**Simple Social Card Collection Battling Game**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Software Test Specification**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Version 1.0

11/07/2017

Lee Kelvin lxk201

Qianxiang Ma qxm28

**1. INTRODUCTION**

1.1 purpose

The purpose of this document is to layout the vision of how this software will be tested for release. It will go to detail regarding test cases and the performances that satisfy the requirements that is laid out in Software Requirement Specification.

1.2 Scope

This test document will contain brief descriptions of the test cases and the software components the test cases involved with. The documentation procedure of finding solution to the bugs during testing is also specified. This document also list out the features that needs to be tested, and test cases will have a description regarding usage, execution, and expected outcome.

1.3 Overview of Contents of Document

Test Plan Description

The goal of this section is to layout the test cases that will be used to test each feature of the program. A brief description of the test cases will be included.

Test Design Specification

This section lays out the approach of testing regarding this program. It will detail how a feature of the program passes/fails the test, and what kind of tests are going to be used. The features that will not be tested and the environment requirement of the software testing. The suspension/resumption criteria will also be described in this section.

Test Specification

This section contains detail information regard each test cases and the features that involves the test cases.

Requirement Traceability

This section provides information regarding test cases’ connect to the Software Requirement Specification.

2. Test Plan Description

2.1 Product Summary

This simple social card game allows players to chat and play a card battle game. Players may request battle with friend, or quick match with another player that is also searching for a quick match battle.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Feature | Use Case | Component | | |
|  | | View | Client | Server |
| Login | Log into an existing account | LoginWindow.java | LoginWindow.java | Server.java |
| Main Window | Main panel for the player to choose actions | MainWindow.java | Game.java | Server.java |
| Matchmaking | Finding a match for a player, then display the battle screen | /\* Not implemented \*/  MainWindow.java | /\* Not implemented \*/  NetClient.java | /\* Not implemented \*/  Server.java |
| Gameplay | This feature processes battle information generate during a game | Match.java  Card.java | NetClient.java | Server.java |
| Deck Editing | Allows user to create a deck to be used in game | MainWindow.java  Card.java | - | - |

|  |  |  |
| --- | --- | --- |
| Activity | Description | Exit Criteria |
| Alpha Testing | Two testers plays game against each other | Test all the possible actions, and report all undesirable behaviors. |
| Submission Testing | Two testers plays each other again before submitting the assignment | All the desired behaviors are present, and no glitches detected in the testing coverage. |

3. Testing Design Specification

3.1 Testing Approach

Tests are prepared according to SRS.

Interface Testing

This software involves interactions between cards and constructions and parsing of the designed transmission protocols. These objects have to be properly constructed and have their fields capable of reading out.

Sub-system Testing

Many of the system runs primarily independently for this software, such as Match and Chat windows. Passing tests on each sub-system can guarantee functionality of the whole.

Field Testing

Because this software is a game, inviting actual testers will provide feedbacks on the controllability, latency, and other aspects.

Acceptance Testing

A Game is meant to be played, and an acceptable game means this game must worth playing.

3.2 Features not tested

Load Test: The stress test for server’s ability to handle internet traffic surge, unusually high volume, is not of important interest. This feature will not be tested.

Security Test: The ability to pass messages between server and client without a third party’s ability to understand the message, if intercepted, and the ability to detect tampering attempt to the message by the third party. The project size does not require security feature, so this feature will not be tested.

Server Reliability: server’s up and operational time will not be tested due to the human resource limitation and time constraint.

Session Timeout/Disconnect: SRS assumes an ideal internet environment. Timeout/disconnect is considered as actively going offline.

3.3 Environment Needs

This section describe the environmental requirement to operate this software.

Client:

* The environment must have internet connection.
* The environment must install Java running environment.
* Standard keyboard and mouse.
* Standard VGA display.

Server:

* The environment must have internet connection.
* The environment must install Java running environment.
* The server must have administrative privilege to open sockets to accept connection.

3.4 Suspension / Resumption Criteria

If a test somehow suspends before completion, then it will resume from step 1. If a test cannot be completed due to continuous application failure, it will be noted in the report.

4. Test Specification

4.1 Login

Login is a window that allows user to input his username and password, and sending them to server. Successful login should also redirect user to mainwindow.

Client side: Login components sends a packet packed using protocol.Login, and retrieves a packed packed using protocol.LoginResult and parse the results.

Server side: Server receives a Login packet and unpacks it verifying its credentials, and pack a LoginResult packet sending it back to client.

Automated tests focuses on Login and LoginResult protocol packing and unpacking process, credential verification and results parsing.

Automated Testing

|  |  |
| --- | --- |
| Test Case ID | STS - 1 |
| Test Name | TestLogin |
| Description | Ensures Login protocol functionality and results reflection. |
| Prerequisites | N/A |
| Test Environment | Server with Login implementation, protocol package |
| Test Strategy | Interface testing, subsystem testing |

Automated Test Description

|  |  |  |
| --- | --- | --- |
| Test | Description | Expected Result |
| 1 | Incomplete fields | No packet sent, notification pops |
| 2 | Completed fields, incorrect credentials | Legitimate protocol, LoginResult shows failing |
| 3 | Completed fields, invalid inputs | No packet sent, notification pops |
| 4 | Completed fields, correct credentials | Legitimate protocol, LoginResult shows success |

Manual Testing:

|  |  |
| --- | --- |
| Test Case ID | STS - 2 |
| Test Name | TestLoginWindow |
| Description | Ensures LoginWindow GUI functionality |
| Prerequisites | N/A |
| Test Environment | N/A |
| Test Strategy | Field testing |

Manual Test Description

|  |  |  |
| --- | --- | --- |
| Test | Description | Expected Result |
| 1 | Window Resizing | All fields remains clear to view |
| 2 | Window Closing | Completely shuts down the program |

4.2 MainWindow

Mainwindow is the main component that allows user to access different aspects of this game, such as view friend list, start a quickmatch, and edit his deck.

Client side: Retrieves message and battle requests from other players and display them, calls edit deck and quickmatch components, and send message and battle requests to others

Server side: Process retrieval and sending requests initiated from client.

Automated Testing

|  |  |
| --- | --- |
| Test Case ID | STS - 3 |
| Test Name | TestMessageRetrieve |
| Description | Ensures Message (also includes battle requests) and Retrieve protocol functionality and results reflection. |
| Prerequisites | N/A |
| Test Environment | Server with Message, User implementation, protocol package |
| Test Strategy | Interface testing, subsystem testing |

Automated Test Description

|  |  |  |
| --- | --- | --- |
| Test | Description | Expected Result |
| 1 | Single Message Client Reflection | Message Displayed |
| 2 | Message Box Client Reflection | All Messages Displayed |
| 3 | Single Battle Request Client Reflection | Battle Request Displayed |
| 4 | Message Construction | Legitimate message is constructed using protocol |
| 5 | Battle Request Construction | Legitimate message is constructed using protocol |

Manual Testing:

|  |  |
| --- | --- |
| Test Case ID | STS - 4 |
| Test Name | TestMainWindow |
| Description | Ensures MainWindow GUI functionality |
| Prerequisites | N/A |
| Test Environment | N/A |
| Test Strategy | Field testing |

Manual Test Description

|  |  |  |
| --- | --- | --- |
| Test | Description | Expected Result |
| 1 | Window Resizing | All fields remains clear to view |
| 2 | Window Closing | Completely shuts down the program |
| 3 | QuickMatch Button | Loads quickmatch panel |
| 4 | View Friends Button | Loads friends panel |
| 5 | Deck Edit Button | Loads deck editing panel |

4.3 Matchmaking

A component that can match players into games.

Client-side: Match info retrieval, and reflection.

Server-side: Respond to retrievals, and create match instances.

Automated Testing

|  |  |
| --- | --- |
| Test Case ID | STS - 5 |
| Test Name | TestMatchMaking |
| Description | Ensures protocol functionality and results reflection. |
| Prerequisites | N/A |
| Test Environment | Server with matchmaking implementation, protocol package |
| Test Strategy | Interface testing, subsystem testing |

Automated Test Description

|  |  |  |
| --- | --- | --- |
| Test | Description | Expected Result |
| 1 | Match Info Reflection | Direct into a match, or keep waiting |
| 2 | Retrieval Construction | Legitimate retrieval packet |
| 3 | Acceptance and Rejection message construction | Legitimate confirmation packet |

4.4 Gameplay

The actual match between two players.

Client side: Respond to player actions, record them and send to server. Also parse opponent’s actions and reflect them on board.

Server side: Respond to retrieval requests and record actions sent from client into a buffer.

Automated Testing

|  |  |
| --- | --- |
| Test Case ID | STS - 6 |
| Test Name | TestGameplay |
| Description | Ensures board functionality, card interactions, and protocol legitimacy. |
| Prerequisites | N/A |
| Test Environment | Server with Action, Match implementation, protocol package |
| Test Strategy | Interface testing, subsystem testing |

Automated Test Description

|  |  |  |
| --- | --- | --- |
| Test | Description | Expected Result |
| 1 | Action construction | Legitimate action packet |
| 2 | Retrieval Construction | Legitimate retrieval packet |
| 3 | Retrieval Reflection | Opponent’s action reflected |
| 4 | Card interaction | Summon, Skill Activation can properly trigger |
| 5 | Pre-turn and aft-turn checks | Reflect mp regeneration and SP, VP generation and consumption |
| 6 | Victory checks | Can determine victory conditions |

Manual Testing:

|  |  |
| --- | --- |
| Test Case ID | STS - 7 |
| Test Name | TestGameActions |
| Description | Ensures Match window can record players actions and send them to server. |
| Prerequisites | N/A |
| Test Environment | N/A |
| Test Strategy | Field testing |

Manual Test Description

|  |  |  |
| --- | --- | --- |
| Test | Description | Expected Result |
| 1 | Window Resizing | All fields remains clear to view |
| 2 | End turn | An End turn action sent |
| 3 | Surrender and Window closing | An surrender action sent and closes the match |
| 4 | Summon and skill activation | Corresponding actions sent with options on targets |

4.5 Deck Editing

Building a deck of user’s own.

Client side: adding and removing cards from the current deck.

Automated Testing

|  |  |
| --- | --- |
| Test Case ID | STS - 8 |
| Test Name | TestDeckEditing |
| Description | Ensures Deck Editing functionality |
| Prerequisites | N/A |
| Test Environment | N/A |
| Test Strategy | Interface testing, subsystem testing |

Automated Test Description

|  |  |  |
| --- | --- | --- |
| Test | Description | Expected Result |
| 1 | Card addition when cards < 15 | Card added |
| 2 | Card removal when that card > 0 | Card removed |
| 3 | Card addition when cards = 15 | Rejection |
| 4 | Card removal when that card = 0 | Rejection |
| 5 | Leaving | Deck saved |

5. REQUIREMENT TRACEABILITY

|  |  |  |
| --- | --- | --- |
| Use Case | System Test ID | Design Component |
| 4.1 Login | STS-1:log in information process (Automated)  STS-2:login window event handling (Manual) | Login.java |
| 4.2 MainWindow | STS-3: message processing, sending, and receiving(Automated)  STS-4: MainWindow event handling (Manual) | MainWindow.java |
| 4.3Matchmaking | STS-5: match information response and process(Automated) | Game.java |
| 4.4 Gameplay | STS-6: game play information process(Automated)  STS-7: player action process and window event handling (Manual) | MainWindow.java  Match.java  Card.java  Game.java |
| 4.5 Deck Editing | STS-8: processes information generated from a player building a deck | MainWindow.java |