Agile Stories

# Team: Jerry’s All-Stars

## Priority: Sprint 1

1. **As an administrator, I want to be able to keep records of the Major League and American League team names, name of the stadium, the location of the stadium, the team’s phone number, the date the team was created, the capacity of the stadium. the grass type of the field, and the souvenirs the team has.**

#### **Description**

The QT API for SQLite has built in functions that allow us to open/close the database and fill the database through a callback function. The purpose of this story is to show that the program is communicating with the database.

#### **Assumptions**

* There will be a SQL database file that will hold all of this information

#### **Tasks**

* Tasked to create the database file and input all the data.

#### **Tests:**

* Testing to ensure the database has been successfully connected, we will be outputting the list of the team's information into the GUI to determine if the database is being read correctly.

#### **Definition of Done**

The GUI will show stadiums with their information (address etc.) and the souvenirs that will be purchasable along with their information from a list in the program. It should match the input files provided with the project description.

#### **Story Point Estimation: 3**

1. **As the employer, I want to view a UML class diagram, three use cases, and three state diagrams for the project so that I know proper consideration has been paid to the design, structure, and usage of the program.**

#### **Description**

The UML Diagram will be consist of a class diagram, at least three use cases, and at least three state diagrams. We will use the UML Diagrams to help us develop the software of our program

#### **Assumptions**

* Assume that there is no code created. Determine the development strategy for the classes of the program.

#### **Tasks**

* Tasked to create the three different diagrams.

#### **Tests**

* Product Owner will check to ensure that diagrams have been created for all classes, as well as three use cases and three state diagrams.

#### **Definition of Done:**

Team will be able to turn in physical copies of the diagrams, and they will match the UML standards and the standards for the class.

#### **Story Point Estimation: 3**

1. **As the employer, I want to see a Team Test Plan so that I know that the program was thoroughly tested.**

#### **Description**

The Test Plan will show how we will be testing the program throughout development. It will include the purpose, scope, what is going to be tested by a user, what is going to be tested on the backend, and the criteria to start and end testing.

#### **Assumptions**

* Assume limited code has been created. We will be testing our code as we implement our project.

#### **Tasks**

* Tasked to create a test plan before implementation of code into project.

#### **Tests**

* Product Owner will check to see if each member is following the test plan, while also performing their own tests.

#### **Definition of Done:**

Team will be able to show in a word document or a printed out copy of the test plan. It will consist of a cover page, the purpose, the scope, the testing areas, and the criteria to start testing.

#### **Story Point Estimation: 2**

## Priority: Sprint Two

1. **As a baseball fan, I don't want to waste time traveling in between stadiums so that I can maximize my time with the stadiums and teams I care about.**

**Description**

Dijkstra's algorithm should be used to determine the shortest path between stadiums on a given trip, and the implementation should work within the business model of the classes designated by the UML diagrams.

#### **Assumptions**

* Stadiums will never be visited twice.
* The classes to manipulate will be as shown in the UML class diagram

#### **Tasks**

* Write implementation of Dijkstra's algorithm

#### **Tests**

* Algorithm must be desk checked and tested by hand to give the correct trips. (Formal unit testing need not be written.)

#### **Definition of Done**

The product owner will be able to verify that the implementation of the algorithm gives the shortest path as determined by hand (or another implementation known to work correctly). The implementation should be clear and documented, and it should be easily used by other members.

#### **Story Point Estimation: 13**

1. **As a baseball fan, I want to visit my stadiums across the U.S. in an efficient order where I am not backtracking and my trip is the shortest possible trip to take between the stadiums I have picked.**

**Description**

Kruskals or Prims algorithms should be used to find the minimum spanning tree and the implementation should work within the business model of the classes designated by the UML diagrams.

#### **Assumptions**

* Stadiums will never be visited twice.
* The classes to manipulate will be as shown in the UML class diagram.

#### **Tasks**

* Write implementation of Either Prims or Kruskals algorithm

#### **Tests**

* Algorithm must be desk checked and tested by hand to give the correct trips. (Formal unit testing need not be written.)

#### **Definition of Done**

The product owner will be able to verify that the implementation of the algorithm gives the shortest path as determined by hand (or another implementation known to work correctly). The implementation should be clear and documented, and it should be easily used by other members.

#### **Story Point Estimation: 13**

Priority: Sprint 3 (Final Sprint)

1. **As a baseball fan, I want to be able to plan a trip to every major league stadium starting from Comerica Park with the least traveling so that I can see every major league team without wasting time in between destinations.**

#### **Description**

The user should be given the shortest trip (using dijkstra's algorithm) and visit every other stadium. There should be no repeats and it should be a round trip. Any changes in the available stadiums should be reflected in the planned trip.

#### **Assumptions**

* Assuming the database has been created that has all of the team's information.
* Assuming that Dijkstra's algorithm is in place and working.

#### **Tasks**

* Tasked to create a planner that will provide the stadiums to visit.

#### **Tests**

* Test to make sure the Dijkstra's algorithm is working and performing correctly.
* Test for Dijkstra's algorithm to be working after there has been something changed to the stadium information/or a team has been added.
* Test for user invalid input.

#### **Definition of Done**

The GUI on the main menu will have a button to take you to this trip. The baseball fan will be able to see all of the stadiums that will be visited. There will be a button to take you on the trip and use the purchase souvenirs page from there.

#### **Story Point Estimation: 8**

1. **As a baseball fan, I want to be able to plan a trip starting at my preferred stadium and visiting every other Major league and American league stadium I want with the least traveling so that I can see just the teams I like without wasting time at other stadiums or in between destinations.**

#### **Description**

The user should be able to choose a starting stadium, and then each other stadium they would like from an updated list of all Major League and American League stadiums. The trip should be planned with the shortest travel distance (given by dijkstra's algorithm). There should be no repeats and it should be a round trip.

#### **Assumptions**

* Assuming the database has been created that has all of the team's information.
* Assuming that Dijkstra's algorithm is in place and working.

#### **Tasks**

* Tasked to create a planner that will provide the stadiums to visit.

#### **Tests**

* Test to make sure the Dijkstra's algorithm is working and performing correctly.
* Test for Dijkstra's algorithm to be working after there has been something changed to the stadium information/or a team has been added.
* Test for user invalid input.

#### **Definition of Done**

The GUI will have a button to take you to the custom trip. It will open up the custom trip planner that will have a selector to choose the starting stadium. There will be a list of the rest of the stadium to choose from. The baseball fan will be able to choose the other stadiums they would like to visit and begin the trip.

#### **Story Point Estimation: 8**

1. **As a vacationer, I want to purchase the teams souvenirs during my trip and be told the grand total at the end of my trip.**

#### **Description**

There will be a list of items the vacationer will be able to select at the given stadium. When the vacationer buys an item, the cost of the item will be added to the grand total.

#### **Assumptions**

* Assuming that the program will be able to plan a correct trip using Dijkstra's algorithm.
* There is access to the list of teams souvenirs.

#### **Tasks**

* Create a menu the vacationer can use to buy items.

#### **Tests**

* Testing the program to see if the team souvenirs are correctly named and priced.
* Testing that the program calculates the grand total correctly.

#### **Definition of Done**

After a trip has been planned, the GUI will show the purchase souvenirs tab. On the purchased souvenirs page there will be the an output of the current stadium. It will also display the souvenirs for that current stadium. The baseball fan will be able to click on a quantity to buy for each souvenir and a button that will purchase those souvenirs. There will be a button that will take the baseball fan to the next stadium.

#### **Story Point Estimation: 8**

1. **As a baseball fan, I would like to view the total revenue for each stadium, as well as a grand total for all the stadiums.**

#### **Description**

Each stadium object should have a way to store purchased items in a list/array/etc., and the program should be able to calculate revenue by adding the sale prices multiplied by the quantity of the items. When calculating the grand total, it will take the revenue from each stadium and add it to a single variable.

#### **Assumptions**

* The program is able to keep track of purchases made by each stadium.

#### **Tasks**

* Track revenue based on the purchases made by users
* Implement GUI interface to display revenue by stadium & the grand total

#### **Tests**

* Determine by hand the correct revenue from a specific set of purchases.
* Check that revenue by stadium is correct
* Check that the grand total revenue is correct

#### **Definition of Done**

At the end of trip the GUI will display a History tab that will have the total souvenirs bought, the Dijkstra's output and the MST output. The total souvenirs bought tab will have the number of items bought at each stadium, at each stadium it will give the baseball fan a total spent at that stadium and then will shot the grand total cost of the trip. The Dijkstra's tab will display the total distance of the entire trip and the cities that were visited in order with the corresponding distance. The MST tab will display the minimum spanning tree. and its total distance.

#### **Story Point Estimation: 1 (BASELINE STORY)**

1. **As an administrator, I want to be able to modify the information for each team, so that I may ensure that fans get to the correct stadium.**

#### **Description**

The admin will be able to change any of the team’s information from an administrative view. This will be used if a team was to move to a different location.

#### **Assumptions**

* Assuming the database has been created that has all of the team’s information.
* Assuming that there is an administrative window that only admins can access.

#### **Tasks**

* Tasked to create a function that will call the database and will save changes to the database.

#### **Tests**

* Test to make sure that the changed information is saved.
* Test for user invalid input.

**Definition of Done**

The GUI interface will have a new tab that will show the Stadium management page. The page will have the capability to click on the table to select which stadium to delete after a team move has been made. There will be a button to commence the deletion. If the admin wants to add a new stadium, they will need to enter the name and click add. From there they will be able to manually add each attribute of the stadium.

#### **Story Point Estimation: 2**

1. **As an administrator, I want to be able to add a new team and corresponding stadium to the database.**

#### **Description**

Administrators will be able to log in to a special portal within the program where they can add teams and stadiums to the database records that will then be persistent across executions and available during the planning and taking of trips.

#### **Assumptions**

* Assuming the database has been created that has all of the team's information.
* Assuming that there is an administrative window that only admins can access.

#### **Tasks**

* Tasked to create a function that will call the database and will save changes to the database.

#### **Tests**

* Test to make sure that the changed information is saved.
* Test for user invalid input.

#### **Definition of Done**

The GUI will have a button to load from file. The button will be located on the admin page next to the Souvenir and Stadium management pages. Once the button is pressed, a message will display saying the team was added. If the admin were to try again to add the file, it will output an error.

**Story Point Estimation: 3**

1. **As an administrator, I want to be able to add, delete, and modify the souvenirs that can be purchased at each stadium so that I can keep my inventory up to date and accurate.**

#### **Description**

Administrators will be able to log in to a special portal within the program where they can modify the souvenirs that baseball fans may buy. They should be presented with the list of current souvenirs that they may need to modify, and should be prompted before deleting a souvenir to avoid mistakes.

#### **Assumptions**

* Assuming the database has been created that has all of the team's information.
* Assuming the database has created a list of souvenirs.
* Assuming that there is an administrative window that only admins can access.

#### **Tasks**

* Tasked to create a function that will call the database and will save changes to the database.

#### **Tests**

* Test to make sure that the changed information is saved.
* Test for user invalid input.

#### **Definition of Done**

The GUI will open a new tab that will have the Souvenir management Page. The page will allow the admin to choose the team to perform an action on. The admin will be able to click on an existing souvenir or enter in a new souvenir. There will be an add/edit/delete button to perform the given functions.

#### **Story Point Estimation: 2**

1. **As an baseball fan or administrator, I want to be able to use a graphical interface to interact with the features of the program.**

#### **Description**

The gui works bug-free, without any major issues. Data is accessible to the user without accessing the backend.

#### **Assumptions**

* Assuming the entire backend of the project is complete and all features are working correctly.

#### **Tasks**

* Tasked to create beautiful Graphical Interface that the baseball fan can have fun and enjoy to use.
* Tasked to make the GUI easy to use and descriptive.

#### **Tests**

* Test to make sure GUI objects are working.
* Test for user invalid input.

#### **Definition of Done**

The main menu of the GUI will have an option for admin login. There will be a button to view all of the teams and stadium information sorted in various ways. There will be a button that will take you on a custom trip of the stadiums. There will be also a button that will take you on a trip to all of the stadiums.

The admin login will prompt for a password and will not advance until it is entered correctly. once on the admin page there will be buttons to access the Souvenir and Stadium Management pages, a load new team button, and a view that has all the updated information.

#### **Story Point Estimation: 13**

Jerry’s All-Stars

Anna Kolev, Chaz Del Prato, Katie Cox, Kamron Mahmoodzadeh

## Team Rules:

1. All communication must be done either in person or over Slack
2. If you are unable to complete a story that you have been assigned by the deadline, you must communicate this to the team immediately
3. Everyone needs to participate, both in planning and coding
4. If someone is unsure of what to do or what to work on, they should ask the team in Slack for help
5. All work should be completed in a timely manner
6. All team meetings will be over Slack when all members are available; 2 meetings a week minimum

## Coding Standards:

1. All code in must be in C++
2. Must code in Qt for optimal integration
3. Code should be formatted for maximum readability so other members can pick up easily where someone else may have left off
4. Naming convention for Git branches: “programmer’s initials”-“topic” (ex: ak-general, kc-gui, etc.)
5. Naming convention must be followed when naming Qt Widgets in the GUI. Name must include type of widget, purpose, and location (if used on multiple pages)
6. Comment any code that may be confusing logically or visually, all other comments will be Doxygen generated
7. When working on a story, must communicate over Waffle that you have assigned this story to yourself.