Team Lakers / Test Plan

Members: Nicholas Hu, Logan Sevcik and Hamidreza Shirazi

Test Plan ID: 1

Purpose of Test Plan:

Design a well-suited GUI application that supports the intentions and purposes for NBA Basketball Project. We will be testing that there are no errors present throughout the software process in order to ensure that the program runs successfully and meets all of the requirements.

Scope of Test Plan:

We will be testing the existing contents and requirements of the given agile stories. Some of the story testing represents the following:

● Depending on the basketball fan input, check whether or not the chosen report's results for any particular is correct.

● Depending on the basketball fan input, check whether or not the requirements been selected for each visit plan is correct.

● Depending on the Admin input, check whether or not the new team information which is saved in SQL database is correct.

Overall Test Strategy:

● The Product Owner will be notified once team members have completed their agile stories.

● Each developer must follow to meet all the requirements given via story and properly error checking user inputs/outputs/bounder values.

● Unit testing will be utilized by each developer to ensure the proper functionality for their code

● The Product Owner will go through a thorough bug/error-check to test the developer's code. All tasks on the story must be completed.

● If the Product Owner encounters bugs/error, the Product Owner must notify the developer immediately. From there, the developer must debug until the bugs/errors are non-existent.

● Black Box testing will be utilized by the entire team the Friday before a sprint check. This will allow the team to search for any requirements that are not present in the code yet.

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 allows for the execution of certain information from the database
● Code that converts xlsx format file to a QT variable type then inputs the data to a SQL database.

What Features will be Tested from a User's Perspective:
● The correct storage and display of information from the database.
● Sending and receiving of information from the database

● Each widget has their own functionality and works properly.
● Addition of teams based on whether or not the member already exists in the selected regular and team visit planned reports.

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
● For Black Box Testing, every developer must have their software written so that the program can be combined. Once merging issues are dealt with, the team must make sure that the software runs properly

Exit Criteria:
● When the program can run with no bugs and the testing strategies have been completed.
● 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 testing are conducted.

Approval Process

- Product owner assigns task, developers design initiate program team visit planning and Scrum master seeks clarification on tasks.
- Product owner gets feedback of team member and developers submit first draft of program.
- Program owner returns comments, developers submits final edition of program and developers make edits.

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. Every developer must utilize Black Box testing nearing the conclusion of the sprint. This will allow the developer to confirm that their code meets the necessary requirements of the story.

Configuration management (GITHUB):
Finished stories must be uploaded on the developers 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. 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.

Necessary training needed for testing:
Each developer has familiarity with using GitHub. Advanced knowledge of QT or C++ isn't required, but each developer must be proficient with both. Same ideas apply to SQL and using the SQL database.

Configuration management (GITHUB):
Finished stories must be uploaded on the developers 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. 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.

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

Software: QT creator, SQL database.

Documents that Support the Test Plan:
QT reference pages, SQL reference pages, UML diagrams

Glossary of Terms:
● Unit testing - Is a software testing methodology used to determine if a "unit" is ready for use. Each test verifies the behavior of one unit. For this project, each developer must perform unit testing during the software development process.

● Integrated Testing - Tests that verify the unit collaborations. Occurs during unit testing, where each developer will check to see that the collaboration of various units work accordingly and do not produce errors.

● Black box Testing- A method of software testing that examines the functionality of an application without peering into its internal structures or workings while every feature will be black box tested. This testing strategy will be used towards the end of each sprint when every developer is finished with his or her code such that consolidation can occur. The team will perform black box testing so that the program meets all its requirements.