Group Members:

Trevor Holm

Mark Qiu

Patrick Lee

Julien Savary

Yaqub Hasan

The Durian Test Plan

Test Plan ID: 1

Purpose of Test Plan: The purpose of this test play is to ensure that the program is displaying the correct information. The program gives the user several options for how they want their information displayed. Depending on the user's selection, the program will either display all teams, Major League teams, American League teams, or National League teams. The program will even sort the list by team name, stadium name, stadium opening date, park typology, seating capacity, and distance to center field.

Scope of Test Plan: We will be testing the different methods of displaying certain information following the requirements given by the agile stories:

- Display all the information related to only one particular baseball team.
- Display the list of all the major league team names and their corresponding stadium names sorted by team name.

- Display the list of all the major league team names and their corresponding stadium names sorted by stadium name.
- Display the list of American League team names and their corresponding stadium names sorted by team name.
- Display the list of National League team names and their corresponding stadium names sorted by stadium name.
- Display the list of stadiums and their corresponding team name sorted by park typology. The program should also display the park typology.
- Display the list of team names that have an open roof type sorted by team name. The program should also display the number of teams with an open roof type.
- Display the list of stadiums and their corresponding team name in chronological order using the date opened. The program should also display the date opened.
- Display the list of stadiums and their corresponding team names sorted by seating capacity. The program should also display the seating capacity and the total capacity of all major league teams.
- Display only the stadium(s) and corresponding team name(s) that has the greatest distance to the center field.
- Display only the stadium(s) and corresponding team name(s) that has the smallest distance to the center field.
- Starting from Dodger Stadium and the user's choice of other teams, validate the legitimacy of the calculations pertaining to the total distance traveled. (should implement Dijkstra's or the A* algorithm)
- Based on the user's input, validate the legitimacy of the calculations

pertaining to the total distance traveled. (should implement Dijkstra's or the A* algorithm)

- Starting from Marlins Park and visit all the teams, validate the legitimacy of the calculations pertaining to the total distance traveled. (recursively choose the team closest to the previous team)
- Based on the user's input, validate the legitimacy of the calculations pertaining to the total distance traveled. (recursively choose the team closest to the previous team)
- Validate the functionality for the widget connections. Confirm that it will switch to the correct widget based on the story.
- The most logical variable types and correct values are used for the different tables in the database. The values must be linear to the values given on the excel sheet.
- Creating a minimum spanning tree (MST) connected to all of the MLB stadiums
- Implement Prim's or Kruskal's Algorithm and check to see that it has correctly connected all of the MLB stadiums
- Use a DFS on all of the stadiums. Ensure when using the DFS, that the trip starts at Oracle Park (San Francisco Giants).
- Use a BFS on all of the stadiums. Ensure when using the BFS, that the trip starts at Target Field (Minnesota Twins).
- Maintenance (administrator only - requires an encrypted password to gain access)

    - Provide the capability to add a new stadium and its corresponding souvenirs by having your program read from an input file given to the class
    - Provide the capability to change the prices of the traditional souvenirs
    - Provide the capability to add new traditional souvenirs
    - Provide the capability to delete traditional souvenirs
- Provide the ability to modify stadium information including capacity, stadium name, playing surface, roof type, ballpark typology, date opened, distance to center field, and location if a team moves into a new stadium. (administrator only) (The Oakland Athletics are planning to move to the Oakland Ballpark with a seating capacity of 35,000 in 2023).
-

<u>Responsibilities of Product Owner, Scrum Master and Team Members:</u>

- Team members will notify the Product Owner once they have completed the various displaying tasks given via story.
- The Product Owner will run the program and have it display the necessary information depending on the different options. While doing this, the Product owner is checking to see if the information displayed is correct while also bug/error-checking to test the developer's code.
- If the Product Owner encounters an error with the information displayed, the Product Owner must notify the developer immediately. From there, the

developer will fix the code to ensure that the program is displaying the information correctly.

- The Scrum Master will check to see if developers are in need of assistance.
- The Scrum Master will ensure that the team rules/guidelines are being followed.
- If the Product Owner encounters bugs/errors, the Product Owner must notify the developer immediately. From there, the developer must debug until the bugs/errors are non-existent.
- Each developer must follow the definition of finished code. In other words, the code must meet all the requirements given via story and properly error checking user inputs/outputs/boundary values.

<u>What features will be tested from a user's perspective:</u>

- The correct information is displayed for each baseball team.
- The program offers multiple display options to the user.
- When the user picks one of the many display options, the program will format the display correctly before outputting the correct list.
- The program's ability to retrieve information from the database.
- The program's ability to display all baseball teams, only Major League teams, only American League teams, or only National League teams.
- The program's ability to only display the desired output.
- The program's ability to sort by team name, stadium name, roof type, opening date, and more.

- The program's ability to calculate the distance to the center field.
- All trips should be able to display the total distance and the chosen order of the campuses the user selected.
- The program provides a choice to start a trip from Dodger Stadium and be able to select other colleges to finish the trip.
- The program provides a choice to allow users to customize their trip in Dijkstra or A* algorithms.
- The program provides a choice to start a trip from Marlin’s Park and be able to select other colleges to finish the trip.
- The program provides a choice to allow users to customize their trip in recursive order.
- The correct distances and calculated from each respective trip
- Sending and receiving of information from the database
- The user will be able to purchase souvenirs after each trip using different algorithms
- The starting location is correct
- The total distance and the most efficient order of the baseball stadiums from the trip that was selected.
- The program’s ability to allow a user to login as an administrator using the correct credentials
- The program’s ability to reject a user from logging in as a administrator if the incorrect credentials are inputted
- The program’s ability to add a stadium from an input file when the user is

logged in as an administrator

- The program's ability to change the prices of traditional souvenirs
- The program's ability to add new traditional souvenirs
- The program's ability to delete traditional souvenirs
- The program's ability to modify a stadium's capacity
- The program's ability to modify a stadium's name
- The program's ability to modify a stadium's playing surface
- The program's ability to modify a stadium's roof type
- The program's ability to modify a stadium's ballpark typology
- The program's ability to modify a stadium's date opened
- The program's ability to modify a stadium's distance to center field
- The program's ability to modify a stadium's location if a team moves into a new stadium

What features will not be tested from a user's perspective and what the system does:

- Code that links the QT application and SQL database together.
- The SQL queries that allow for the execution of certain information from the database
- How widgets are connected with one another (Signals and Slots).
- Much of the code implements the specific functions needed to meet the requirements of all college students and administrators.
- Code that converts the variable types in a Txt file to a QT variable type, thus

transferring the data to a SQL database.

Entry Criteria:

- Each developer must have their code written so that they can check for proper functionality. Each developer will test their code during the development process

Exit Criteria:

- When the program has no bugs present, and displays the correct list depending on the user's choice. As well as having the testing strategies for the displaying aspect of the program completed with successful outcomes.
- The program accounts for invalid or undesirable inputs and prevents any harm to the code/executions.

Suspension Criteria: The case where bugs are present despite various tests being conducted.

Approval process: The case will be approved once the product manager agrees with the condition.

Schedule: A developer will utilize testing strategies, such as unit testing when working on a story. This will allow the developer to debug code if necessary. Furthermore, every developer must utilize Black Box testing near the conclusion of the sprint. This will allow the developer to confirm that their code meets the necessary requirements of the story.

Necessary training needed for testing: Each developer has familiarity with using version control systems such as Github. Advanced knowledge of QT or C++ isn't required, but each developer must be proficient with both. The same ideas apply to SQL and using the SQL database.

Environment description:

Hardware: laptops/desktops that have good internet connections and work well.

Software: QT creator, SQLITE database browser such as DB Browser.

Configuration management (GITHUB): Finished stories must be uploaded on the developer's own branch. The Product Owner will run the code with the developer and check for errors. If any errors are found, the developer must debug and reupload the debugged code onto their own branch; thus, the process will resume again. Nearing the end of each sprint, the developer must have their branches equipped with the proper and completed code. Furthermore, Black Box testing will begin. Where the testing fails, the whole entire team must work together to debug. Once Black Box testing is completed without any existing errors, the code will be merged into the master branch.

Documents that support the test plan: UML diagrams, QT reference pages, SQL reference pages

Glossary:

GUI: Graphic User Interface

QT: Program used to create the GUI and overall program

Black Box Testing: Testing a part of the program by only checking to see if the output of the program is correct and not checking anything else.

White Box Testing: Testing the part of the program and checking every single use case and scenario to make sure the output for every scenario works.

UML Diagrams: Unified Modeling Language Diagrams to show how classes will join together and how each class will be used.

SQLITE/SQL: Program used to create the database for all the stadiums and other necessary info

Suspension Criteria: When testing for a certain part of the program needs to be paused as the testing for the program is creating errors or does not work.

Entry Criteria: When testing the part of the program can begin as the code is functional and can be tested further to elaborate further changes needed for the code.

Exit Criteria: When testing a certain part of the program

Schedule:

Sprint One: UML - Agile stories - Test plan - Display one team at a time - Display and sort by team name and stadium name - Login functionality - Finish database **(April 19)**

UML: Patrick(PO) & Julien (SM)
Display one team at a time: Mark(TM)
Display and sort by team name and stadium name: Trevor (TM) - Yaqub (TM)
Complete database: Trevor(TM)
Login functionality: Mark (TM)

Sprint Two: Finish displaying - Include DFS and BFS - Include souvenirs -Implement Dijkstra's Algorithm - Provide the capability for users to visit any other team of their choice staring at the Dodger stadium - Provide the capability to allow users to plan their dream vacation - Maintenance: Add/Delete/modify souvenirs - Modify stadium information **(May 3)**

Finish displaying all the needed information: Yaqub(TM)
Add the required souvenirs / Allow users to buy souvenirs: Trevor(TM)
Provide the capability for users to visit any other team of their choice staring at the Dodger stadium: Mark(TM)
Provide the capability to allow users to plan their dream vacation: Julien(SM)
Maintenance: Add/Delete/modify souvenirs - Modify stadium information: Patrick (PO)

Sprint Three: Provide the capability to visit all the teams starting at Marlins park traveling the shortest distance - Provide the capability for users to plan dream vacation by allowing them to choose their starting team - Finish maintenance requirements - Finish all the artifacts **(May 17)**

Provide the capability to visit all the teams starting at Marlins park traveling the shortest distance: Trevor (TM)
Provide the capability for users to plan dream vacation by allowing them to choose their starting team: Julien(SM)
Finish maintenance requirements: Patrick (PO)
Artifacts: Mark(TM) - Yaqub(TM)