CMPT 275: Software Engineering I
Assignment 2 - Quality Assurance Plan

Group: 8

Team Name: The Great Eight

Joseph Dillman

Hoang Bao Ngan Nguyen

Wei Da (David) Song

Huy Thong Bui

Kevin Norman Scott Jerome

Payam Partow

Table of Contents

Revision History

Table 1 - Revision history of document

Revision	Status	Publication/Revision Date	By
1.0	Initial doc creation	October 8, 2019	HuyThong Bui
1.1	**Created** - Internal deadlines - Other considerations	October 14, 2019	Joseph Dillman
1.2	**Created** - Testing tools	October 14, 2019	David Song
1.3	**Created** - Unit/Integration Testing	October 15, 2019	HuyThong Bui
1.4	**Created** - Acceptance testing	October 15, 2019	Hoang Bao Ngan Nguyen
1.5	**Created** - Size/Complexity	October 15, 2019	Payam Partow
1.6	**Changed** - Formatting	October 16, 2019	Kevin Jerome

1. Testing Tools

We will utilize several unit and integration test methods in order to ensure that our code will be formally verified throughout the development process. In terms of unit testing, we will make use of the built in Xcode unit testing library, XCTest to write automated test scripts for our iOS application. Xcode also offers UI testing in the form of an iPhone simulator which may be used for both unit testing of individual features as well as integration testing when used in conjunction with our database [1].

Our database manager, Firebase offers a test SDK for cloud functions called *firebase-functions-test* which we will use in conjunction with the testing framework *Mocha*. This will allow us to unit test our database setup as well as create an environment for certain integration tests with the web or iOS application [2].

Finally, we plan to create a manual integration testing database which we will generate based on our requirements document and common use cases. This will be stored in a Google Sheet file and periodically updated to reflect any changes to our design or new test cases to consider.

2. Internal Deadlines

The following internal deadlines shown in Table 2 below will be set in order to ensure unit/integration testing is in progress before a release can be made. The three testing periods will correspond to the three main versions to be released. The testing periods encompass the Tuesday and Thursday prior to the Monday class where the release must be completed, to ensure team meetings may aid periods of testing. The time allotted between the start and end date is intended to be spent only on unit/integration testing as well as development involved in any new issues that may have arisen during testing. The weekend will be left after the end date before the due date to act as a buffer for any previously unknown or unresolved areas.

Table 2 - Internal deadlines for testing

Version	Start Date	End Date	Due Date
1	Oct 29, 2019	Nov 1, 2019	Nov 4, 2019
2	Nov 12, 2019	Nov 15, 2019	Nov 18, 2019
3	Nov 26, 2019	Nov 29, 2019	Dec 2, 2019

3. Acceptance Testing

User acceptance testing for the final version will tentatively take place on Thursday, November 28th which is 4 days before the due date so that adjustments can be made based on users' reviews We will attempt to make contact with Parkinson's Society of Vancouver to arrange for a group of 5-15 people at different ages and stages of disease that can test the final version. A simple review survey will be created for testers to write down their review. The survey might contain rating and ranking levels of satisfaction for each feature under the scale from 1 to 5.

Testers will be asked to do the following:

1. Set up the application on their phone
2. Create their own accounts from the authentication link
3. Check the profile to see if all the data imported properly and that they can edit incorrect data.
4. Able to follow all the instructions of the exercises.
5. Hit the synchronize button once they completed the exercises.
6. Set up time preference and frequency for exercise and medication reminder.

The goal for acceptance testing is to confirm that the app's user interface is suitable for Parkinson's patients and all major features are easily accessible.

4. Unit and Integration Testing

To discover problems in the early stage of the development process, when a unit feature of the app has been implemented, we will be performing unit testing to determine if the functionalities fit our requirements. A unit could be an interface object, a class, or it can be a method. Also, regression testing will be conducted to guarantee that previously developed and checked parts still perform properly after the any changes or bug fixes occur.

There are three integration tests, in which individual pieces are combined and tested together, corresponding with three main iterations of our app will be performed. The first integration test will evaluate our top priority features and ensure that they are working harmoniously; the other two integration tests will work on the lower rank features. The integration tests will cover more test cases and more comprehensive than unit testing.

5. Size and Complexity

The primary software tool we have decided to use is Swift Code Metrics, a software package that analyses our source code for synthetic metrics and modules coupling.

Synthetics metrics contain parameters such as:

- LOC (Lines of Code)
- NOC (Numbers of Comments)
- POC (Percentage of Comments)
- NOM (Number of Methods)
- Number of concretes (Number of classes and structs)
- NOT (Number of Tests)
- NOI (Number of Imports)

These parameters can aid us in understanding the layout of the project code and forecast future code growth rates. As shown in Figure 1 below, we may also automatically generate diagrams such as code distribution.

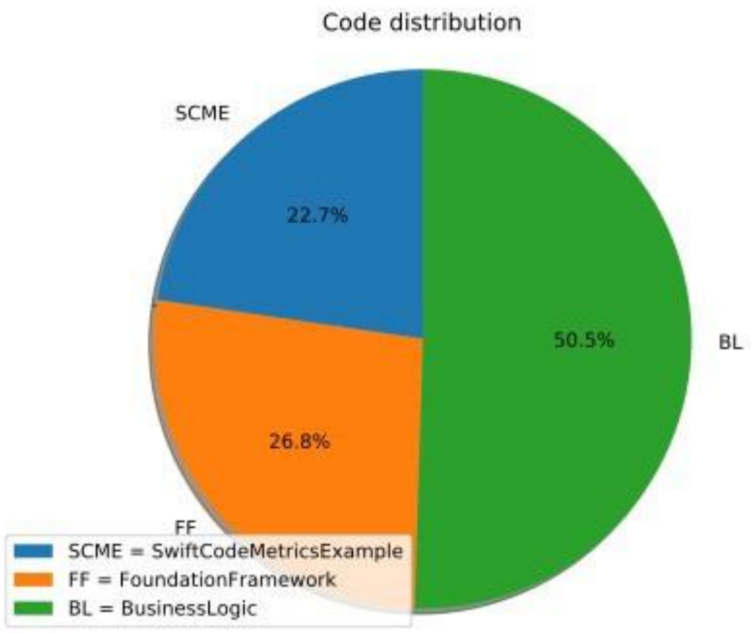

Figure 1: An example of a code distribution graph for an app based on LOC

Modules coupling measures the degree of coupling across all imported libraries.
Swift Code Metrics can help us visualize modules coupling by producing a dependencies graph which describes the number of import calls between frameworks. In Figure 2 below, an example dependencies graph is shown where the relative size of each box represents the size of a framework and the numbers show the import relationship to the framework from which the arrow originated [3].

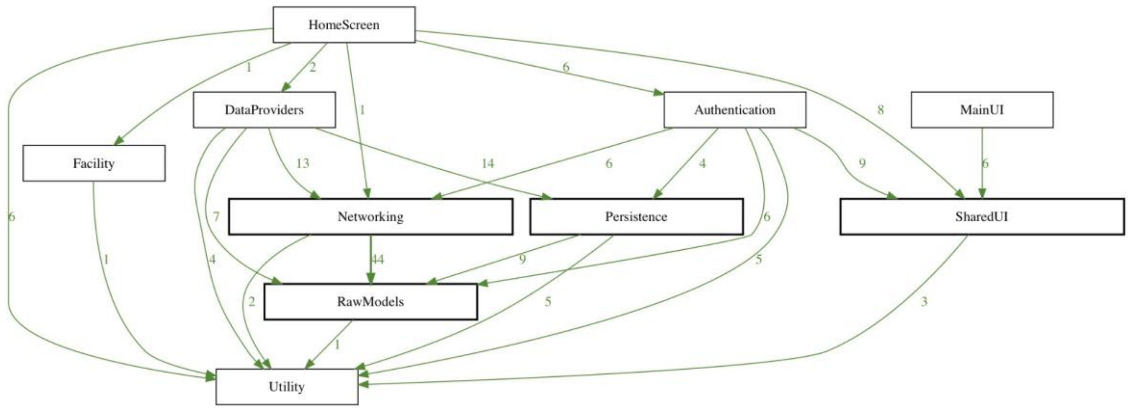

Figure 2: An example of an internal dependencies graph

6. Other Considerations to Ensure Quality

Several other measures are taken to ensure the quality of the application. In addition to weekly meetings where future plans are discussed, personal reports from each member of the group will be conducted on their previous contributions to the project. During these sessions, suggestions, comments and other input from the rest of the team will be shared in order to ensure agreement and satisfaction of the quality of the project components. During periods of software development, code reviews will also take place during team meetings to ensure all members of the team are up to date on technical portions. During periods of integration testing, a pair of iOS 7 and X devices and internet browsers will be tested during meetings to ensure progress is on track. All these measures will occur during the scheduled team meetings occurring on Tuesday and Thursday mornings from 10:30am to 12:30pm.

Where team decisions aren’t clear, or when general guidance on a decision is desired, communication with the course instructor may be scheduled. This is to ensure quality in any specific domain, specifically where several ideas are considered, however settling on one is not particularly obvious. This action is reserved only when all previous options have been exhausted.

7. References

[1] "Getting started with Xcode UI testing in Swift," *Swift by Sundell*, May 21, 2017. [Online]. Available: https://www.swiftbysundell.com/articles/getting-started-with-xcode-ui-testing-in-swift/. [Accessed Oct. 15, 2019].

[2] Google, "Unit testing of Cloud Functions," *Google*. [Online]. Available: https://firebase.google.com/docs/functions/unit-testing. [Accessed Oct. 15, 2019].

[3] Mattia Compollese, "Swift code metrics" [Online]. Available: https://blog.usejournal.com/swift-code-metrics-ea9ebf85416f. [Accessed Oct. 15, 2019].