Software Requirements Specification

Multi User Game System – Blackjack

Team 6

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Revision** | **Description** | **Author** |
| 06/14/2020 | 1.1 | Section 1 | Cory Yang |
| 06/14/2020 | 1.2 | Section 4 | Angelo Rosario |
| 06/14/2020 | 1.3 | Section 3 | Guodong Lu |
| 06/15/2020 | 1.4 | Section 2 | Michael Maddux |
| 7/14/2020 | 1.5 | Updated requirements for Server/Client interaction | Michael Maddux |
| 7/21/2020 | 1.6 | Updated requirements to reflect console build | Michael Maddux |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

1. Purpose.......................................................................................................................................... 4

1.1. Scope............................................................................................................................................. 4

1.2. Definitions, Acronyms, Abbreviations............................................................................................ 4

1.3. References.................................................................................................................................... 4

1.4. Overview..................................................................................................................................... .9

2. Overall Description........................................................................................................................ 10

2.1. Product Perspective......................................................................................................................10

2.2. Product Architecture......................................................................................................................10

2.3. Product Functionality/Features......................................................................................................11

2.4. Constraints.....................................................................................................................................11

2.5. Assumptions and Dependencies.................................................................................................. 11

3. Specific Requirements...................................................................................................................12

3.1. Functional Requirements...............................................................................................................12

3.2. External Interface Requirements...................................................................................................12

3.3. Internal Interface Requirements.....................................................................................................13

4. Non-Functional Requirements...................................................................................................... 14

4.1. Security and Privacy Requirements...............................................................................................14

4.2. Environmental Requirements.........................................................................................................14

4.3. Performance Requirements...........................................................................................................14

**1.**             **Purpose**

This document outlines the requirements for the Multi User Game System - Blackjack (MUGSB).

**1.1.**          **Scope**

This document will catalog the user, system, and hardware requirements for MUGSB. It will not, however, document how these requirements will be implemented.

**1.2.**          **Definitions, Acronyms, Abbreviations**

Client - Means by which user connects to server and plays with other users. Keeps track of client sessions, allows user to log in, only allows users to access authorized data.

Server - Keeps track of client sessions, maintains security/logs.

Deck - 52 cards, split with 4 suits of 13 cards each

Table – A singular instance of the Blackjack module. Represents one active game of Blackjack consisting of 1 to 4 users, of which there can by many running at once.

**1.3.**          **References**

**1.3.1 Use Cases**

Use Case ID: 1.1

Use Case Name: User Log in

Relevant Requirements: Refer to section 1.2 of SRS for definitions

Primary Actor: User

Pre-Conditions: User already has username/password

Post-Conditions: User logged in and assigned to a game of Blackjack

Basic Flow or Main Scenario:

1. User opens client
2. User enters username/password
3. Server verifies if credentials are correct
4. Server logs the user in
5. Server automatically assigns the User to a game of Blackjack

Extensions or Alternate Flows:

1. User does not yet have a username/password
   1. Server sets up new account for user

Exceptions:

1. User enters incorrect username/password

Related Use Cases:

1. User enters incorrect username/password
2. User logs out
3. User sets up new account
4. User access logs/match history

Use Case ID: 1.2

Use Case Name: User Log Out

Relevant Requirements: Refer to section 1.2 of SRS for definitions

Primary Actor: User

Pre-Conditions: User ready to stop playing

Post-Conditions: User has exited server/client and will need to log in again

Basic Flow or Main Scenario:

1. User exits the client or presses “log out”
2. Connection with Server terminates

Extensions or Alternate Flows: N/A

Exceptions: N/A

Related Use Cases:

1. User logs in

Use Case ID: 1.3

Use Case Name: User Creates New Account

Relevant Requirements: Refer to section 1.2 of SRS for definitions

Primary Actor: User

Pre-Conditions: User does not have existing account

Post-Conditions: User now has account

Basic Flow or Main Scenario:

1. User opens client
2. User does not yet have user/pass to log in
3. User creates new account
4. User may log in with new account credentials

Extensions or Alternate Flows:

1. User plays game
2. User log out

Exceptions:

1. User tries to create account using username that is already in use

Related Use Cases:

1. User Logs in

Use Case ID: 2.1

Use Case Name: Server keeps track of client sessions

Relevant Requirements: Refer to section 1.2 of SRS for definitions

Primary Actor: Server

Pre-Conditions: Users play

Post-Conditions: Session information stored

Basic Flow or Main Scenario:

1. Users log in
2. Users enter game and start playing
3. Server keeps logs of match history and security events

Extensions or Alternate Flows:

Exceptions:

1. User has never played on this account before, no logs

Related Use Cases:

Use Case ID: 2.2

Use Case Name: Server groups Users together to play

Relevant Requirements: Refer to section 1.2 of SRS for definitions

Primary Actor: Server

Pre-Conditions: Enough players are logged on for multiple rooms

Post-Conditions: Multiple rooms/sessions are established

Basic Flow or Main Scenario:

1. Users log in looking to play the game
2. Users automatically allocated to a room
3. Eventually more users want to play than room occupancy allows
4. A new room is formed, new players are put into that room

Extensions or Alternate Flows:

1. Rooms are closed and when people leave

Exceptions:

1. Not enough people ever play to create more than one room

Related Use Cases:

1. Users log in
2. Users log out

Use Case ID: 3.1

Use Case Name: User Wins/Loses

Relevant Requirements: Refer to section 1.2 of SRS for definitions

Primary Actor: User

Pre-Conditions: User logged in

Post-Conditions: Server records win

Basic Flow or Main Scenario:

1. User enters room with other users
2. User plays a round
3. User wins/loses
4. Server updates logs with new balance and round summary

Extensions or Alternate Flows:

1. Server records session

Exceptions: N/A

Related Use Cases:

1. Server keeps track of client sessions

**1.4.**          **Overview**

The Multi User Game System - Blackjack, is designed to run multiple games of Blackjack at once. This game will match users together to play against one another and keep records of match history.

**2.**             **Overall Description**

**2.1.**          **Product Perspective**

This Multi-User Game System for Blackjack is meant to provide its users a way to play a game for entertainment with other people online. Currently, it is meant to be provided for free, without real money being bet, however “earnings” will be tracked, alongside recent game history, in order to give players a sense of progression and information about their own development as Blackjack players.

**2.2.**          **Product Architecture**

The system will be organized into 3 major modules: the server module, the client module, and the Blackjack module.

The server module contains a singleton Server which registers new users or logs in existing users with a valid username and password combination, both of which connect via the client module. Active Client Sessions are tracked and maintained by the Server. The Server module has access to the User, Security Log, and Event Log files, which are maintained on the Server’s machine. The Server will place players into existing Blackjack tables or create new instances as needed.

The client module connects to the Server, by providing a valid username and password combination, or by creating a new account, following which their chosen username and password will be added to the list of valid username and password combinations. A client can only access authorized information linked to their specific username. The client module will be responsible for displaying information received from the server module and Blackjack module. All login attempts, successful or otherwise, will be logged in the Security Log.

The Blackjack module will handle the functionality of a singular instance of a Blackjack game, or table. There may be up to four Clients per table. Each Blackjack game will begin with a Deck of 52 unique cards. As cards are dealt, they are removed from the deck. The same Deck will be used until a round is set to begin and the Deck has less than 26 cards, after which a new Deck will be created and used for the table. The Blackjack module handles the logic for the dealer, bets, and determining wins and losses. Game state information, such as hands and bets, are handled in the Blackjack module, which is then passed to the Clients associated with the particular Blackjack module instance to display. After each round, the Blackjack module will communicate with the Server in order to update Client earnings and history in the User file and to send a log of the round to update the Events Log.

**2.3.**          **Product Functionality/Features**

The high-level features of the system are as follows:

Clients can connect and play games of Blackjack online with up to three other people.

The Server can create new instances of Blackjack tables to allow for many unique instances of Blackjack to be played simultaneously.

Clients can create and log in to existing accounts to play games of Blackjack

Encryption for passwords is used for security.

Event logs of games and login attempts are maintained by the server.

**2.4.**          **Constraints**

Each Client will be required to run the Client program on their system in order to connect to the singular Server.

All code will be in Java.

A singular Server will interact with all Clients, therefore the IP address of the Server must be known by each Client, or preset into the Client program for a known, fixed server.

**2.5.**          **Assumptions and Dependencies**

It is assumed the maximum number of users at a given time is 50.

It is assumed the maximum number of registered users is 500.

The singular Server must be running in order for any Client to connect and use the system.

A Client must remain connected to the Server during play.

**3.**             **Specific Requirements**

**3.1.**          **Functional Requirements**

**3.1.1.**     **Common Requirements:**

3.1.1.1 The system will establish a Diffie-Hellman key exchange upon initial connection between Server and Client. Thereafter, any sensitive information, hashed passwords, will be encrypted.

3.1.1.2 The shared secret established by the key exchange will be used to encrypt information using AES.

3.1.1.3 The shared secret will be unique for every thread and transient.

**3.1.2.**     **Server Module Requirements:**

3.1.2.1 The Server Module should be able to handle up to 50 threads simultaneously

3.1.2.2 The Server Module should maintain logs of login and registering attempts

3.1.2.3 The Server Module should maintain the logs of Users and load them upon startup

**3.1.3.**     **Client Module Requirements:**

3.1.3.1 The Client module should provide a GUI for abstraction of connection with the Server

3.1.3.2 The Client module should allow for playing with other players solely through the connection with the Server.

**3.1.4.**     **Blackjack Module Requirements:**

3.1.4.1 The Blackjack system should allow for the Clients to play a form of the game of Blackjack

3.1.4.2 The Blackjack system should provide an automated dealer

3.1.4.3 The Blackjack system should provide Messages to the Server to update User balances

3.1.4.4 The Blackjack system will have a deck of 52 cards, with a new one being used every round.

**3.2.**          **External Interface Requirements**

3.2.1 The system should provide an interface to the user login system, so the player can know where they can enter the username and password. The interface should be intuitive, with labeled buttons that correspond to their action. Feedback should be provided upon actions.

3.2.2. The system should provide an interface to the user during gameplay. It should display output from the system and take user input in the console.

**3.3.**          **Internal Interface Requirements**

3.3.1  The system must process a data-feed from the user login system that holds all user's login history. The feed will be in the form of slash-separated text data. The subject included in the file contains the user name, hashed password and balance.

3.3.2 The system must process a data-feed from the Blackjack system that holds game history. The feed will be in the form of slash-separated text data. The subject included in the file contains the cards played and money won or lost.

**4.**             **Non-Functional Requirements**

**4.1.**          **Security and Privacy Requirements**

4.1.1 The System must encrypt sensitive data being transmitted over the Internet.

4.1.2 The user must set their own password when creating their new account.

4.1.3 Any information transferred over the internet will not be changed.

4.1.4 Passwords will never be shown back to any user for any reason.

4.1.4 Passwords will be hashed with the username for additional security.

4.1.5 Diffie-Hellman shared secrets will be re-established upon every connection

**4.2.**          **Environmental Requirements**

4.2.1 The system will be portable so it will be playable regardless of OS.

4.2.2 The system will be implemented in Java.

4.2.3. Clients should be able to access the system on the network.

**4.3.**          **Performance Requirements**

4.3.1 Users will be able to perform any action with minimal wait time.

4.3.2 The system will progress to the next turn of the table no more than 1 second after the active player has picked an option for their hand.

4.3.3 The system will be interactive, and any delays involved with it will be minimal and much less than any interaction available.

4.3.4 The system will traverse quickly between its states and will perform accordingly with its actions.

4.3.4 Connecting to the server will not have a delay longer than 10 seconds.

4.3.5 The system will be reliable and provide the proper tools for their respective operations