Team Beta Test Plan

Ethan Lew (PO), Ethan Bockler (SM), Tim Diersing, Caleb Yun, Alan Vuong

Test Plan ID: 1

Purpose

We will be testing our program to ensure that it meets all the requirements of the baseball tour project. Testing will be done to find and eliminate errors or bugs and will be done throughout the software development process.

Scope

We will be testing our program against the project requirements as outlined in the requirement specification. For example:

- When modifying a team, confirm that the team has been modified and written to the database.
- Validate user input and confirm that invalid input is handled
- Validate calculations made pertaining to distances between campuses
- Validate that all buttons, drop down boxes, and other UI elements are functional to their intended use.

Test Strategy

1. Team members will submit a pull request with their changes kept in a separate branch.
    a. If applicable, team members will resolve merge conflicts, working with the other team member whose code needs to be modified.
2. Developers will verify that their code follows the definition of done and development team coding standards.
3. Developers will perform unit testing on their proposed changes before merging their pull request.
4. Black box testing will be performed by the development team prior to a sprint review to search for missing requirements.

Testing from a user's perspective

- Widgets meant to hold information are populated
- Buttons, drop down boxes, and similar interactive UI elements are functional
- Users can successfully modify the database and their changes should persist.
- Users can add or modify teams *and* stadiums.
- Distances and paths calculated by the program are correct

Testing from a developer's perspective

- The SQLite database correctly interfaces with Qt
- SQLite queries are all functional

- Signals and slots are all connected to each other correctly, providing accurate functionality per the requirement.
- Code checks for administrator when necessary, administrator status is passed between program states.

Entry Criteria

- Program has been unit tested, adhering to testing standards.
- Proposed changes can be merged successfully.

Exit Criteria

- Program has been tested and is found to be free of bugs.
- Program validates any input and rejects invalid input, if applicable.

Suspension Criteria

- Bugs are present in the code after extensive testing
- Extensive merge conflicts are present, requiring code to be restructured.

Approval Process

- Proposed changes will be approved by the product manager and by any other developers whose code is being modified due to a merge conflict, if applicable.
- If approved, the issue pertaining to the user story in question should be marked “closed” on GitHub, and a screenshot of the merged changes (if applicable) should be added to the issue as a comment.

Schedule

- Unit Testing will be performed as a developer works on a user story, allowing debugging as the user story is completed.
- All developers must use Black Box testing prior to a sprint review, verifying that the program meets the expected result.

Necessary Training

- Familiarity of GitHub as a version control system and agile management tool.
- Knowledge of Qt GUI development and C++ programming.
- Basic SQLite operation

Environment

- Hardware
    - Desktops or laptop computers
- Software
    - Microsoft Windows
    - MinGW
    - Qt Creator

- SQLite tool such as sqlite3 cli or SQLiteStudio

Configuration Management

- Development of each user story should take place on a dedicated branch.
- All development and testing should take place on the branch corresponding to the user story in question.
- If a test fails, developers are responsible for modifying their code and notifying the team to receive help.
- When black box testing is completed successfully, the branch shall be merged into the main branch.

Support Documents

- Qt Class Documentation
- SQLite Reference Documentation
- Class UML diagrams

Glossary

1. Unit Testing
    a. A testing method that is used to test a certain method or class. Usually occurs during and/or after the development process.
    b. Each developer must perform unit testing during the development process of each user story.
2. Black Box Testing
    a. A testing method that is used to test the program without considering any internal structures or code.
    b. All developers will perform black box testing prior to a sprint review.