**Stadium Tour**

Project Members: Bryce Callender, Brandom Kem ,Jared Agle, Brian

Clinkenbeard

**Test Plan**

By: My group jeff

Project Manager: Brian Clinkenbeard

Scrum Master: Bryce Callender

Prepared by: My group jeff

**Scope of the test plan:**

All methods and functions will be tested to ensure that the code meets its design and behaves as intended. Inputs for functions/methods will be tested with boundary values and unexpected values as well as using qDebug. Interactive UI elements (such as buttons, drop menus, etc.) will be tested to ensure they are working as intended by comparing the results of the test with the expected results. What will be tested:

* **Dijkstra**
  + Determines the quickest way to traverse through the graph. Verify that the distances are correct and the algorithm works as intended.
* **Minimum Spanning Tree**
  + Determines the best way to greedily traverse through the graph while hitting every single vertex. Verify it takes the correct paths through the graph.
* **Login Interface**
  + Allows the user to login as well as the admin to login and perform maintenance. Verify that the admin and user login works as intended.
* **SQL Database**
  + Database that holds information of the stadiums, distances, as well as souvenirs. Persistence of data should be tested after modification of the DB.
* **UI Elements**
  + Box/button to select number of desired stadiums to visit.
  + Drop-down selection box to select the stadium to start at.
  + Purchase souvenir button with quantity box.

**PLAN:**

● **Dijkstra**

The dijkstra function will be tested by trying various runs of the trips and then, by hand manually checking if the output the function gave is appropriate and correct. Only testing a certain amount. This means that if the amount we test are correct all corresponding tests of this algorithm should work.

● **Minimum Spanning Tree**

The MST function will be tested by trying various runs of the trips and then, by hand manually checking if the output the function gave is appropriate and correct. Only testing a certain amount. This means that if the amount we test are correct all corresponding tests of this algorithm should work.

● **Login Interface**

The user or admin should be able to type in a username and password in the UI. The program will check if the name and password match any data. We will test this by inputting bad data and seeing if the program does anything. Then we will test a regular user and admin user and see if the ui is different for both logins. An admin-login should allow admin abilities while a user login should only allow user abilities. This will be tested by entering both valid and invalid combinations of usernames/passwords into the UI and verifying that only registered users/admins can login and non-registered combinations.

● **SQL Database**

The database is only here to store and keep the program persistent. This will be tested by running the program and for this program we will test changing a souvenir and going back and seeing if it has changed. The same applies for adding new colleges and seeing if the combo boxes have actually changed values with the new ones.

● **UI Elements**

Ui elements are all tested with qDebug()’s and we can test the buttons by seeing if they add or change any ui elements on the screen or output what we need to the console.

**TASK ASSIGNEES:**

The assignee will be responsible for testing their assigned task.

* **Dijkstra**
  + Each team of 2 members from assignment 12
* **MST**
  + Each team of 2 members from assignment 12
* **Login Interface**
  + Brandon Kem
* **SQL Database**
  + Bryce Callender
* **UI Elements**
  + Jared Agle

**Degree of Tester Independence:**

The degree each tester will have is that they will test their own respective assigned elements and then once they tested and completed their task they will show each other member of the group to verify that what they have done and tested is to the complete degree that we as the group have determined as good enough.

**Entry Criteria:**

The entry criteria for the dijkstra and minimum spanning tree algorithm is that we will need the database working in order to test it since it grabs all the values from the database. The database has no real condition that we need to hit to start it, only needs to be populated at first. The entry criteria for the ui is that we need to have designed the ui and what we want it to look like before we can actually test it. Finally in order to start testing the login we will need initial login information to actually test logging in.

**Exit Criteria:**

In order to stop testing the dijkstra algorithm it must pass multiple testing phases. It must then output the correct distances to each vertex from the starting vertex. In order to stop testing the MST it must choose the correct path to traverse through the graphing make it the minimum distance of the whole graph. It must visit all vertices to pass its testing as well. Once it passes these two tests its cleared to be deployed for final version. The database must show the changed values and new values added before it can pass its final testing and be in the final version. The UI elements must have functionality and do exactly as they say on name or function, like combo boxes. The login is cleared once a user can login, and admin can login, and a invalid user can’t login. Only then will this whole test plan be done with its testing phase.

**Schedule:**

The schedule for the testing phase will be the wednesday before each sprint. Each user will test their functionality and jobs before the sprint to make sure the code and program is what we would want to show to the customer.

**Necessary Training for Testing:**

The only training we need is to teach each member how to use the database and the qDebug() in order to see that they are testing and getting the right values back to test.

**Environment Description:**

The environment that is needed for this testing is QT and DB browser for SQLite. These two things are needed to test UI and database files.

**Approval Process:**

This testing plan and product will only be approved by the product owner as they know what the customer wants and exactly needs us to do.

**Glossary of Terms:**

**QT:** Ui program

**DB browser for SQLite:** Database third party program

**qDebug():** function in QT to show console output

**SQL:** Language for storing, manipulating, and retrieving data in database files.

**Dijkstra:** Path finding algorithm

**MST:** Minimum spanning tree algorithm