

Software Requirements Specification

for

Blackjack

Version 1.0 approved

Prepared by Severin Klapproth, Flavia Taras, Luca Wolfart,
Pascal Engeler, Noé Canevascini, Stanislaw Piasecki

Jack Blacks

10.03.2022

Table of Contents

Introduction	1
Purpose	1
Document Conventions	1
Intended Audience and Reading Suggestions	1
Product Scope	1
References	1
Overall Description	1
Product Perspective	1
Product Functions	2
User Classes and Characteristics	2
Operating Environment	2
Design and Implementation Constraints	2
User Documentation	3
Assumptions and Dependencies	3
External Interfaces and Requirements	3
User Interface	3
Communications Interfaces	6
System Requirements	7
Functional Requirements	7
FREQ-1: Game Server	7
FREQ-2: Connection	7
FREQ-3: Lobby	7
FREQ-4: GUI	7
FREQ-5: Game Start	7
FREQ-6: Round Start	8
FREQ-7: Turns	8
FREQ-8: Insurance	8
FREQ-9: Make a Move	8
FREQ-10: User Input	9
FREQ-11: Dealer's turn / Dealer AI	9
FREQ-12: Round End & Returns	9
FREQ-13: Game End	10

FREQ-14: User Messages/Restrictions	10
FREQ-15: Player Exit	10
System Scenarios	11
Use-case Diagrams	11
Scenarios	11
SCN-1: Starting a game	11
SCN-2: Starting a round	12
SCN-3: Playing a turn	13
SCN-4: Ending a round	14
SCN-5: End of the game	16
System Constraints	18
Important Nonfunctional Requirements	18
NFREQ-1: Response time	18
NFREQ-2: Game Cards and Layout	18
NFREQ-3: Language	18
NFREQ-4: Reliability	18
NFREQ-5: Ease of use	18
Other Requirements	19

Revision History

Name	Date	Release Description	Version
Felix Friedrich	18.03.22	Template for Software Engineering Course in ETHZ.	0.3
Jack Blacks	18.03.22	Requirements Specification for Blackjack	1.0

Introduction

Purpose

<Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem.>

Document Conventions

<Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.>

Intended Audience and Reading Suggestions

<Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.>

Product Scope

<Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here.>

References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

Overall Description

Product Perspective

<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that

shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>

Product Functions

This project implements a slightly modified version of the card game Blackjack. This implementation adapts it to be a “real-money free” multiplayer game for 2-5 people with clearly defined win conditions. The provided functionalities will be to:

- 1) Allow players to host a game on a server and choose the maximal number of rounds they want to play.*
- 2) Allow players to join hosted games using the client provided by the software.*
- 3) Start a game if enough players joined the host (at least one).*
- 4) Track and inform each player about the current state of the game (dealt cards, amount of money each player has, round count, bet sizes).*
- 5) Allow players to interact with the game interface using a GUI.*
- 6) Allow players to make bets before each round or make insurance bets in the case where the dealer is dealt an ace.*
- 7) Take turns, where the player can choose to either “hit”, “stand”, “insure”, “split” or “double-down”, but only if this move does not violate the game rules.*
- 8) Notify the player if their count goes above 21 during their turn.*
- 9) Have the dealer be controlled by an AI, representing the “casino”.*
- 10) Notify the player when they have no money left (cannot afford minimal bet), which means they lost.*
- 11) Start and end rounds until the round cap set by the lobby host is reached. Then, the winner is determined (if there are still two or more players in the game, the one with the most money wins).*
- 12) Notify all players if the winning condition is fulfilled for some player (they are the only player left in the game).*
- 13) End the game.*

User Classes and Characteristics

User class – A regular player:

- 1) Knows and understands the standard rules of Blackjack.*
- 2) Can access all features of the software.*
- 3) Has no technical knowledge besides understanding how to run an executable file on a UNIX system, how to use the GUI to interact with the game and how to start/join games by creating/connecting to a server.*

Operating Environment

The game runs on any modern computer with a UNIX-based system.

Design and Implementation Constraints

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).>

User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

External Interfaces and Requirements

User Interface

The interaction with the game happens using a Graphical User Interface (GUI). This will provide a login menu and a game interface. The interaction between the game and the user can be done through clicking on buttons or filling in the appropriate text boxes.

In the login menu the following inputs are available/can be made:

- Start and name a game.
- Choose a username.
- Option to join a game.
- Boxes to enter existing server information.

The game interface must provide the following:

- Usernames of the other players.
- Amount of money each player has.
- Current bet sizes.
- Visible cards drawn by all players.
- Clear display of whose turn it is.
- The possible moves in each player's turn.

Figures 1-6 are sketches of the different interfaces a user can see while playing the game. The different events which the user must be made aware of, winning or losing a hand, playing and having the possibility of leaving the game are all correctly displayed. If a player loses all their money, they cannot continue playing and must leave the game.

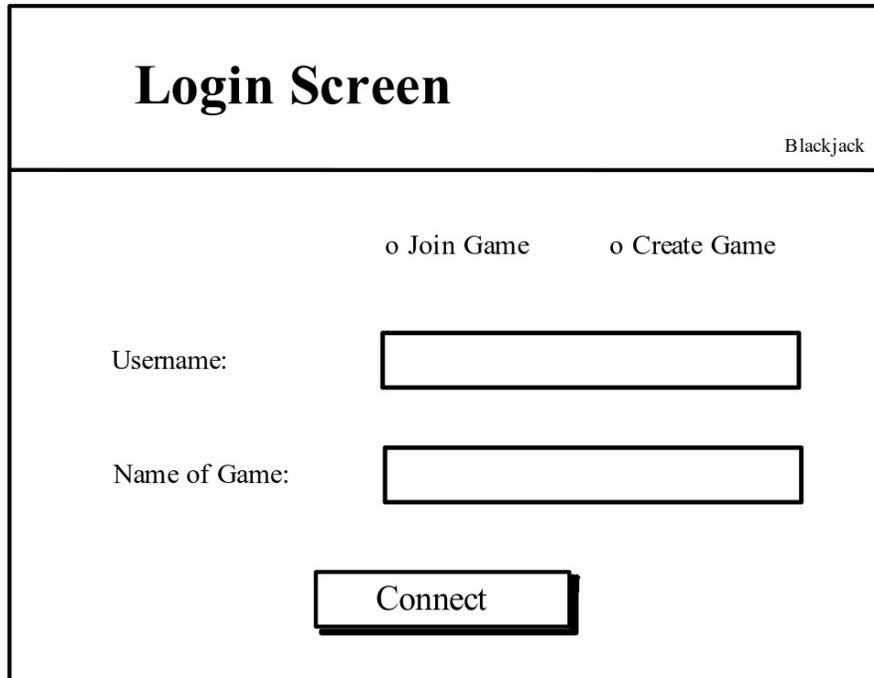


Figure 1: The login screen.

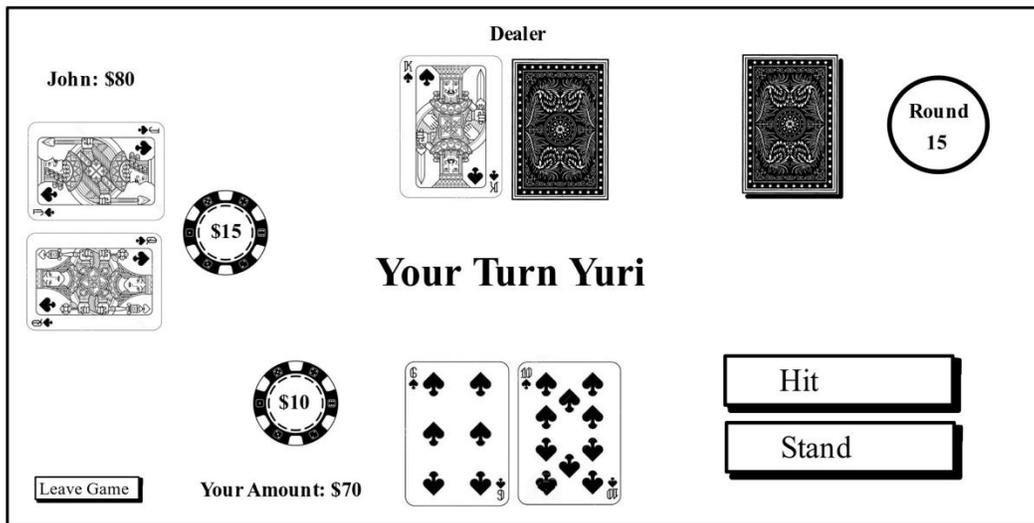


Figure 2: Player's perspective when it is their turn to make a move.



Figure 3: The player has run out of money and has lost. Their only option is to leave the game.

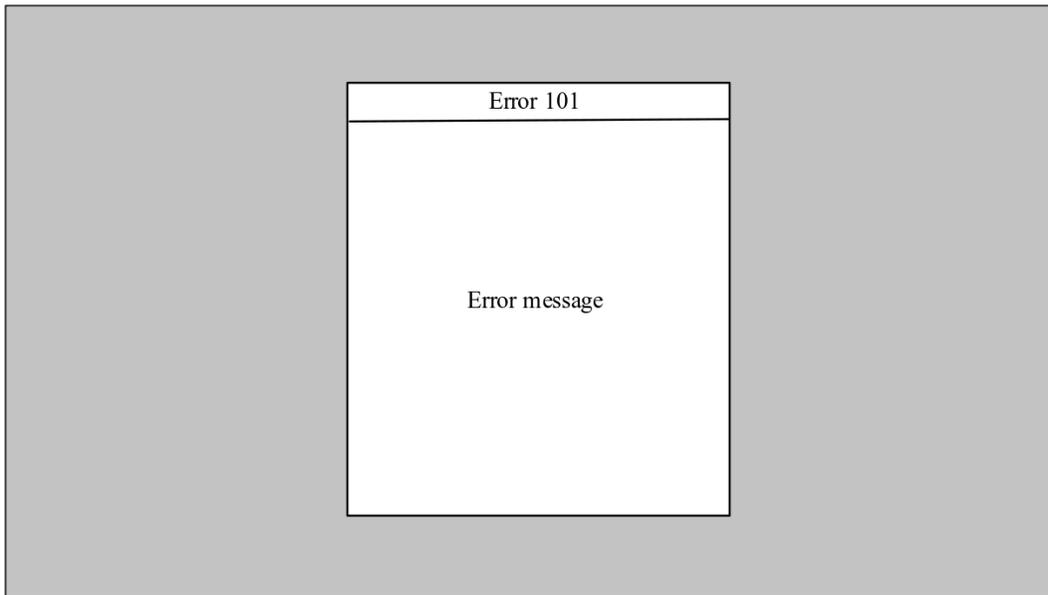


Figure 4: An example of an error.



Figure 5: The player won the round and receives 15\$.



Figure 6: The player lost the round and their bet.

Communications Interfaces

The interactions between the client and the server will be handled as follows: the communication will be established between one server and multiple clients. Their communications will adhere to the TCP/IP protocol suite. The JSON format will be used to encode the data.

System Requirements

Functional Requirements

FREQ-1: Game Server

The system runs a server to which up to five players can connect via the provided client.

User Priority (5/5): *the game cannot be played without this requirement.*

Technical Priority (5/5): *the game cannot be played without this requirement.*

FREQ-2: Connection

The system provides a client which allows players to connect to the game lobby/server.

User Priority (5/5): *the users cannot join without this requirement.*

Technical Priority (5/5): *the game cannot be played without this requirement.*

FREQ-3: Lobby

The system has a game lobby which can host up to five players at the same time. The lobby displays the currently joined players and allows the game to start if and only if at least two players are inside. The first player to join the lobby is considered the host.

User Priority (5/5): *the players need to be able to wait for each other before starting the game.*

Technical Priority (2/5): *the game cannot be played without at least two players. However, the lobby is technically not necessary to play the game.*

FREQ-4: GUI

The system provides a GUI, displaying the game state (player and dealer cards, player pots, total player budgets), and possible actions (hit, stand, insure, split, double down).

User Priority (5/5): *makes it easier for the players to track the game.*

Technical Priority (2/5): *not strictly necessary in order to run the game.*

FREQ-5: Game Start

The first round is initialized once the lobby host decides to start the game. Each player is provided with 100\$ as a starting amount. A round counter is initialized.

User Priority (5/5): *the game cannot be started without this requirement.*

Technical Priority (5/5): *the game cannot be played without this requirement.*

FREQ-6: Round Start

Each player must place a bet or leave the game. The value of the bet must be an integer, at least 5\$, and is capped by each player's remaining money. Subsequently, each player receives two up cards, visible to all other players. The dealer receives one up card and one hole card. Players alternatingly start rounds, in clockwise order.

User Priority (5/5): *This requirement is necessary to play rounds.*

Technical Priority (5/5): *The game cannot be played without this requirement.*

FREQ-7: Turns

The system keeps track of the player state and displays whose turn it is. The players perform actions in clockwise order. Only the player whose turn it is can choose to perform any actions (hit, stand, insure, split, double down). After their first move, they can repeatedly "hit" until they choose to "stand" or the sum of their card values exceeds 21.

User Priority (4/5): *this makes it easier to keep track of whose turn it is.*

Technical Priority (2/5): *the users could keep track of this by themselves.*

FREQ-8: Insurance

In the case that the up card of the dealer is an ace, each player can decide whether they want to place an insurance bet.

User Priority (3/5): *this feature makes the game more enjoyable for users*

Technical Priority (0/5): *this is not relevant for the game functionality.*

FREQ-9: Make a Move

During their turn, the system allows the player to repeatedly make a legal move until they decide to stand. After each legal move the game state is updated accordingly for everyone. Possible legal moves are:

- **Hit** (Take another card)
- **Stand** (Take no more cards and end the turn)
- **Split** (Create two hands from a starting hand where both cards are of the same rank. Each new hand gets another card so that the player has two starting hands. This requires an additional bet on the second hand. The two hands are played out independently and sequentially, and the wager on each hand is won or lost independently.)
- **Double Down** (Increase the initial bet by 100% and take exactly one more card. The additional bet is placed next to the original bet. This can only be done at the beginning of a player's turn.)

- **Insurance** (This is a side bet that the dealer has blackjack and can only be placed if the dealer's visible card is an ace. Insurance bets are equal to half of the player's current bet. In case the dealer has a blackjack, insurance pays two to one.)

(Sources for Player Decisions: https://en.wikipedia.org/wiki/Blackjack#Player_decisions)

User Priority (5/5): *the game cannot be played without this requirement.*

Technical Priority (5/5): *the game cannot be played without this requirement.*

FREQ-10: User Input

The system allows the players to make inputs by clicking on the displayed move options in the GUI. It also allows each player to determine the value of the bet they want to place by interacting with the GUI through keyboard input/clicking.

User Priority (4/5): *this is very convenient.*

Technical Priority (1/5): *the inputs could also be delivered in a different way.*

FREQ-11: Dealer's turn / Dealer AI

After all players finish their turn, a program/AI decides the following moves of the dealer. First, the hole card of the dealer is now turned and visible to all players. They will continue to draw cards (hit) until they have 17 points or more. After that they will stand.

The dealer never splits or doubles down.

User Priority (5/5): *the game cannot be played without this requirement.*

Technical Priority (5/5): *the game cannot be played without this requirement.*

FREQ-12: Round End & Returns

Once the dealer has ended their turn, the system calculates the return rates on the bets of every player. At this point, the system also needs to increment the internal round counter.

Bets have the following returns:

- If the player went over 21, had less points than the dealer or if the dealer had a blackjack and the player didn't, the player loses the money they bet.
- If the player ties with the dealer (same number of points or both dealer and player have blackjacks), the player receives their main bet back.
- If the player wins by having more points than the dealer, they receive two times their main bet.
- If the player wins with a blackjack, they receive 1.5 times their main bet.

Insurance:

- If the player made the insurance bet at the beginning of their turn and the dealer has blackjack he receives two times the insured amount. Otherwise, he loses the money invested in the insurance bet.

After having made the calculations, the system updates the budget of every player and displays the winnings of each player until a new round starts.

User Priority (5/5): *the game cannot continue without this requirement.*

Technical Priority (5/5): *the game cannot be played without this requirement.*

FREQ-13: Game End

After each round the system checks...

- ... if a player went bankrupt. If so, they have lost and they have to leave the game.
- ... if all but one player went bankrupt. If yes, the game ends and the only player who has money left is declared the winner.
- ... if the internal round counter has reached the chosen maximum number of rounds. If so, the game ends and the player with the most money wins.

User Priority (4/5): *this is an important part of the game flow and it is convenient for the players if the system keeps track of this.*

Technical Priority (2/5): *the game needs to end at some point, but the users could keep track of this event by themselves.*

FREQ-14: User Messages/Restrictions

The system restricts the player by changing the mouse-interaction options inside the GUI, depending on their cards and budget, such that the selection of illegal moves is not even a possibility.

If the user tries to enter an illegal move via keyboard or mouse the system displays an error message.

User Priority (3/5): *gives the user more information why something is not possible.*

Technical Priority (1/5): *not necessary for the game to run.*

FREQ-15: Player Exit

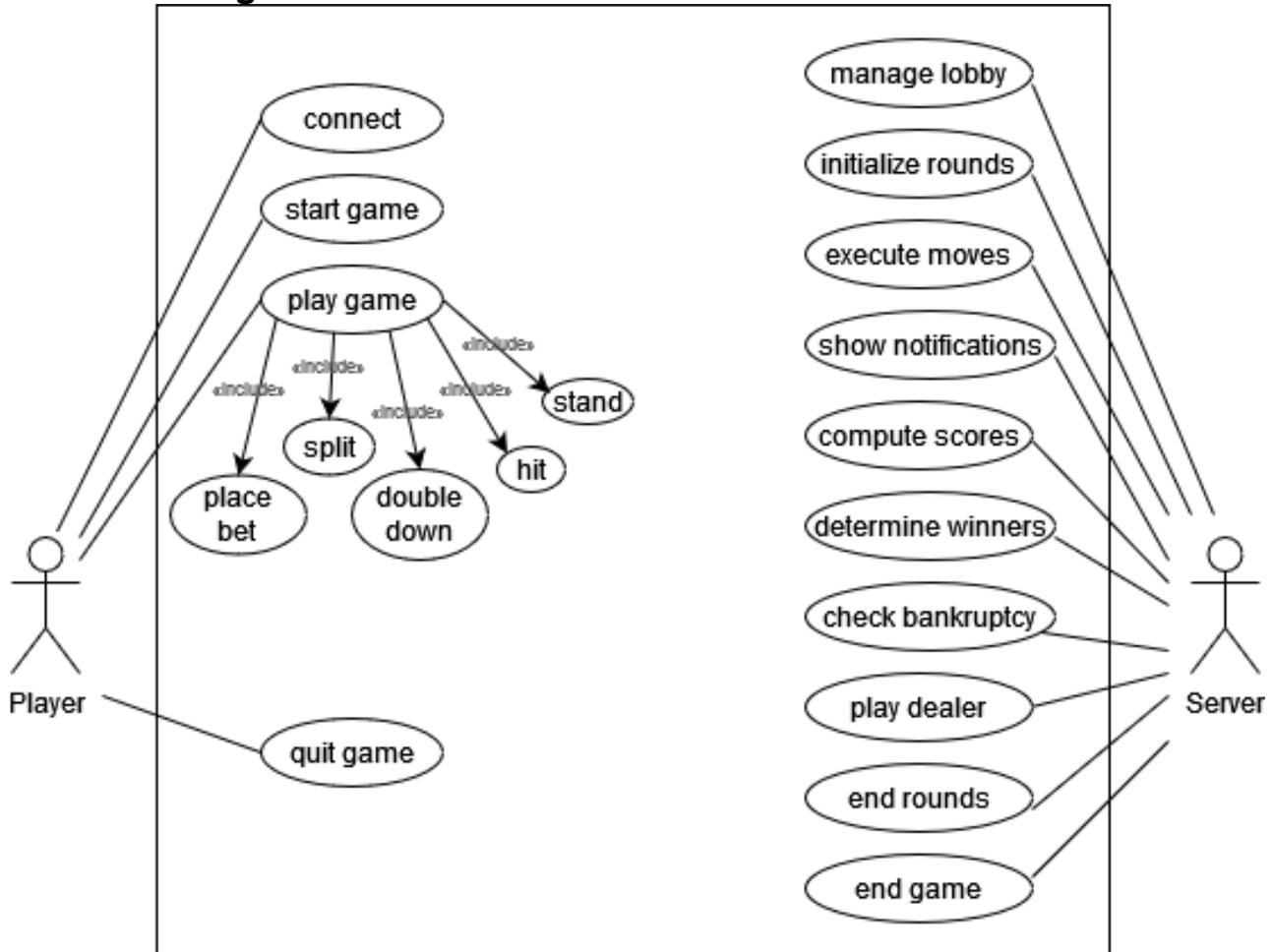
The system continues running even if one or more of the game clients exit, crash or disconnect.

User Priority (3/5): *it is inconvenient if your client crashes because someone else disconnected.*

Technical Priority (3/5): *necessary for the system in order to keep the game state, but the state might be irrelevant if the game cannot continue due to a missing player.*

System Scenarios

Use-case Diagrams



Scenarios

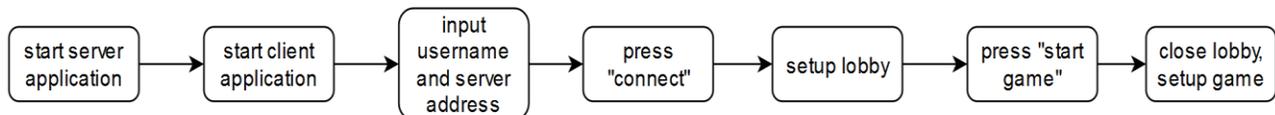
SCN-1: Starting a game

FREQ reference	1,2,3,4,5,6,10
NFREQ reference	1,2,3,4,5
Short Description:	John, Jane and Peter want to play a game of Blackjack. John starts the server and communicates the address to Jane and Peter. All three start their clients, enter their usernames and connect to the server by filling in the address and pressing the “connect” button. Jane presses the “start game” button.
Activation action:	John starts the server, all players connect, Jane presses the “start game” button.
Precondition:	All players have the ability to run and control the game client, John

	also has the ability to run the server. All players have access to the same network as the server.
--	--

Basic flow: Starting a game		
Step	User action	System response
1	John starts the server application	The server application starts and displays its address.
2	Players start the client	The client application starts and shows text input fields for username, and for a server address
3	Players fill in the required information	
4	Players press the “connect” button	All players are connected to the server and added to the lobby, the connected players are displayed
5	Jane presses the “start game” button	The lobby is closed, a new game is initialized with all players that were present in the lobby, the amount of money of each player is set to 100, a shoe is initialized, the round counter is set to 0, players are assigned seats and the game starts by prompting each player to specify a bet.
Post-condition:		All players are connected to the server, the game is initialized and players have the ability to specify their desired bets for the first round.

Scenario Diagram for SCN-1 Starting a game

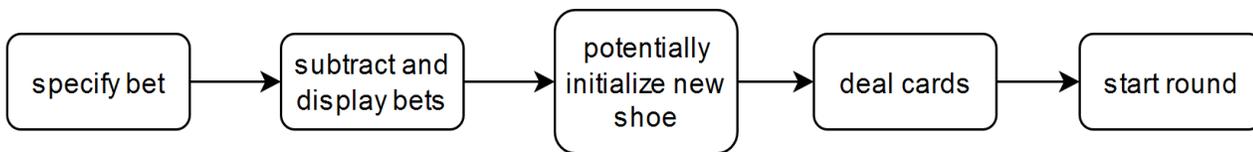


SCN-2: Starting a round

FREQ reference	4,6,10
NFREQ reference	1,2,3,4,5
Short Description:	Peter, Jane and John have just finished a round. All of them have enough money left to pay the minimum bet. They each enter their bets for the next round. The next round starts, it’s Peter’s turn.
Activation action:	A round has just ended and all players still have enough money to pay the minimum bet.
Precondition:	John, Jane and Peter are playing Blackjack and a round has just ended. Peter is the starting player in this round.

Basic flow: Starting a round		
Step	User action	System response
1	All players specify their desired bets for the next round	The bets are displayed, the bet of each player is subtracted from their money.
2		Checks if a new shoe needs to be initialized. Initializes new shoe.
3		Each player is dealt two cards face up, the dealer is dealt one card face up and one card face down. The dealer's face up card is a 7 of spades. It's displayed that Peter is to play.
Post-condition:		A new round has started. Two cards are shown face up for each player, for the dealer one card is shown face up and one face down. It's Peter's turn.

Scenario Diagram for SCN-2 Starting a round

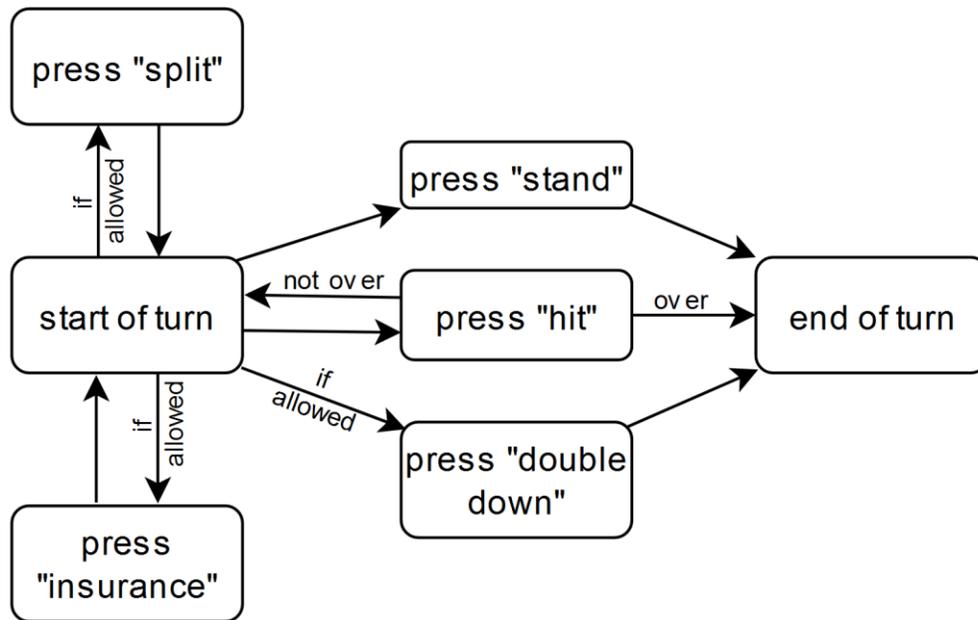


SCN-3: Playing a turn

FREQ reference	4,6,7,8,9,10,14
NFREQ reference	1,2,3,4,5
Short Description:	Jane, John and Peter are playing a round of Blackjack. Peter started the round and has already finished his turn by pressing "stand". It's Jane's turn, she sits to Peter's left. Her two face up cards are a 6 of spades and a 2 of hearts. She presses 'hit' and receives an ace of hearts. She presses the "Stand" button. Her turn ends, it's John's turn.
Activation action:	Peter presses "stand".
Precondition:	Jane, John and Peter are playing a round of Blackjack. Peter's turn has just ended, it's Jane's turn.

Basic flow: Playing a turn		
Step	User action	System response
1	Peter presses "stand"	Peter's score is calculated and displayed. Checks whose turn it is and displays that it's Jane's turn.
2	Jane presses "hit"	The next card is drawn from the shoe and dealt to Jane. The card is displayed face up. Jane's intermediate score is calculated, it's less than or equal to 21, so her turn continues.
3	Jane presses "stand"	Jane's final score is calculated and displayed. Checks whose turn it is next and displays that it's John's turn.
Post-condition:		It's John's turn. Jane's score is displayed.

Scenario Diagram for SCN-3 Playing a turn



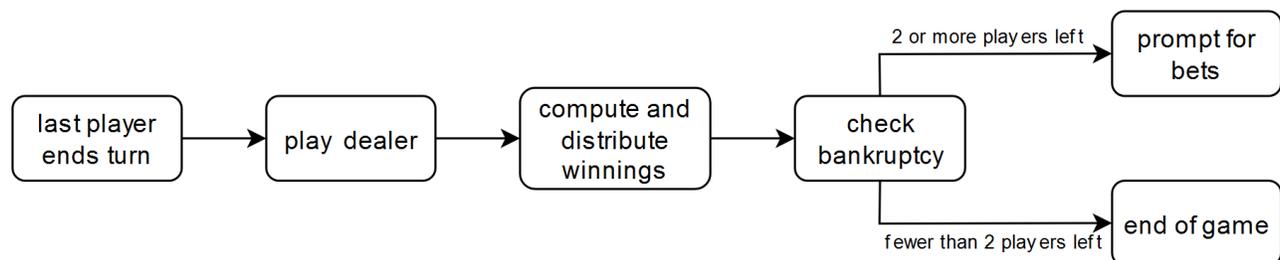
SCN-4: Ending a round

FREQ reference	4,7,9,10,11,12,14
NFREQ reference	1,2,3,4,5
Short Description:	Peter, Jane and John are playing a round of Blackjack. All players have a bet of 50. Peter and Jane have already finished their turns and achieved scores of 19 and 20, respectively. It's John's turn. John has a jack of hearts and a king of spades. He presses "hit" and receives a 6 of clubs. John is over, and his turn ends. The dealer's

	face down card is turned around. The dealer’s score is 16, so she is dealt another card. It’s a 2 of spades. The dealer has a score of 18 and the round ends. 100 is added to Peter’s and Jane’s money. All players still have money. The players are prompted for their next bets.
Activation action:	John presses “hit”.
Precondition:	John is the last to play in the round and starts with a score of 20. All players have enough money left to pay the minimum bet.

Basic flow: Ending a round		
Step	User action	System response
1	John presses “hit”	The next card off the shoe is dealt face up to John, it’s a 6 of clubs. The intermediate score of John is calculated and compared to 21. It’s greater, a notification is shown. John’s score is displayed and his turn ends. All players have finished their turns.
2		The dealer’s face down card is displayed. The dealer’s score is calculated. It’s 16. The dealer is dealt another card. It’s a 2 of spades. The dealer’s score is calculated, it’s 18. The dealer is finished, her score is displayed.
3		All players’ scores are ranked. 100 is added to Jane and Peter’s money. A notification of who won how much is shown.
4		The system checks if a player has gone bankrupt. The round ends and players are prompted to specify their next bets.
Post-condition:		100 has been added to Jane’s and Peter’s money. The bet entry prompt is shown.

Scenario Diagram for SCN-4 Ending a round

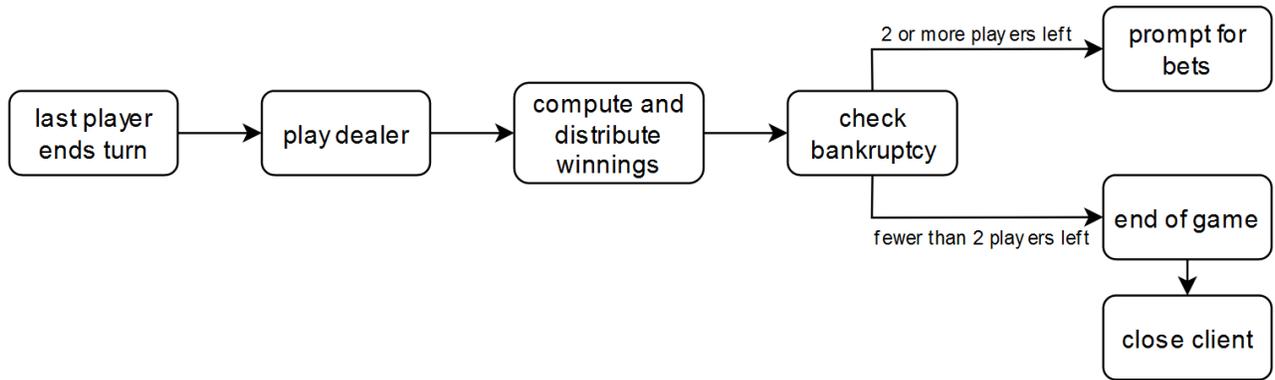


SCN-5: End of the game

FREQ reference	4,7,9,10,11,12,13,14,15
NFREQ reference	1,2,3,4,5
Short Description:	John and Jane are playing a game of Blackjack. Both have a bet of 5, Jane's remaining money is at 200 while John's is at 0. In the current round, John has already finished playing and achieved 15 points. Jane was dealt 20 points and decides to stand. The dealer achieves 19 points. Jane's money is increased by 10. John lost and Jane is the only player to have money left. The game ends. Jane wins the game. John closes the game client.
Activation action:	Jane presses "stand"
Precondition:	John has no more money left.

Basic flow: End of the game		
Step	User action	System response
1	Jane presses "stand"	Jane's score is calculated and displayed. It's 20. The system determines that all players have played.
2		The dealer's face down card is turned and her score is calculated. It's 19. The system determines that the dealer's play is finished. Her score is displayed.
3		The player's scores are ranked. John loses and Jane wins, so 10 is added to Jane's money.
4		The system determines that John is bankrupt. It displays a notification that John is bankrupt. The system determines that Jane has won the game. It displays the end screen.
	John closes the game client	John's client closes, all other instances of the system keep running.
Post-condition:	Jane sees the end screen	

Scenario Diagram for SCN-5 End of game



System Constraints

Important Nonfunctional Requirements

NFREQ-1: Response time

Every action of each player is processed and executed in less than one second.

User priority (4/5): *speed is essential for fun gameplay.*

Technical priority (1/5): *speed does not influence the functioning of the system.*

NFREQ-2: Game Cards and Layout

A classic poker deck can be displayed. The displayed images should be similar to a typical casino setting, but there can be some deviations. For example, if the players' cards are displayed around a semicircle in the casino, in the GUI the cards could be displayed in a linear fashion.

User priority (3/5): *visual inputs/outputs make the game more enjoyable.*

Technical priority (1/5): *visuals do not influence functioning of the system.*

NFREQ-3: Language

The system displays all texts in English.

User priority (4/5): *the product is intended for an international audience, so comprehensibility is important.*

Technical priority (2/5): *texts are designed for the end user but could be useful for identifying and solving problems in case the system malfunctions.*

NFREQ-4: Reliability

The system does not crash more than once every 50 games.

User priority (4/5): *reliability is important, as users would be unsatisfied with the product if it crashed often.*

Technical priority (5/5): *if the game crashes, all the other components of the software cannot work as intended.*

NFREQ-5: Ease of use

A player who is not familiarized with the game can typically find the button to execute the next action within five seconds. This action doesn't have to be a tactically good one; the aforementioned five seconds only refer to the time needed to find the button to proceed in the game.

User priority (4/5): *ease of use promotes fun gameplay.*

Technical priority (1/5): *ease of use does not influence other components of the system.*

Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>