able to share the current packing list from this screen. Items in the ListView can be deleted by swiping the view. The floating action button will present options to create a new “bag” in the current packing list or to create a new packing list.

## Phone - Main - Add Note



This is the dialog that is presented to the user when tapping on an item in the ListView on the main screen. The user will be able to edit the quantity of the item to pack and will also be able to add a note that will be displayed underneath the item’s title in the main packing list view.

## Phone - Navigation Drawer



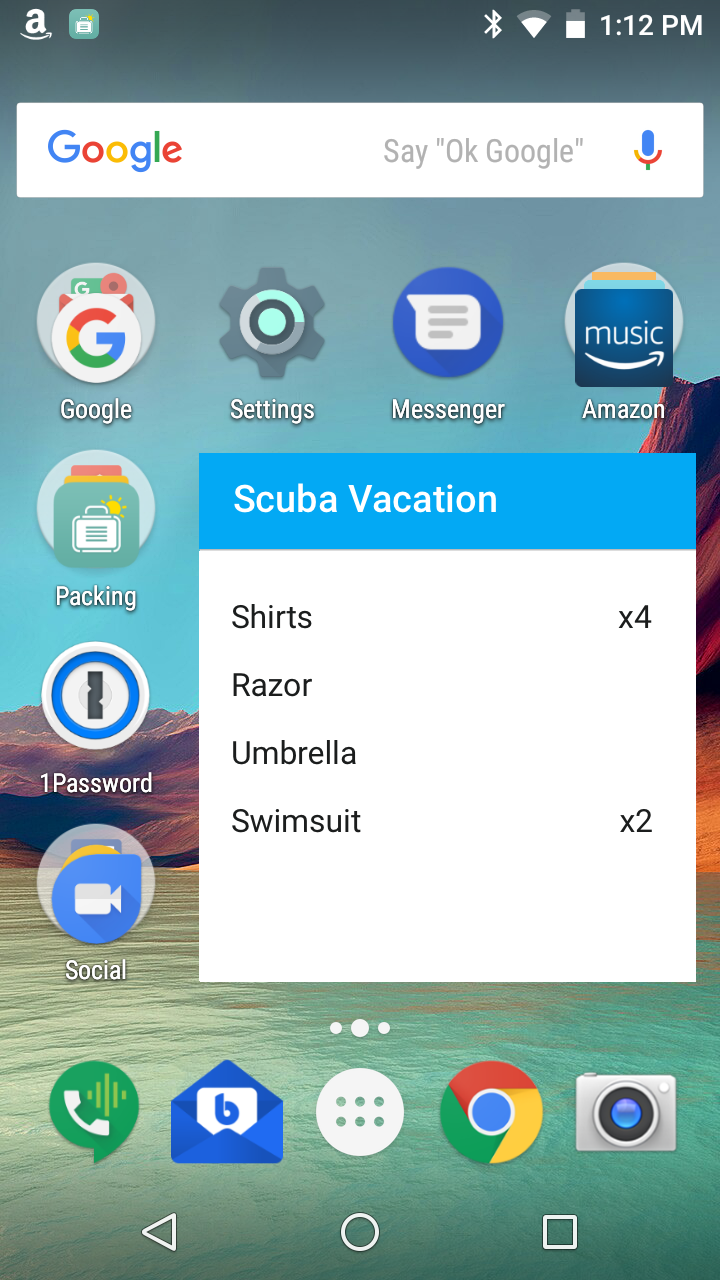
The navigation drawer will present extra options. Users can log in, log out, or switch to another account. The option to view all of the user’s saved lists will be available in the drawer.

## Phone - Saved Lists

## Phone - Saved Lists.png

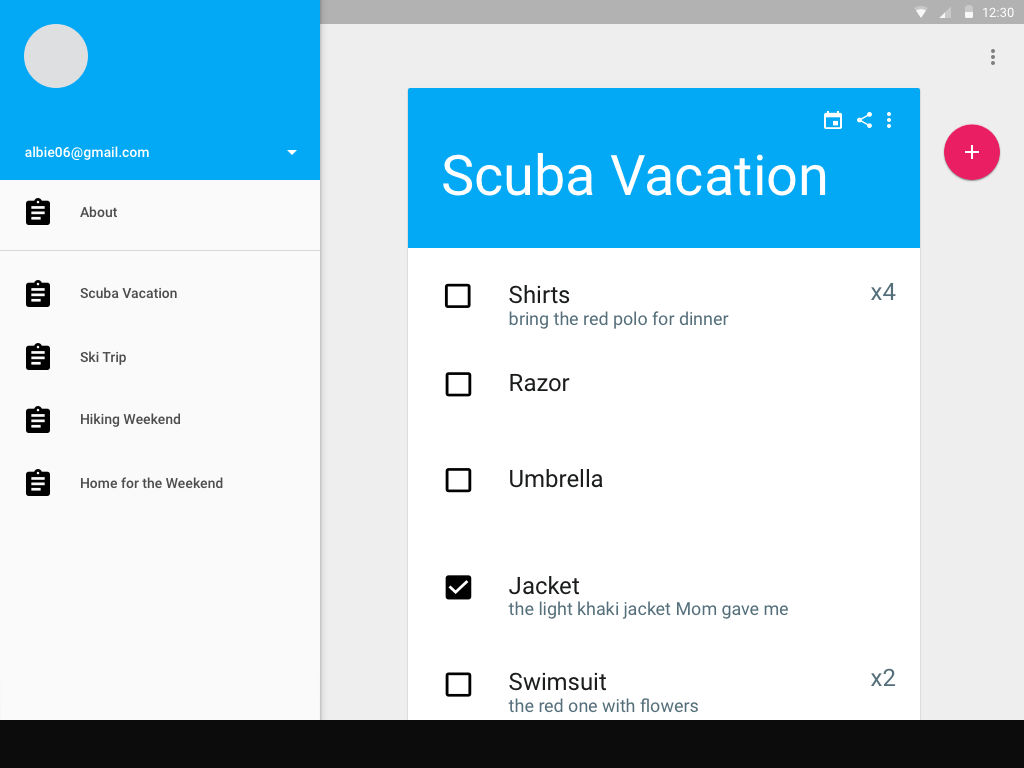
This screen is accessed by tapping on the “View Saved Lists” option in the navigation drawer. The overflow button will give users the ability to add new lists. Users will be able to delete saved lists by swiping the item’s view. Tapping on one of the views will load that packing list into the main screen and the “Saved Lists” view will be dismissed. The app will have pre-installed lists stored locally on the device that users can select and customize.

**Phone - Home Screen Collection Widget**



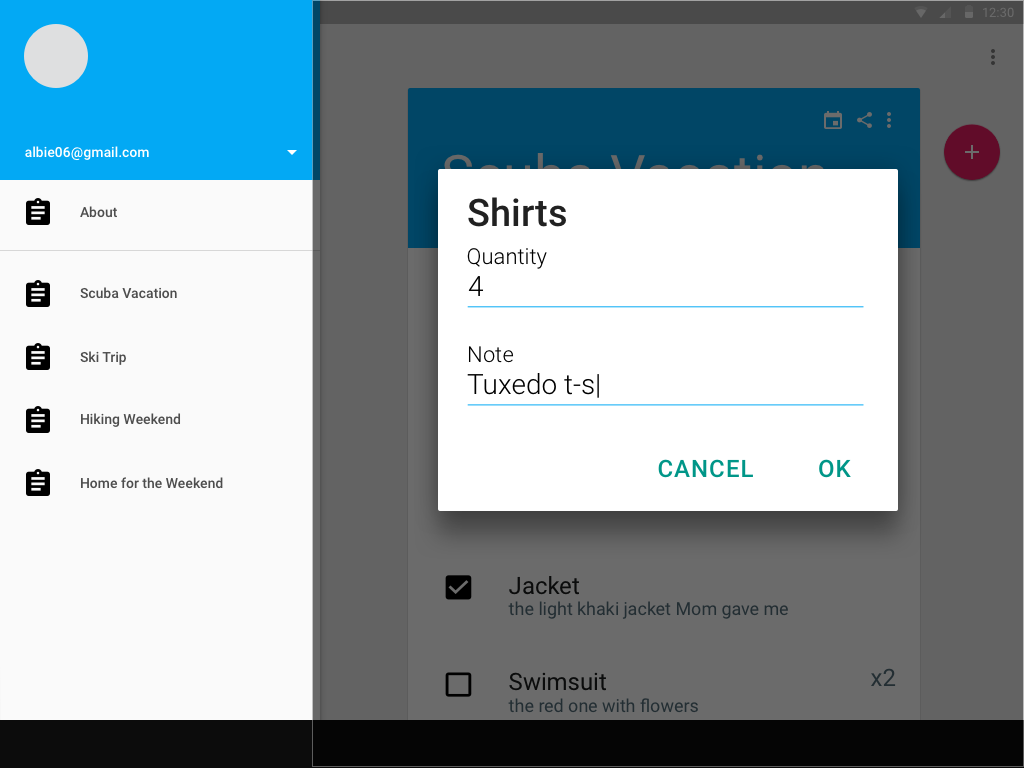
The collection widget allows the user to view items that have not been packed yet along with their quantities. Tapping the widget will open the app and display that specific packing list.

Tablet - Main



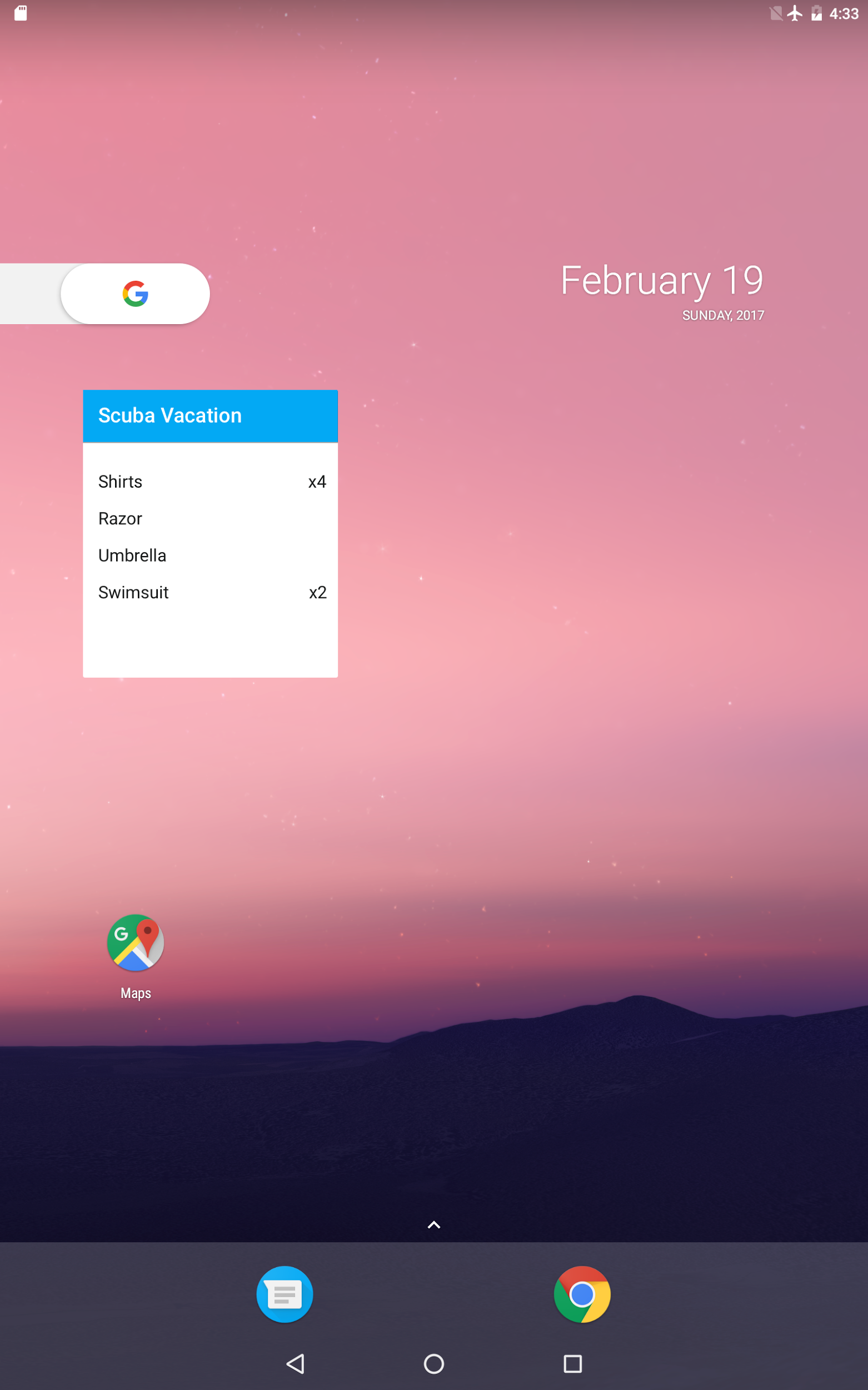
For the tablet layout, users will be able to see both the saved lists available to use and the detail of the currently selected packing list. Users can log in, log out, or switch accounts using the header at the top of the navigation drawer. Tapping any of the lists in the navigation drawer on the left will cause that list to become the currently selected packing list. The detail view will reload with that selected list. The packing list will function similarly to the phone version. Users can delete the packing list item by swiping the view. The share button will be implemented to share the current list of items. The calendar icon will allow a user to save the date of their trip to their calendar using the Google Calendar API.

## Tablet - Main - Add Note



This dialog will be presented when tapping on a packing list item in the detail view. Similar to the functionality in the phone layout, users will be able to edit the quantity and add a note for the selected item.

## Tablet - Home Screen Collection Widget



The collection widget allows a user to readily view items that have not been packed yet (checked off) along with their quantities. Tapping the widget will open the app and display that specific packing list.

# 

# Key Considerations

### How will your app handle data persistence?

Firebase’s real-time database will be used to save and sync users’ lists across multiple devices. When the user is offline, their data will be saved locally using Firebase and will be synced up remotely when their internet connection returns. The app will come loaded with pre-made packing lists that users can customize. They will be stored in a SQLite database and the data will be loaded into the main ListView using a CursorLoader.

### Describe any corner cases in the UX.

Even simple lists would benefit from a variety of editing features. An item in a list could be quickly removed by utilizing swiping to delete the item. There are also cases when individual items need to be prioritized so dragging items to desired positions within the ListView would be needed.

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

**Butter Knife** <http://jakewharton.github.io/butterknife/> - Allows me to easily bind views to fields and write more concise code. It will eliminate the need to constantly cast views to the proper class.

**Gson** <https://github.com/google/gson> - I can easily map JSON representing saved packing lists to models. The interface for serializing to JSON and de-serializing JSON seem to be the easiest to use versus other third-party libraries despite being slower.

### Describe how you will implement Google Play Services.

**Firebase Analytics** - Analytics will be used in the app to track a variety of user interactions with the app. Custom events will be implemented to track when lists are added, edited, and shared.

**Firebase Realtime Database** - Firebase’s real-time database will be used to save users’ lists and sync across devices. Lists will be stored in JSON format and data will be persisted to disk if the user is offline. When an internet connection is reestablished, the locally stored data will be synced with the remote database.

**Firebase Authentication** - Authentication will be used to enable users to log in and save and access their packing lists. Users will be able to sign in using their Google email credentials.

# Next Steps: Required Tasks

## Task 1: Project Setup

1. Create a new project called PackOverflow
   1. Minimum SDK 15
   2. Target SDK 25
2. Configure third-party libraries
   1. Add ButterKnife to project gradle - <http://jakewharton.github.io/butterknife/>
   2. Add gson to project gradle - <https://github.com/google/gson>
3. Sign up for a Firebase account - <https://firebase.google.com/docs/android/setup>
   1. Add Firebase core dependency to project gradle
   2. Add analytics dependency to project gradle
   3. Add realtime database dependency to project gradle
4. Create Signing Configuration
   1. Include keystore and password in repository

## Task 2: Implement UI for Each Activity and Fragment

1. Build UI for MainActivity
   1. Extend AppCompat
   2. Create a tab bar so the user can switch between different “bags” for the items they need to pack.
   3. Create a ListView that will contain the packing list items.
   4. Create view for packing list items - Include checkbox, title label, note label, and quantity label.
2. Implement functionality for MainActivity
   1. Implement floating action button - ability to create a new packing list.
   2. Implement sharing feature so packing lists can be shared with friends.
   3. When tapping into a ListView item, a dialog will be displayed allowing the user to enter a quantity and an optional note for the item.
3. Build UI for Navigation Drawer
   1. Implement transition when tapping on the hamburger menu in the MainActivity.
   2. Navigation drawer will display menu items. The user will be able to load saved packing lists.
   3. Create a header view that will contain information about the user. Information will be retrieved from the Firebase Auth process.
4. Build UI for SavedListsActivity
   1. Create a ListView to display saved packing lists.
   2. Implement functionality when tapping a ListView item. The user will be returned to the main screen and the items in the selected packing list will be loaded into the ListView.
   3. Ready-made packing lists will come pre-installed with the app. The lists will be stored in a SQLite database and will be loaded into the app’s main ListView using a CursorLoader.

## Task 3: Implement Firebase

1. Set up Firebase Analytics
   1. <https://firebase.google.com/docs/analytics/android/start/>
   2. Create custom events for creating and deleting packing lists and items.
   3. Create custom events for sharing packing lists.
2. Set up Firebase Authentication
   1. <https://firebase.google.com/docs/auth/android/start/>
   2. Implement Google sign-in.
3. Set up Firebase Real-time Database
   1. <https://firebase.google.com/docs/database/android/start/>
   2. Structure data models - for saved packing lists and items.
4. Set up Firebase crash reporting
   1. <https://firebase.google.com/docs/crash/android>
   2. Intentionally cause error and check logs.

## Task 4: Integrate Google Calendar API

1. Activate Google Calendar API in Google Developer Console
2. Update project gradle for Google Play Services and Calendar dependencies
3. Subclass AsyncTask to make the Calendar API background calls to access and update the user’s calendar.

## Task 5: Home Screen Collection Widget

1. Create ContentProvider
2. Declare an app widget in the manifest.
3. Create the app widget layout
4. Extend AppWidgetProvider
5. Extend RemoteViewsService
6. Implement RemoteViewsFactory

## Task 6: Testing

1. Test offline functionality and ensure packing list data sync once internet connection is re-established.
2. Ensure tablet layout is correct and consistent with phone experience.
3. Test rotation and ensure state is preserved.