Revised Test Plan - Receipt Scanner

Intro

The unit testing for the Receipt Scanner component will attempt to exercise all non-trivial methods in the modules/classes *ReceiptFileCollector*, *ReceiptDataExtractor* and *ReceiptItem*. I plan on using Jest <https://github.com/facebook/jest> as my main testing tool for this component. Jest has great runtime speed and is most suitable for the Test Driven Development approach which I plan to implement with this component. In addition, I will also use Enzyme <https://enzymejs.github.io/enzyme/>, for whole component testing in conjunction and to supplement Jest.

I will create folders for every module and a single folder will contain the main class and its own test file in Jest. The unit test functions will run and provide test data when the test files are compiled and run. For more sophisticated testing with a multimodular interest, I will make use of Enzyme by creating a main test suite folder that will instantiate and make mocks of the different classes/modules to facilitate whole component testing.

Whole component functionality testing

The main functionality paths for this component that i’ll seek to exercise are:

1. If a user provides permission to access the camera: obtaining a grocery receipt photo directly from a camera capture, and using it’s URI, successfully extract data from it by calling the veryfi API with the URI as one of the parameters. The data we are interested in for every item is: *description, amount/quantity,* and the *units.*
2. If a user fails to provide permission for camera access a warning flag will be displayed and halt proceedings.
3. If a user provides permission to access the gallery: obtaining a grocery receipt photo file path directly from the gallery, and using it’s URI, successfully extract data from it by calling the veryfi API with the URI as one of the parameters. Then assert the expected data we are interested in for every receipt i.e *description, amount/quantity,* and the *units.*
4. If a user fails to provide permission for gallery access a warning flag will be displayed and halt proceedings.

Test cases for this would be:

1. Using a short receipt with few items, create a *ReceiptItemList* of all the receipt items with the properties *description, amount/quanity,* and *units* set correctly, then assert their values on the *ReceiptItemList* output from *ReciptDataExtractor.*
2. Using Enzyme, we’ll create a shallow version of *ReceiptFileCollector* and obtain a button response for denial of camera/ gallery access, and assert a warning/notification flag.

## Per-Class Unit Testing

ReceiptFileCollector

The functionalities to be exercised in this class are mainly the pickFromGallery() and pickFromCamera() routines. I’ll seek to exercise them by:

1. If user grants permission for gallery access,