GitHub Username: jQN

Busy Mama

Description

Busy mothers assistant budget app written solely in the Java programming language - uses geolocation to check if the mom was near a store and reminds her to note how much she spent in the store. The data can be stored and used later to balance the budget.

The problem this is trying to solve is how busy mothers are so they're not always able to create a budget or to keep track of their spending.

Intended User

The intended user is mothers but could also be used by anyone interested in keeping track of their expenses.

Features

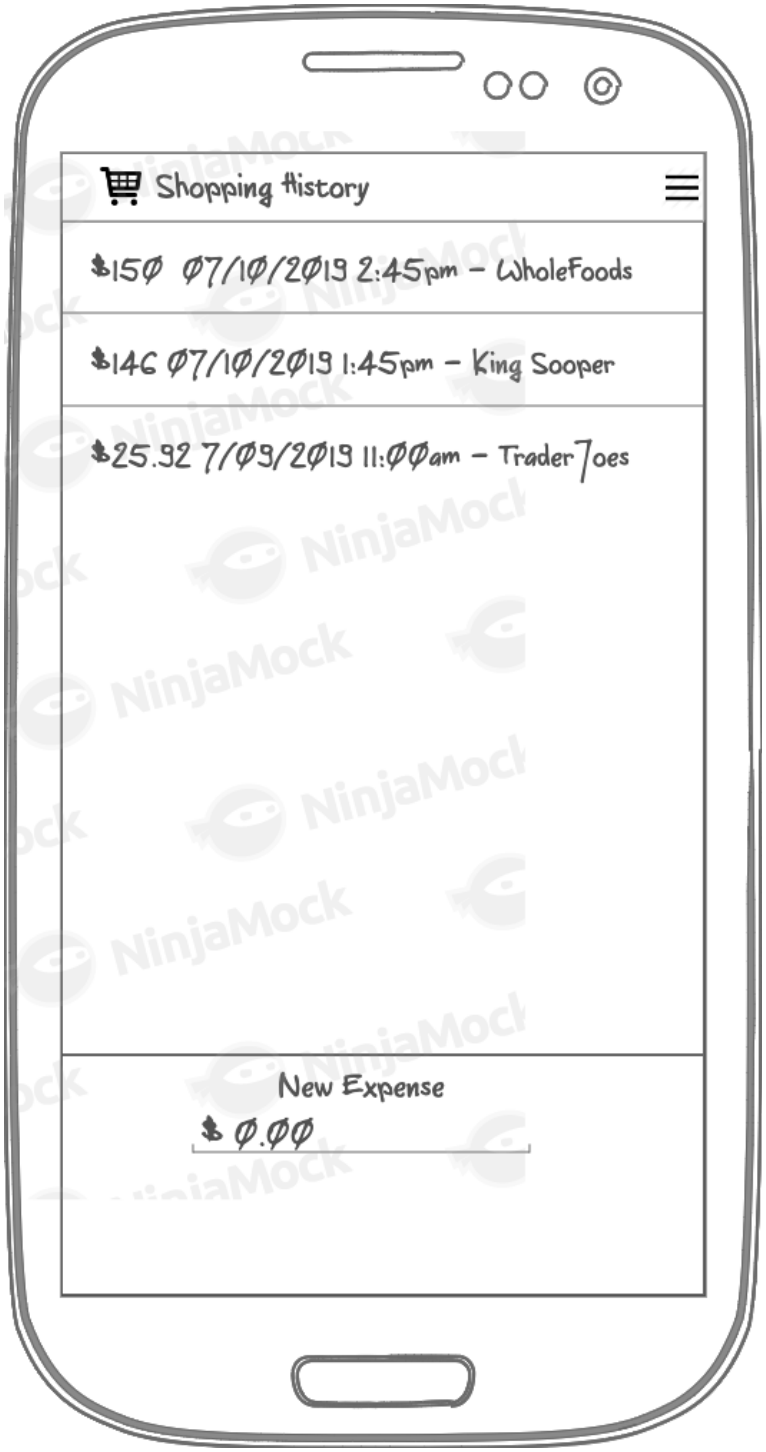
Main features

- Uses geolocation
- Send notifications
- Captures data from an input field
- Saves the data to a firebase database

User Interface Mocks

Login Screen

Home Screen - initial landing screen after login



Editing expense screen - add/change category and/or payment method


Edit expense
$150 07/10/2013 2:45pm – WholeFoods
Category
Payment Method
Groceries
Checking
Done

Push notification - notifies the user is near a store and reminds him/her to add expense to the report

Place picker - Allows the user to generate locations that will trigger a push notification whenever their device is near a store in their list.

Widget - Shows latest transaction.

Key Considerations

How will your app handle data persistence?

Firebase Realtime Database

Describe any edge or corner cases in the UX.

Figure out a way to export data to another app

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

- Firebase 16.0.4 - For user authentication, storing user spending data, and cloud messaging
- Gradle 3.1.4 - Required for all android projects
- Retrofit 2.4.0 - For network calls and easy json parsing
- Timber 4.7.1 - For logging debug messages in debug mode
- Play services location 17.0.0 - For geolocation

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

Google Places API for geolocation and for triggering notifications if the user is near a store.

Next Steps: Required Tasks

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

Task 1: Project Setup

You may want to list the subtasks. For example:

- Create new Git repo
- Configure Firebase
- Get a google API key
- Configure Geofencing
- Configure notifications

Task 2: Implement UI for Each Activity and Fragment

List the subtasks. For example:

- Build UI for LoginActivity
- Build UI for Shopping History list activity
- Build UI for Place picker activity
- Build UI for editing expenses activity
- Create login, shopping history, edit expense and place picker layouts.
- Store resources in the resource folder where theme primary and secondary colors will be stored, as well as material design margins and paddings.

Task 3: Configure and implement libraries

- Implement firebase notifications
- Implement geolocation
- Implement firebase database
- Handle errors when there is no WiFi

Task 4: Implement Async Requests

- Using asyncTask perform on-demand requests to locate the user and to persist data.
- Implement asyncTask loader to properly persist data to a Firebase database

Task 5: Accessibility

- Implement material design best practices to allow anyone to navigate and interact with the app more easily.
- Label UI elements with android:ContentDescription
- Use accessibility friendly titles for screen panes with android:accessiblityPaneTitle
- Arrange related content into groups to create natural groupings to reduce swiping, scanning or waiting too long to process the information on the screen.
- Make touch targets large enough to make it easier for anyone to interact. Minimum size 48 by 48

Task 6 : Generate build and release

- Generate production build
- Release to the play store

Requirements

- **Minimum SDK version 17**
- **Target SDK Version 28**