**Group 7**

Nathan Tannar

Jason Tsang

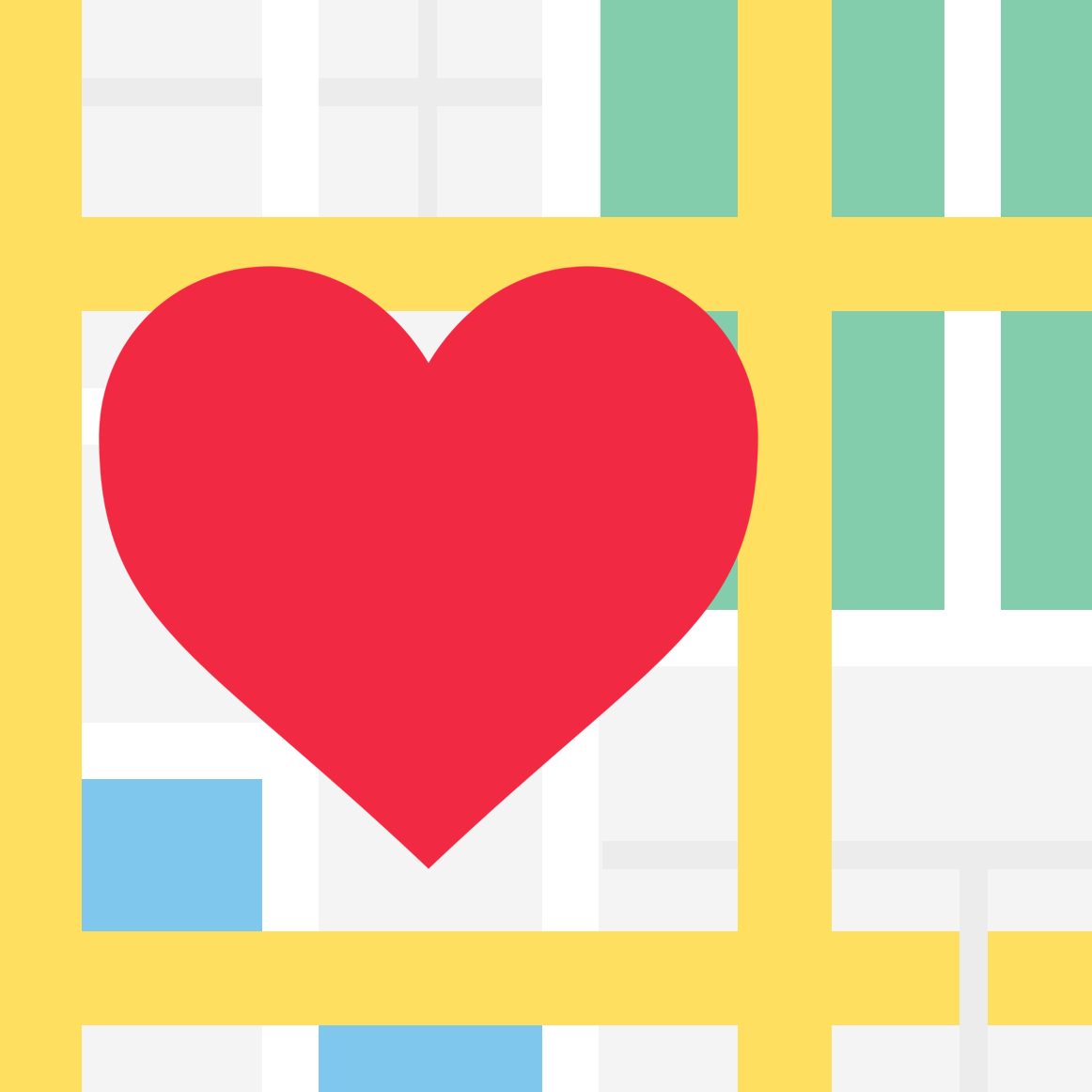
Philip Leblanc

Josh Shercliffe

Youjung Kim

**Website**

github.com/nathantannar4/Safety-Beacon



**Quality assurance  
plan**

**SAFETY BEACON - CMPT 275**

Figure 1: Safety Beacon Logo

**Table of Contents**

[1. Revision History 3](#_Toc496295824)

[2. Automated Testing 4](#_Toc496295825)

[3. Internal Testing 4](#_Toc496295826)

[4. User Testing 5](#_Toc496295827)

[5. Continuous Integration 6](#_Toc496295828)

[6. Performance Metrics/ Testing 6](#_Toc496295829)

[6.1 Projected Complexity Metric – Lines of Code 7](#_Toc496295830)

[6.2 Projected Complexity Metric – Number of Classes 8](#_Toc496295831)

[7. Other Quality Assurance 8](#_Toc496295832)

# 1. Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Status** | **Publish/Revision Date** | **Authors** |
| 1.0 | Created | October 19, 2017 | Nathan Tannar  Jason Tsang  Philip Leblanc  Josh Shercliffe  Youjung Kim |

# 2. Automated Testing

XCode’s XCTest framework will be utilized to create and execute test scripts, which can be used to automate the testing of basic functions and features such as data synchronization with the server. Unfortunately, location tracking and augmented reality components of the app cannot be tested using the same methodology, as they require the hardware of a physical device. However, by utilizing the Crashlytics framework from Fabric (owned by Google), if a crash or error occurs when testing those components, an automatic report can be generated and sent to Fabric’s servers to be analyzed. This will help alleviate the need to carry both a computer and iPhone around in testing the location tracking and augmented reality, but instead just the iPhone itself.

As Git will be used for version controlling, it is important that new pushes to the master branch do not cause the master build to fail. By utilizing Travis CI, a build of the app will be automatically triggered when new changes are committed. The build results are then displayed and viewable on GitHub. This will prevent changes breaking a functional build.

# 3. Internal Testing

|  |  |  |
| --- | --- | --- |
| **Version** | **Feature** | **Test Date Deadline** |
| 1 | Login/Account Setup | October 22, 2017 |
| 1 | Patient Menu | October 26, 2017 |
| 1 | Settings Menu | October 26, 2017 |
| 1 | Location Tracking | November 3, 2017 |
| 1 | Bookmark Locations | November 3, 2017 |
| 1 | “Take Me Home” button | November 3, 2017 |
| 2 | Safe Zones | November 20, 2017 |
| 2 | Location History | November 20, 2017 |
| 2 | Analytics | November 20, 2017 |
| 2 | Basic Navigation | November 20, 2017 |
| 3 | Augmented Reality Navigation | November 25, 2017 |
| 3 | Turn-by-Turn Navigation | December 1, 2017 |
| 3 | UI Refinements | December 4, 2017 |
| 3 | Video Tutorials | December 4, 2017 |

# 4. User Testing

Version 1 of the application will be tested on Thursday, November 2nd. Version 3 of the application will be tested on Saturday, November 26th, and Sunday, November 27th. This will give the develop team adequate time to transition from Version 2, and implement the final features. Test users will consist of a pair of family friends, and will be given iPhones with the application installed. These iPhones will serve as the test devices.

The first round of testing will focus on the account login, account setup, core location tracking, and “take me home” button. Users will be split up into patients and caregivers, given Version 1 of the application, and asked to execute the following:

     Caregivers:

* Create an account for the patient and link it to themselves (the caregiver)
* Track the patient as they move around the city
* Access the settings menu

    Patients:

* Bookmark locations
* Test the “take me home” button

The second round of testing will focus on features added in Version’s 2 and 3. Users will be split up into Patients and Caregivers once more, and given Version 3 of the application. The patient will be asked to travel around the city, while the caregiver tracks the patient. The following features will be tested:

     Caregivers:

* Create a Safe Zone for the patient
* Confirm a push notification is received when the patient leaves the Safe Zone
* View the patient’s location history
* View the analytics section of the patient’s history

    Patients:

* View the patient’s location history
* Use basic navigation to navigate around the city
* Use augmented reality navigation to navigate around the city
* Use turn by turn navigation to navigate around the city

Any discrepancies for the above features will be reported immediately to the development team, alongside a step by step description of what is causing the issue (if possible). Ample time is given between testing dates and due dates, in order to ensure the user tested discrepancies will be fixed.

# 5. Continuous Integration

By following incremental development, and utilizing Git and Travis CI, changes can be first tested to ensure that they do not break existing features before being integrated. As a general rule, each new feature or component will be developed and tested in a separate Git branch. When it is complete and tested, it will be integrated into the master branch. Once all the features for a given version are integrated, final tests can be ran to ensure all components work together seamlessly without any bugs. This sanity testing will avoid the “Big Bang” approach that makes debugging issues difficult.

# 6. Performance Metrics/ Testing

The team behind Safety Beacon will be using a range of software and tools to assist with the development. These include but are not limited to:

|  |  |
| --- | --- |
| **Software/ Tool** | **Relevance** |
| XCode 9.0 | To develop the source code in Swift 4 |
| XCode Instruments | To test the performance of the app |
| iOS Simulator | To simulate the app and test general functionality |
| iPhone 6s Plus | To test the core features using a real device |
| CocoaPods | 3rd party framework integration |
| Mapbox | A map view API |
| Sketch | To design views prior to implementation |
| Jazzy | To generate documents for the source code classes and functions |
| Git and GitHub | Version control and remote source code host |
| Microsoft Word/ Google Documents | To write documents and reports |

## **6.1 Projected Complexity Metric – Lines of Code**

Below is the expected projection for the number of lines of code this app will gain throughout its development.

|  |  |
| --- | --- |
| **Projected Date** | **Number of Lines of Code** |
| October 20, 2017 | 0 |
| October 27, 2017 | 2100 |
| November 3, 2017 | 4200 |
| November 10, 2017 | 6300 |
| November 17, 2017 | 7350 |
| November 24, 2017 | 8820 |
| December 1, 2017 | 10080 |
| December 4, 2017 | 10500 |

|  |
| --- |
| Version 3  Version 2  Version 1 |
| *Figure 6.1 – Number of Lines of Code vs Date* | |

## **6.2 Projected Complexity Metric – Number of Classes**

Below is the expected projection for the number of classes this app will gain throughout its development.

|  |  |
| --- | --- |
| **Projected Date** | **Number of Classes** |
| October 20, 2017 | 0 |
| October 27, 2017 | 10 |
| November 3, 2017 | 14 |
| November 10, 2017 | 15 |
| November 17, 2017 | 15 |
| November 24, 2017 | 18 |
| December 1, 2017 | 20 |
| December 4, 2017 | 21 |

|  |
| --- |
|  |
| *Figure 6.2 – Number of Classes vs Date* | |

# 7. Other Quality Assurance

In general, the team behind Safety Beacon will adhere to the recommended developer guidelines set by Apple, in addition to the material design guidelines by Google.