**BULK CLUB**

CS1C Project 2

**TEST PLAN**

Team: Code Miners

Version Num. : 2

Effective Date: 25 Nov 2019

Prepared by: Adam Hurd and Moe Soliman

**-TEST PLAN-**

**PROGRAM OVERVIEW:** Bulk Club is a customer and inventory management program for membership-based wholesale retailers. It utilizes an SQLite database to store information. Bulk Club is designed for two kinds of users: store managers and system administrators. Managers can view sales and item inventory as well as customer transactions and account status. Administrators can make modifications to the database such as adding and removing customer store memberships or item information.

**PURPOSE:** This test plan is intended to accomplish two tasks: Verify that information is processed, stored, and retrieved properly, and to check that all user-facing features are functional and work as intended. Additionally, testing is necessary to catch errors or omissions not evident during the development process.

**SCOPE:** The following program aspects are subject to testing:

**- Database**

**- UI**

**- Manager and admin account functionality:**

1) Adding/removing customers (Admin)

2) Database manipulation (Admin)

3) Adding/removing items (Admin)

4) Sorting and viewing database information (Store Manager)

5) Any additional agile stories or features as identified by the Product Owner.

**DOCUMENTS:**

- Test log

- Trello project board (Agile stories)

- Daily sales files

**USER FEATURES TO BE TESTED:**

Black Box:

**NON-TESTED USER FEATURES:**

**-** Account creation: Admin and Manager accounts are created by developers. The database must be created first.

**STRATEGY:** Utilize black box testing. All testing shall conform to the following checklist:

1) The Scrum Master will perform testing or delegate tests to other developers.

2) Record name of function or feature to be tested.

3) Data input: Attempt bad inputs. These include out-of-range and boundary values.

4) UI: Click every button and check functionality against the description; for example, a button marked “Clear” should clear all inputs in line edits.

5) Note the result of each test (pass or fail). Multiple test attempts are allowed in case of tester error, but more than three consecutive failures must be marked as a Test Fail.

6) Give results to Product Owner to verify against the Agile Definition of Done.

7) The Product Owner will notify the developer of the tested function/feature if there are failures. The developer will attempt to correct the error. All results, pass or fail, are passed on the Scrum Master to enter into the test log.

8) Once corrections are made, the Scrum Master will be notified to resume or assign testing.

White Box testing will not be performed due to time and manpower constraints.

**ENTRY CRITERIA:** Database and basic UI must be created. Functions must be nominally operational. Once an agile story is complete, the developer will perform unit tests on their functions for that story and notify the Scrum Master to schedule testing. White box tests should be integrated into Travis CI.

**EXIT CRITERIA:** When all features pass the test criteria. Outputs must match product requirements as defined by the Product Owner and the definition of done for agile stories.

**SUSPENSION CRITERIA:** Program crashes or otherwise becomes unstable (multiple errors). Suspension will also occur when a new branch is set for release.

**TEAM ROLES:** Product Owner - Collects and verifies test results

Scrum Master - Maintains test log and performs testing

Developers – Perform testing.

**SCHEDULE:** Testing will commence in Sprint 2. Testing will continue barring suspensions. Testing must conclude 2 business days prior to release. If feasible, a unit testing framework will be integrated into Travis CI.

**TRAINING:** Must be familiar with following frameworks:

- C++

- Qt

- SQLite

- GitHub

- Agile Management

**ENVIRONMENT:** A computer (Windows, Mac OS, or Linux) with Qt. Note the project has been predominantly developed on Mac OS and thus will need to be re-compiled on Windows/Linux if downloaded directly from the GitHub.

**CONFIGURATION MANAGEMENT:** When an Agile story is complete, the developer will update their personal project branch on GitHub. The product owner is notified and checks the story’s Definition of Done by running the developer branch on his machine. If the story passes, then the product owner merges the story into the GitHub master branch. Travis CI launches automatically in the background, compiling and building the master branch. The product owner notifies the scrum master to add the agile story to the test log and schedule testing.

**DATABASE MANAGEMENT**: The database is written in SQLite. When the program is ran for the first time on a machine, the program checks for a database file named “bulk\_club.db” in the C:\USER directory (or OS equivalent). If the file is not, the program will create an SQLite database and populate it with information from files included in the Bulk Club master branch. All database interaction and modification is handled by the “Controller” class.

**DELIVERABLES:**

**-** Test log

- Agile stories

- UML diagrams

**APPROVAL PROCESS:** Product Owner reviews and validates all completed testing.

**GLOSSARY:**

Qt: A C++-based framework for developing programs with graphic user interfaces (GUI).

SQLite: A database language based on SQL. A database built in SQLite can be ran locally and does not require a server or network connection.

GitHub: A website for managing software development and version control. Allows multiple copies of a program to exist simultaneously.

Travis CI: Travis Continuous Integration. An external program that is linked to a project’s GitHub page. When a change is a made to a given branch, Travis will automatically compile and build the program to facilitate faster deployments and detect compilation and integration errors.

Trello: a website for managing task lists.

Black box: A method of testing program features by examining input and output. Emphasizes program requirements over efficiency or algorithm use.

White box: A method of testing program features by examining the algorithms and data flow within a program. Emphasizes how tasks are performed over whether such features meet the specifications.

Unit test: A test of a specific program function or class that checks for functionality.

OS: Operating System

UI: User interface. Often used interchangeably with Graphic User Interface (GUI).

Agile: A software development methodology that is designed to maximize programmer autonomy, efficiency, and responsiveness to change. SEE SCRUM.

Scrum: A variation of Agile development that dictates team organization and workflow. SEE AGILE.

Product Owner: A member of the Scrum team whose primary role to understand customer expectations and communicate them to the rest of the team. Verifies that the program features are meeting customer requirements via the Definition of Done.

Scrum Master: A member of the Scrum team whose primary role is to coordinate team efforts, remove barriers to productivity, and enforce Scrum/Agile standards and practices.

Developer: A member of the Scrum team who works to plan, build, and test the program.

Sprint: A period of time for software development within the Agile framework.

UML: Unified modeling language. A collection of abstract graphic representations of the program to visualize processes and algorithms.