Bulk Club

Version 1.0

Test Plan

Table of Contents

1. TEST PLAN ID……..
2. PURPOSE OF THE TEST PLAN……..
3. SCOPE OF THE TEST PLAN……
    3.1. What Will Be Tested……
4. OVERALL TEST STRATEGY……
5. WHAT FEATURES WILL BE TESTED FORM A USER'S PERSPECTIVE……
6. WHAT FEATURES WILL NOT BE TESTED FROM A USER'S PERSPECTIVE AND WHAT THE SYSTEM DOES……..
7. ENTRY CRITERIA ……..
8. EXIT CRITERIA.……
    8.1. When To Stop Testing…….
9. SUSPENSION CRITERIA……..
    9.1. Conditions When To Temporarily Stop Testing……...
10. ROLES AND RESPONSIBILITIES ……..
    10.1. Product Owner, Scrum Master, Team……..
11. APPROVAL PROCESS……..
12. WHITE BOX OR BLACK BOX TESTING OR BOTH?……...
13. NECESSARY TRAINING NEEDED FOR TESTING……..
14. ENVIRONMENT DESCRIPTION……..
    14.1. Hardware And Software Needed For Testing…….
15. CONFIGURATION MANAGEMENT APPROACH……..
    15.1. GITHUB Utilization……..
        15.1.1. Branch structure…….
        15.1.2. What happens when a test fails……
16. TEST DELIVERABLES……...
17. DOCUMENTS THAT SUPPORT THE TEST PLAN……...
18. GLOSSARY OF TERMS…….

1. Test Plan ID:

Bulk Club Version 1.0

This is the first working and testable version of this program. The program “Bulk Club” will use a Qt GUI to allow managers to see all existing information of members and items. It also allows an admin to edit and change the information about the members and items.

2. Purpose of Test Plan:

The purpose of the test plan is to outline the approach the team members will take to complete, test, and integrate the tasks into the overall project. It will include the objectives, responsibilities, schedule of each sprint, and approach.

3. Scope of Test Plan:

Functions that will be tested:

- GUI - will be tested on its ability to accurately facilitate other code functionality as well as for its ease of use
- SQL Database manipulation - will be tested on the front end for its visual representation but it is also essential for its testing to run parallel to other functionality in order to ensure it is correctly connected to the backend (white box testing is necessary).
- Quantity and Price Calculators - testing will be required to use the front end(black box testing) for much of it but it will be extensive enough so that no front end functionality is negatively affected by code.

4. Overall Test Strategy:

- Each person will work on the task they are assigned to, once a task is complete the team will come together to develop a test for the function, this may involve comparing resulting data to input file, stressing the system, etc.
- When a team member feels that their task is functional enough to begin testing they will commit it to main so that the other members can implement, test and edit it.
- Any member that is available in cooperation with the story’s author will test the output to make sure that it is functional and that it is ready for further implementation.
- If the testing of a task’s implementation is required and requires another task’s functionality, then they will be tested in conjunction with one another before their implementation.

- If no other team members are free to help test the pseudo finished story, then the original assignee will continue to refine and test the task. Final implementation will NOT occur until at least one other team member has tested the functionality of the task.

5. What features will be tested from a user's perspective:

The display functions, functions that require something to be clicked, and sorting functions will be tested from the user's perspective.

6. What features will not be tested from a user's perspective:

Any functions that will operate in the background, including reading in the database, updating the database, etc. will be tested separately.

7. Entry Criteria:

- User is using a stable version of the project and if possible the most up to date version possible
- The task must be functional enough to the point where it does not terminate the program
- All necessary requirements and documentation of the story should be complete.
- Must be compatible with QT software, Github, and other functions that have already been implemented
- Documentation must clearly show how the function works; any team member should be able to understand how each part of the function works.

8. Exit Criteria:

- The runtime of the function if it significantly affects the total time
- All obvious risk areas have been fully tested and are functional
- Any minor bugs seen during testing have all been fixed
- All requirements of the task are thoroughly tested and complete.

8.1 When to stop testing

  - For a story to be finished, it should be tested by multiple people and all should agree that it is finished and ready to be implemented into the UI.
  - If the story is considered not finished, the team members will tell the member who worked on the task what needs to be improved for the story to be complete.

The member can then fix those critiques and ask the members again when they are ready to retest it.

9. Suspension Criteria

- The build contains serious bugs that stop the program from working as intended
- The client wants to change something significant about the requirements
- The client says that part of the project that has been done is not what they wanted/ is unnecessary
- Team members have problem with the software

9.1 Conditions When To Temporarily Stop Testing

  - When any of the problems above are present, testing will stop in order to solve the immediate problem at hand
  - When all problems that caused suspension are resolved, testing will resume as normal

10. Roles and responsibilities

- Team Members - Cyrus, Saul, Parsa, Neda, Juli
  - Self managed team members who work on individual tasks
  - Can help others with tasks if they finish theirs early
  - Helps test tasks when other members finish.

11. Approval Process

- Any member that is available in cooperation with the story's author will test the output to make sure that it is functional and that it is ready for further implementation.
- If the testing of a task's implementation is required and requires another task's functionality, then they will be tested in conjunction with one another before their implementation.
- If no other team members are free to help test the pseudo finished story, then the original assignee will continue to refine and test the task. Final implementation will NOT occur until at least one other team member has tested the functionality of the task.
- For a story to be finished, it should be tested by multiple team members and all should agree that it is finished and ready to be implemented into the UI.
  - These members should perform a desk check to find the expected output, then compare it to the program's output. If the program's output matches the expected output, it should be approved. If it is different, it should be rejected.

- If the story is considered not finished, the team members will tell the member who worked on the task what needs to be improved for the story to be complete. The member can then fix those critiques and ask the members again when they are ready to retest it.
- Once a task is done, the team member will be able to start on a new task based on the priority of the remaining tasks.
- The project will be considered complete when all tasks are considered complete by all team members.

12. White Box or Black Box Testing or Both?

- We will be using Black Box Testing for this project
    - Black Box testing focuses on the functionality of the application
    - Tests parameters of the function without fully diving into the source code.
    - Testers will choose a valid input, and then possible boundary invalid inputs in order to test possible bugs or errors in the program.

13. Necessary Training Needed for Testing

- Knowledge of QT
- Basic understanding of SQLite
- Basic understanding of what the function being tested should accomplish

14. Environment Description

- Team members will primarily work independently on their home devices and in classroom
- Testing can happen either at home while team members are communicating online, or at Saddleback while members are together.

14.1 Hardware And Software Needed For Testing

- Testing will be done on using Github and QT
- Doxygen will be used to format the comments and the code

15. CONFIGURATION MANAGEMENT APPROACH

15.1 GITHUB Utilization

- Github will be used to test and merge each member's individual tasks, as well as to keep track of what tasks have been completed, are in progress, and need to be done.

15.1.1 Branch structure

- Each member will create their own branch of the most recent version of main, and implement their task to main using this branch
- Once the member feels that their task is complete, they can create a request for the other team members to test it
- If the members confirm that the task is fully complete, the branch will be merged with main.
- The second approach is for a team member to merge main into their local branch for all implementation and testing so that only after the branch is caught up to main it may make an easy pull request.

15.1.2 What happens when a test fails

- If the test fails and the task is considered not finished, the team members will tell the member who worked on the task what needs to be improved for the task to be complete. The member can then fix those critiques and submit another request when they are ready to retest it

16.Test Deliverables

- The Test Plan
- Doxygen HTML comments
- Uses C++ and SQLite

17. Documents that support the test plan

- Testing Powerpoint
- Instructions PDF
- Bulk Club User Story Document
- UML Diagrams

18. Glossary of terms

- GUI : Graphical user interface
- QT: Software used to design and test GUI
- SQL: Language that the database for the restaurants will be using.