GitHub Username: Calebzor

CheapList

Description

The app allows users to compare products from two different grocery stores side by side in order to find the best deals.

Intended User

People doing grocery shopping. Mainly hungarians doing online shopping from the two biggest online stores.

Features

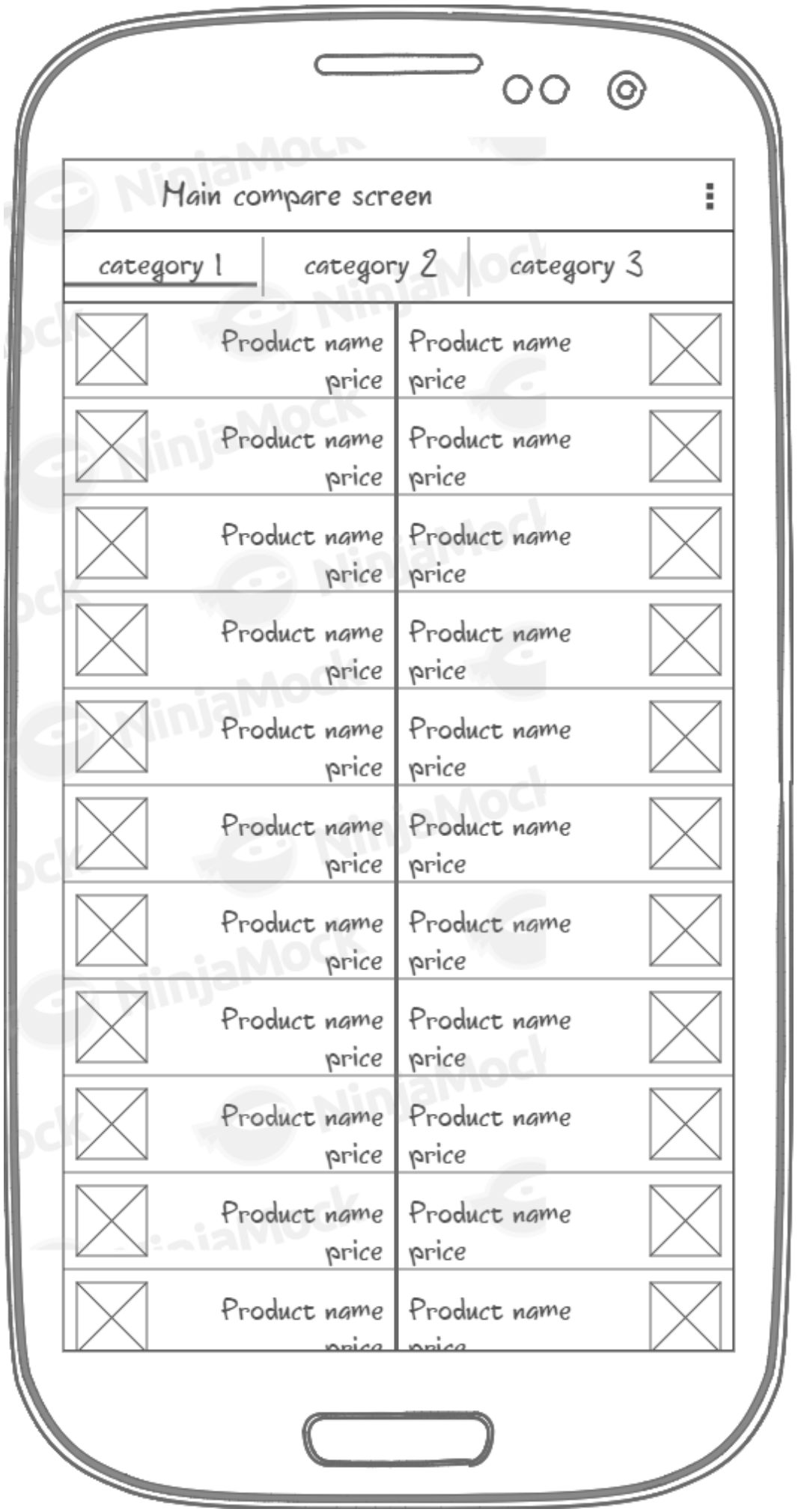
- Get more information of a product sold by a given merchant such as nutrition information, manufacturer information and more. (if available from the merchant)

- Manage a shopping list of products
- View shopping list in a widget

User Interface Mocks

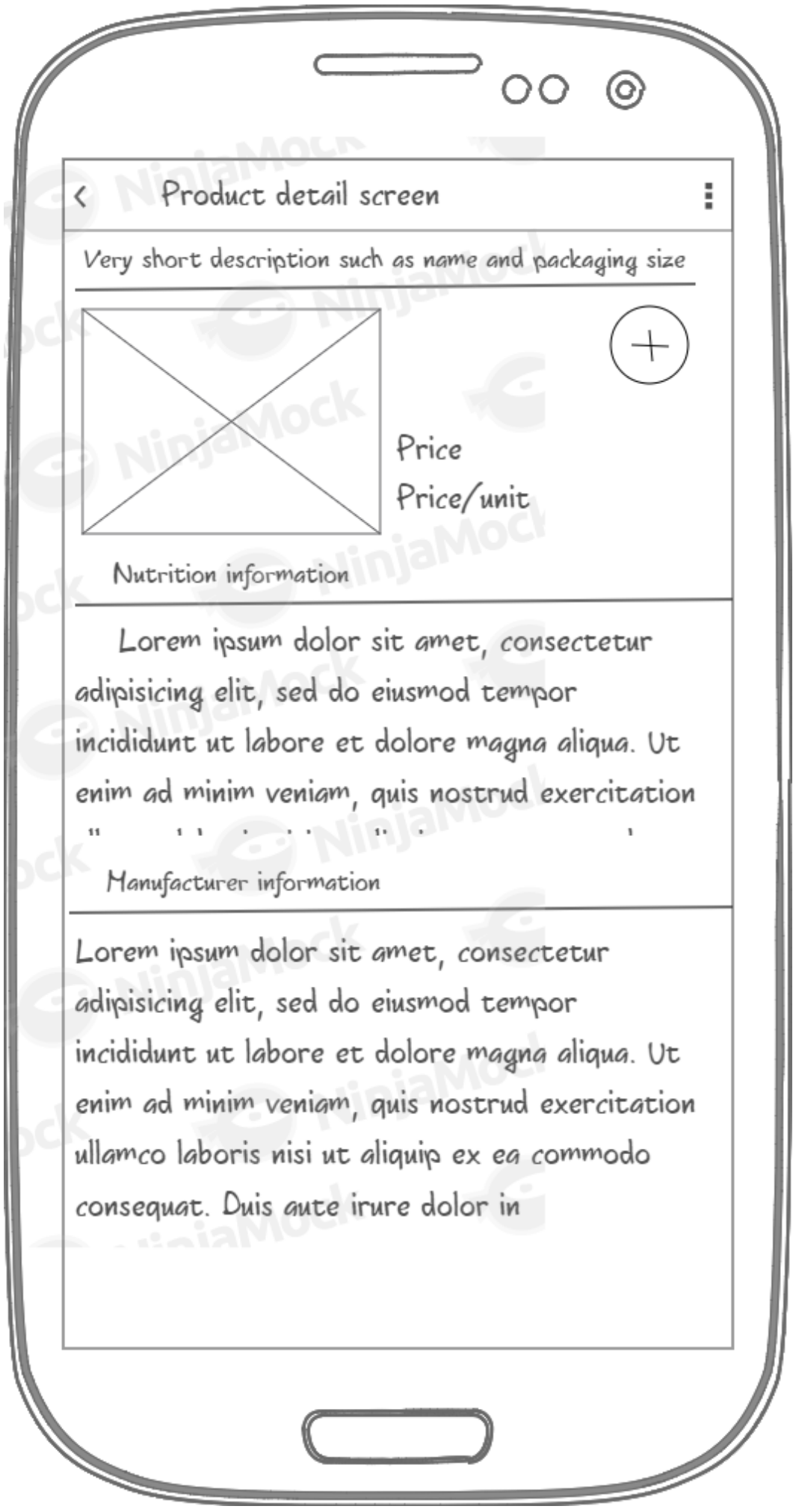
https://www.ninjamock.com/s/C6TVRWx a bit interactive

Main - categories


Main compare screen
category 1
category 2
category 3
Product name
price
Product name
price

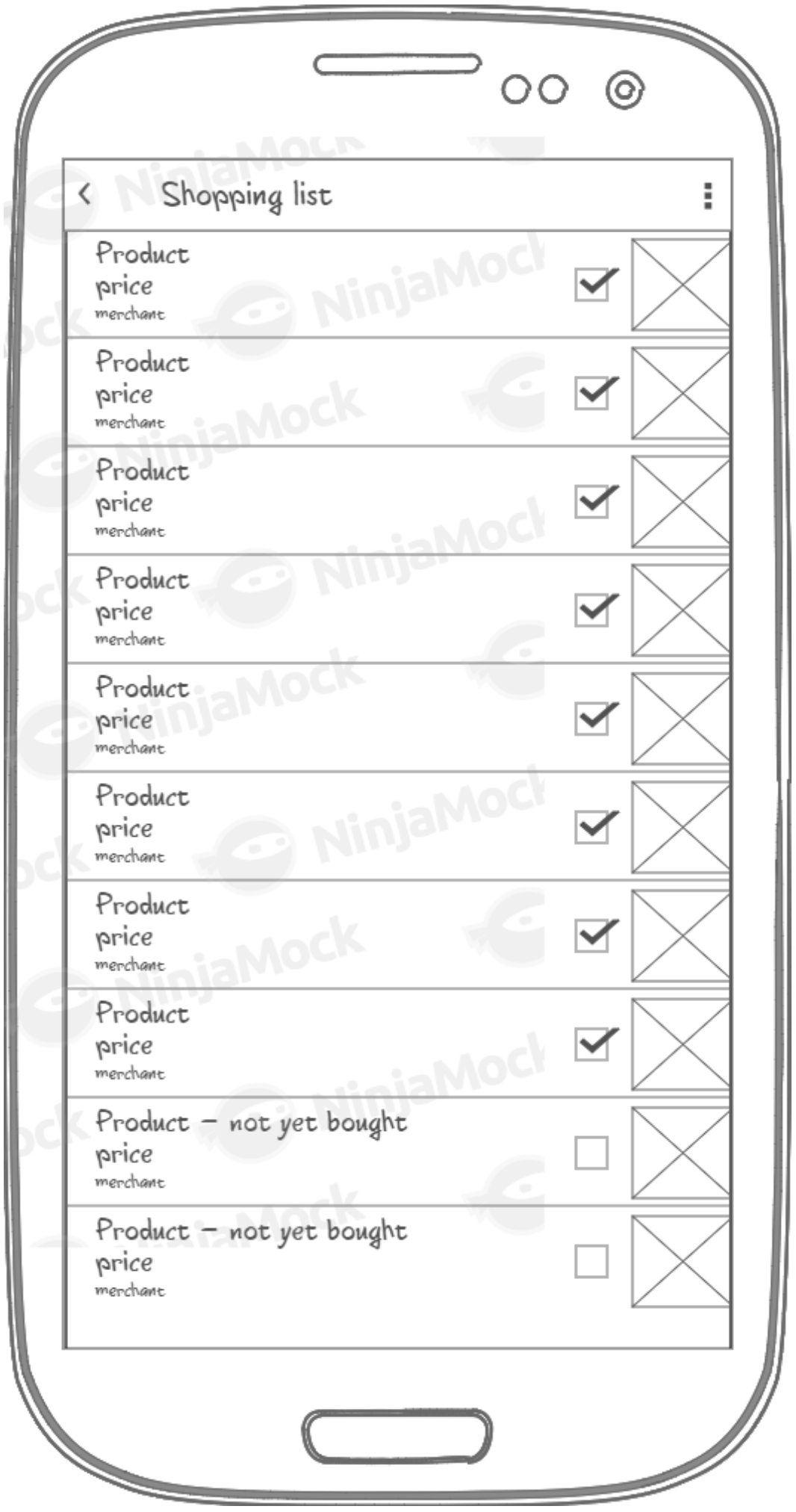
Main screen with two independently scrolling recycler views. Some items might not have images. There are tabs at the top for product categories.
Long pressing an item adds it to the shopping list. Snack bar action notifies about this and there is undo action.

Detail

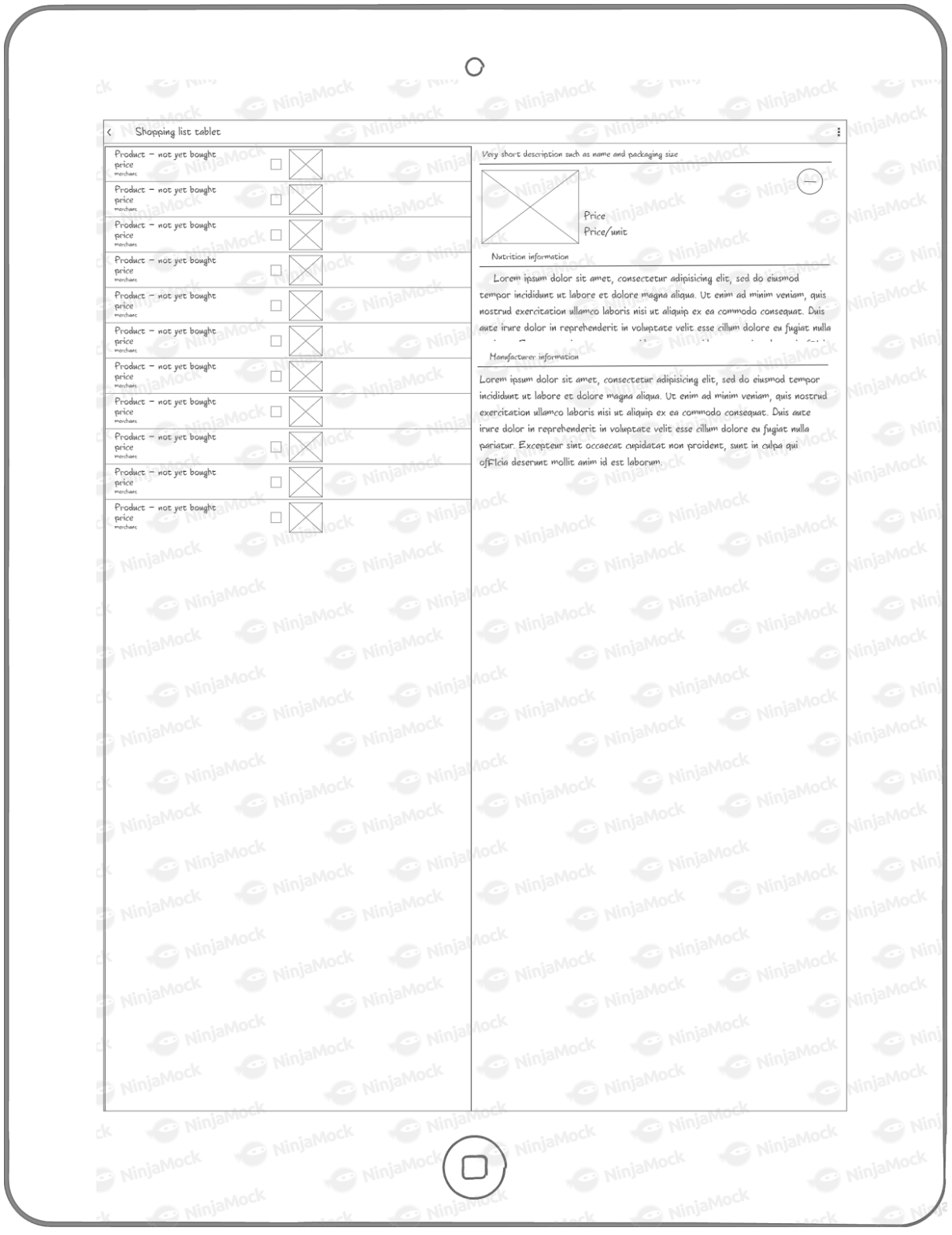

Product detail screen
Very short description such as name and packaging size
Price
Price/unit
Nutrition information
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation
Manufacturer information
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in

Fab button: add/remove product to/from shopping list

Shopping list

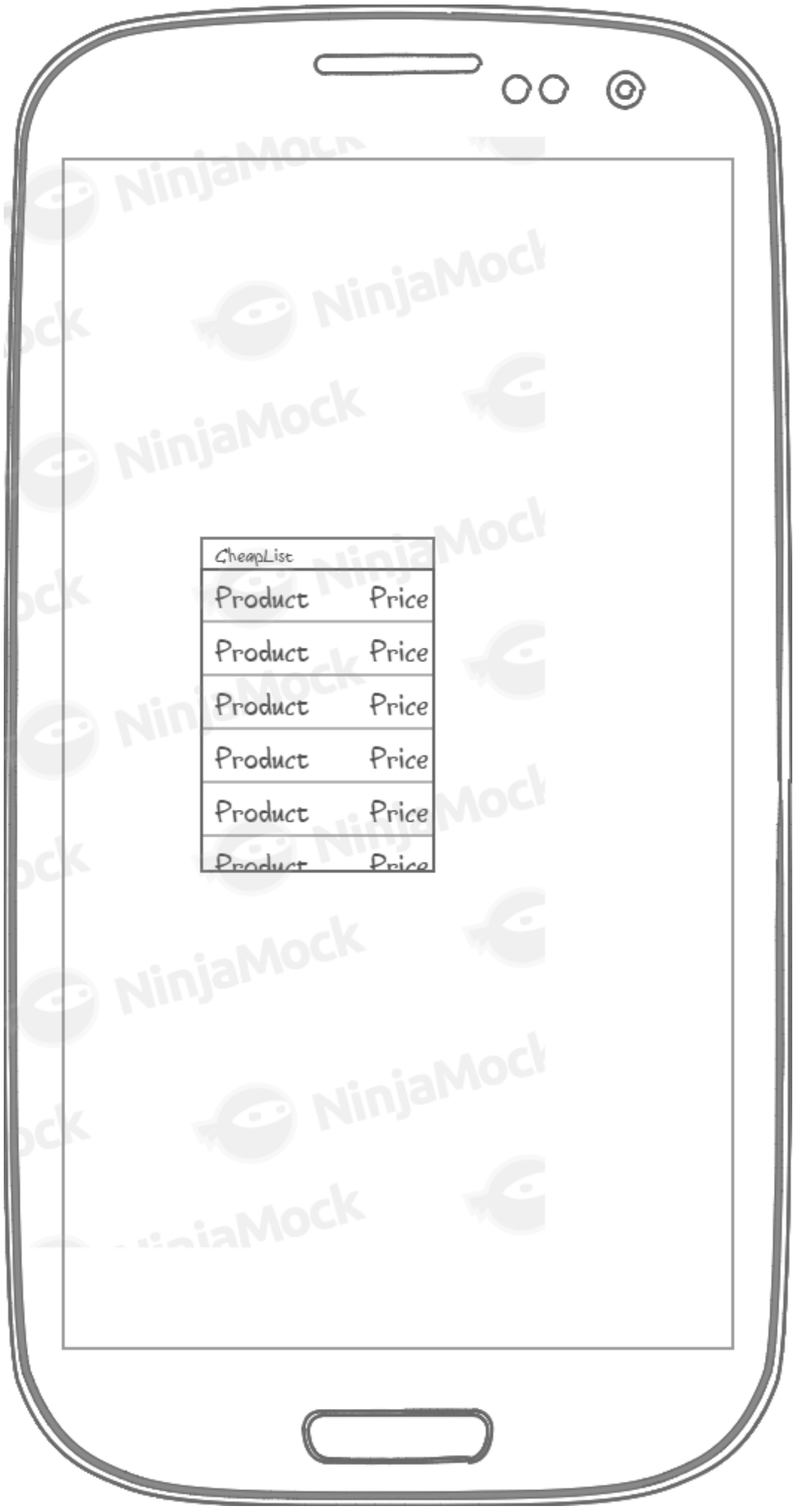

Shopping list
Product
price
merchant
Product
price
merchant
Product
price
merchant
Product
price
merchant
Product
price
merchant
Product
price
merchant
Product
price
merchant
Product
price
merchant
Product – not yet bought
price
merchant
Product – not yet bought
price
merchant

Not like on the screenshot: Not yet bought products are always at the top



Tablet view consists of the shopping list and the product detail. Fab button removes product from the shopping list. Just like in the phone view the list items are side swipable to remove them from the list.

Widget


CheapList
Product Price
Product Price
Product Price
Product Price
Product Price
Product Price

Tapping anywhere in the widget opens the shopping list screen.

Menu


Screen Title
Compare
Shopping list
Log in/out

Login


Google login
Firabse Ul Login Screen

Key Considerations

How will your app handle data persistence?

Firebase Realtime Database will be used for data persistence.

Describe any edge or corner cases in the UX.

Inside the app in the shopping list view items can be removed from the list with a swipe action, but when they are clicked a detail view of the product opens which has a fab button to remove the item from the list. On tablets the shopping list and the product detail view are on the same screen and both actions work, so you can swipe off an item from a list, in this case the detail view will jump to to the previous item on the list.

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

Picasso will be used for image loading and to allow easy caching of images.
Butterknife will be used to avoid having to write findViewById a lot
Firebase libraries will be used to work with firebase

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

Google analytics will be implemented with firebase, it'll be explicitly tracked which item gets added to the shopping list. When an item is removed from the shopping list it'll also be tracked.

AdMob: Interstitial adds will be shown when the users opens the detail view of a product from the shopping list (only every 10th detail view)

Next Steps: Required Tasks

Task 1: Project Setup

- Set up firebase for the project

- Import this json file https://pastebin.com/fUaPuc9q of merchants and products into the database. This is just some dummy data for development. In production cloud functions/backend would fetch and populate the publicly accessible part of the firebase real time database from third party sources. The database would be more structured too.
- Import this json file https://pastebin.com/BHvSTSWa for database rules (not production ready)
- Import and configure libraries

Task 2: Implement UI for Each Activity and Fragment

- Build UI for categories activity
- Build UI for product detail fragment
- Build UI for shopping list fragment
- Build UI for widget

More info about each screen can be found in the UI mockups. Especially pay attention to mentioned animations.

Task 3: Firebase

- Implement Firebase auth with Firebase UI
- Implement menu navigation
- Implement Firebase real time database connection, use adapters provided by firebase whenever possible to provide data to recycler views
- Implement FirebaseJobDispatcher to update categories data once a day

Task 4: Play services

- Use google analytics to track when an item gets added or removed from the shopping list
- Use AdMob to display interstitial adds every 10th opening of a products detail view

Task 5: Publishing

- Set up publishing of the app

Submission Instructions

- After you've completed all the sections, download this document as a PDF [File → Download as PDF]
    - Make sure the PDF is named "**Capstone_Stage1.pdf**"

- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
- Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"