CMPT 275: Software Engineering I
Fall 2017

HW5 - Quality Assurance Plan

Project Group 2 - The Night Owls

Fahd Chaudhry - 301215679

Karamveer Dhillon - 301209928

Ryan Serkouh - 301267718

Shawn Thai - 301291243

Yagnik Vadher - 301267298

Table of Contents:

Revision History:

Revision	Status	Publication / Revision Date	By
1.0	Created.	10/1/2017	Shawn Thai
1.1	Added Internal Deadlines	10/16/2017	Karamveer Dhillon
1.2	Added Software Testing, Unit and Integration Testing, User Testing.	10/19/2017	Karamveer Dhillon
1.3	Proofreading/Editing.	10/20/2017	Shawn Thai
1.4	Ensuring Quality section.	10/20/2017	Karamveer Dhillon
2.0	Added more information on testers in "User Acceptance Testing." Added details on code length in "Size and Complexity."	11/05/2017	Shawn Thai

1. Software Testing Tools

The development of the goTalk app will require the use of many different software tools. The main tool used to create the app will be the Xcode IDE. All of the coding and debugging will be done in Xcode. Additionally, Xcode provides a performance tuning application called Instruments. Instruments will be used to profile resource usage and check for memory leaks.

Finally, we will use Apple's TestFlight App to implement beta testing of our app. With TestFlight, we will be able to get feedback by inviting testers to try out our app in a closed beta environment.

2. Test Case Generation

In order to effectively debug the software, an exhaustive list of test cases will be required. Test cases will be generated individually and in groups. During development, any test case ideas related to the unit being developed will be added to a document on Google Drive. Additionally, we will hold brainstorming sessions at the end of each development phase in order to generate more test cases. The list of test cases will be divided amongst the testers during the testing phase.

3. Internal Deadlines

There will be three main testing phases, which will be conducted before the due date of each version of the App. The testing begins after the development phase, and ends at the upcoming due date. Due to the importance of testing, ample time has been allocated for the testing process. The testing schedule is outlined in *Table 1* below.

Table 1: Dates and Length of Each Main Testing Phase

	Start Date	**End Date**	**Length (Days)**	**Location**
Version 1	November 1, 2017	November 6, 2017	5	CSIL Lab, SFU
Version 2	November 16, 2017	November 20, 2017	4	CSIL Lab, SFU
Version 3	November 27, 2017	December 4, 2017	7	CSIL Lab, SFU

4. User Acceptance Testing

User acceptance testing for the final version of the app will begin on November 30, 2017, three days after the start of Version 3 testing. This will insure that some integration testing can be performed on the version that will be presented to beta testers.

A randomly selected group of children, split into different age groups, will act as our testers. Ideally, we will be looking to select children who have difficulty speaking, and seeing how they well they reciprocate with the app. This shall help represent the final audience for the app, and help us understand what expectations they may have for the app. The beta test will span over a four day period, and our own testing will be augmented by the feedback from the beta testers.

As mentioned previously, TestFlight will be used to conduct the beta test. The users will need to download the TestFlight app, which will then allow them to download the beta version of goTalk. The main purpose of the beta test will be to gather large amounts of feedback from users in order to find bugs and fix design issues.

5. Unit and Integration Testing

Unit testing will be conducted once individual units of the app have been implemented. Basic functionality and regression tests will be performed to make sure the unit works according to the requirements, and that there are no major bugs. This includes tests on results vs. actual values, boundary cases, corner cases, and unexpected return values.

There will be three integration tests, coinciding with the three main testing phases shown in *Table 1*. The first integration test will bring together the highest priority features and make sure they work seamlessly together. The next two integration tests will cover the remaining low priority features. Integration tests will explore test cases and be much more thorough than the preliminary unit testing.

6. Size and Complexity

The app will consist of approximately 12 classes to implement its various functions, such as classes for buttons, categories, cells, etc. We estimate that each class will have on average 100 lines of code, since some classes will have low hierarchy (cells) which may take < 50 lines, while classes with higher hierarchy (display bar) may have > 150 lines for their implementation. Therefore, we expect our app will contain approximately 1500 lines of code in total.

This does not include the code used to design the UI of the app, since we used Xcode's interactive storyboards to visually create the UI and did not physically code it.

We will use Xcode to count the lines of code in our final version, and we will use bash commands to count the number of total files. Additional metrics, such as CPU and memory usage, will be profiled using Instruments in Xcode.

7. Ensuring Quality

Quality will be monitored throughout the development process and polished during the testing phases. During development, we will continue to have our regular weekly meetings, in which we will discuss issues encountered during development and ideas for test cases. We will also review the code to maintain consistency and coding conventions. Meetings are scheduled weekly at 1:30pm on Mondays and 6pm on Wednesdays.