CPSC 3720 Project

Requirements Specification Template

Blackjack

Brent Hawkings

Ben Hunt

Huu Hoang

Table of Contents

Atomic Requirements Shell

1. Card
    a. **Requirement ID: 1**
    b. **Requirement Type:** Functional
    c. **Use Case ID:** 1,2,3,4,8,12,14
    d. **Description:** A playing card
    e. **Rationale:** We need a card to make a card
    f. **Fit Criterion:** Does the card exist in the desk
    g. **Priority:** Essential
2. Ability to hold cards for any actor, dealer, or player (i.e., A hand of cards)
    a. **Requirement ID:** 2
    b. **Requirement Type:** Functional
    c. **Use Case ID:** 4
    d. **Description:** Dealer and player can look at their cards and decide what to do next
    e. **Rationale:** The dealer and the player need to look at their cards to progress in the game, whether to hit or stand
    f. **Fit Criterion:** The player and dealer must have access to separate hands of cards
    g. **Priority:** Essential
3. Make a deck of cards
    a. **Requirement ID:** 3
    b. **Requirement Type:** Functional
    c. **Use Case ID:** 14
    d. **Description:** A deck of regular playing cards is necessary to play blackjack (i.e., 52 cards, 4 suits, 13 cards each suit, Ace through King)
    e. **Rationale:** You cannot play the game without a deck of cards
    f. **Fit Criterion:** The deck exists for the entire length of the game
    g. **Priority:** Essential
4. Shuffle a deck:
    a. **Requirement ID**: 4
    b. **Requirement Type**: Functional
    c. **Use Case ID**: 12
    d. **Description**: Dealer must deal from a shuffled deck, make a deck and have it shuffled.
    e. **Rationale**: The game does not play well if the cards are already sorted
    f. **Fit Criterion**: The deck must not be in any form of sorted order
    g. **Priority**: Essential
5. Player
    a. **Requirement ID**: 5
    b. **Requirement Type**: Functional
    c. **Use Case ID**: 1, 2 ,3, 4, 5, 7, 8, 10, 11, 13
    d. **Description**: The player will receive two cards, one up and one down. They must decide whether to stand or hit to get as close to 21 as possible but not more
    e. **Rationale**: The possible actions and responsibilities of the players
    f. **Fit Criterion**: The user will control this character to play the game
    g. **Priority**: Essential

6. Dealer
    a. **Requirement ID**: 5
    b. **Requirement Type**: Functional
    c. **Use Case ID**: 12, 14
    d. **Description**: The dealer must shuffle the deck and handle the player's cards. Count chips and handling the game procedures
    e. **Rationale**: The possible actions and responsibilities of the dealers
    f. **Fit Criterion**: The bot or another user will control this character
    g. **Priority**: Essential
7. Betting Mechanism
    a. **Requirement ID**: 7
    b. **Requirement Type**: Functional
    c. **Use Case ID**: 5
    d. **Description**: The player must make the betting chip before the game is started
    e. **Rationale**: The game does not play well if the cards are already sorted
    f. **Fit Criterion**: Bet the money if you win gain the bet money + the win bet. If you lose you lose all your money
    g. **Priority**: Desirable
8. Hit
    a. **Requirement ID: 8**
    b. **Requirement Type:** Functional
    c. **Use Case ID:** 2
    d. **Description:** The act of accepting another card from the dealer and adding to the actor's hand
    e. **Rationale:** To gain cards and add to your score
    f. **Fit Criterion:** actor's hand will increase, and their score increases according to the card's value
    g. **Priority:** Essential
9. Stand
    a. **Requirement ID: 9**
    b. **Requirement Type:** Functional
    c. **Use Case ID:** 3
    d. **Description:** The act of refusing a card from the dealer when you are close to a score of 21
    e. **Rationale:** To hold the card and maintain the score
    f. **Fit Criterion:** The actor's hand and score will not increase
    g. **Priority:** Essential
10. Win
    a. **Requirement ID: 10**
    b. **Requirement Type:** Functional
    c. **Use Case ID:** 6
    d. **Description:** The game is over and either the player or the dealer has the highest score or reaches 21
    e. **Rationale:** Need a definite end to the game and incentive to keep playing the game
    f. **Fit Criterion:** The winner's wallet will increase because they earn their bet back and then the same value again
    g. **Priority:** Essential

11. Lose
    a. **Requirement ID: 11**
    b. **Requirement Type:** Functional
    c. **Use Case ID:** 6
    d. **Description:** The actor has gone “busted” with a score over 21 or has the lowest score
    e. **Rationale:** Necessary so that there is also a winner
    f. **Fit Criterion:** The actor will lose their bet and not gain it back and the score is less than the winner
    g. **Priority:** Essential
12. Tie
    a. **Requirement ID: 12**
    b. **Requirement Type:** Functional
    c. **Use Case ID:** 6
    d. **Description:** Both the dealer and player have the same score or have both reached 21
    e. **Rationale:** If a tie happens, we need to decide how the game ends
    f. **Fit Criterion:** The actor will lose their bet and not gain it back
    g. **Priority:** Essential
13. Wallet Tracker
    a. **Requirement ID: 13**
    b. **Requirement Type:** Functional
    c. **Use Case ID:** 7
    d. **Description:** Amount of chips in the player’s wallet, upon initial startup, the player should be gifted a certain amount of chips to play with
    e. **Rationale:** The player can only bet while they have chips in their wallet
    f. **Fit Criterion:** Wallet will change with each round of the game played (increase or decrease)
    g. **Priority:** Desirable
14. Doubling Down
    a. **Requirement ID: 14**
    b. **Requirement Type:** Functional
    c. **Use Case ID:** 10
    d. **Description:** If the first 2 cards dealt to the player equal 9, 10, 11 then the user can double the original bet and get a new card face down and then evaluate the hand.
    e. **Rationale:** More ways to increase score
    f. **Fit Criterion:** User can choose to double down after the first two cards equal 9,10, or 11
    g. **Priority:** Desirable
15. Splitting Pairs
    a. **Requirement ID: 13**
    b. **Requirement Type:** Functional
    c. **Use Case ID:** 8
    d. **Description:** If a pair exists in the user’s hand, then he/she can split it and create 2 hands placing the original bet on one and a new bet of equal value on the other hand. Not possible for the dealer
    e. **Rationale:** Give the user more ways to increase their score
    f. **Fit Criterion:** User can split pairs and assign bets to each hand
    g. **Priority:** Desirable

16. Code in C++
    a. **Requirement ID:** 16
    b. **Requirement Type:** Solution Constraint
    c. **Description:** Create the game in C++
    d. **Rationale**: Grading depends on the solution being in C++
    e. **Fit Criterion**: Source code in C++
    f. **Priority***:* Essential
17. Runs on U of L Computer Science Lab Computers
    a. **Requirement ID:** 17
    b. **Requirement Type:** Solution Constraint
    c. **Description:** Demonstrations of the finished product can be done in the C or D lab of U of L
    d. **Rationale**: Execution of program happens at the U of L CS lab computers
    e. **Fit Criterion**: Game compiles and runs on CS lab computers
    f. **Priority:** Essential
18. Command-line application
    a. **Requirement ID**: 18
    b. **Requirement Type:** Solution Constraint
    c. **Description:** An application that runs on the command line in a terminal emulator
    d. **Rationale:** Using GUI libraries would take too long and slow down the development process
    e. **Fit Criterion:** There is no GUI component to the application
    f. **Priority:** Essential
19. Must be testable using GTEST
    a. **Requirement ID: 19**
    b. **Requirement Type:** Non-functional
    c. **Description:** Use the GTEST library on the school computer
    d. **Rationale**: All methods should have unit tests to ensure proper function
    e. **Fit Criterion**: Can be run with GTEST and everything passes.
    f. **Priority**: Essential
20. Must have code coverage of 90% or higher
    a. **Requirement ID:** 20
    b. **Requirement Type:** Non-functional
    c. **Description:** Use tools like gcov and lcov to report code coverage
    d. **Rationale**: Dead code is wasted code so do not include it
    e. **Fit Criterion**: Gcov reports code coverage of 90+%.
    f. **Priority**: Essential
21. Must have no memory leaks
    a. **Requirement ID:** 21
    b. **Requirement Type:** Non-functional
    c. **Description:** Use tools like ‘valgrind’
    d. **Rationale**: We respect the user’s property and seek to use the resources appropriately
    e. **Fit Criterion**: The memory checker valgrind reports no leaks
    f. **Priority**: Essential

22. Code Hosted on U of L GitLab instance
    a. **Requirement ID:** 22
    b. **Requirement Type:** Process Constraint
    c. **Description:** Create the game in C++
    d. **Rationale**: Grading depends on the solution being in C++
    e. **Fit Criterion**: Source code in C++
    f. **Priority***:* Essential
23. Code will be styled with Cpplint
    a. **Requirement ID: 23**
    b. **Requirement Type:** Non-functional
    c. **Description:** A software design pattern
    d. **Rationale:** Make a game that is easy to understand and extend
    e. **Fit Criterion:** The code follows the MVC pattern
    f. **Priority:** Essential
24. Documentation generated with Doxygen
    a. **Requirement ID: 24**
    b. **Requirement Type:** Non-functional
    c. **Description:** Create docs from source code comments
    d. **Rationale:** Documentation of the code makes it easier to work on for long-term
    e. **Fit Criterion:** There is a generated HTML interface for your code based on doxy strings
    f. **Priority:** Essential
25. Score Board
    a. **Requirement ID: 25**
    b. **Requirement Type:** Non-functional
    c. **Use Case ID:** 11
    d. **Description:** A place to record all your previous scores and (positive/negative) earnings from previous matches
    e. **Rationale:** Can see wins, losses, and ties
    f. **Fit Criterion:** User can enter a name and add it to a database of previous playthroughs
    g. **Priority:** Optional
26. ***Permanent*** Wallet Tracker
    a. **Requirement ID: 26**
    b. **Requirement Type:** Non-functional
    c. **Use Case ID:** 7
    d. **Description:** Keep track of your wallet across gaming sessions
    e. **Rationale:** If you play for a long time you may want to maintain your earnings across play sessions.
    f. **Fit Criterion:** User's wallet earnings will be the same compared to last play session
    g. **Priority:** Optional
27. The user can exit the game at any time
    a. **Requirement ID:** 27
    b. **Requirement Type:** Functional
    c. **Use Case ID:** 15
    d. **Description:** The user can exit the game at any time
    e. **Rationale:** Need to allow the user to end the game gracefully

f. **Fit Criterion:** The user can type ‘exit’ to quit the game
g. **Priority:** Essential

28. Help/Info screen
    a. **Requirement ID:** 28
    b. **Requirement Type:** Functional
    c. **Use Case ID:** 16
    d. **Description:** User can type help on the command line
    e. **Rationale:** If the user needs help with what they can do in the game they can type ‘help’ to get the help message
    f. **Fit Criterion:** Typing help on the command line gives you a help message
    g. **Priority:** Essential

1. The Purpose of the Project

Goals of the Project

The Game

(Essential Goals)

Dealing:

The dealer thoroughly shuffles the deck leaving the last 60 to 75 cards in reserve. (Not dealing with the bottom of all the cards makes it more difficult to count cards). The dealer then gives one card face up to each player in a clockwise rotation, finishing with giving one card to themselves. Another round of cards is then dealt face up to each player, but the dealer takes the second card face down.

The Play:

The player to the left goes first and must decide whether to "stand" (not ask for another card) or "hit" (ask for another card to get closer to a count of 21, or even hit 21 exactly). Thus, a player may stand on the two cards originally dealt to them, or they may ask the dealer for additional cards, one at a time, until deciding to stand on the total (if it is 21 or under) or goes "bust" (if it is over 21). In the latter case, the player loses, and the dealer collects the bet wagered. The dealer then turns to the next player to their left and serves them in the same manner.

Dealer Logic:

When the dealer has served every player, the dealers face-down card is turned up. If the total is 17 or more, it must stand. If the total is 16 or under, they must take a card. The dealer must continue to take cards until the total is 17 or more, at which point the dealer must stand. If the dealer has an ace and counting it as 11 would bring the total to 17 or more (but not over 21), the dealer must count the ace as 11 and stand. The dealer's decisions, then, are automatic on all plays, whereas the player always has the option of taking one or more cards.

Soft Hands:

The combination of an ace with a card other than a ten-card is known as a "soft hand," because the player can count the ace as a 1 or 11, and either draw cards or not. For example, with a "soft 17" (an ace and a 6), the total is 7 or 17. While a count of 17 is a good hand, the player may wish to draw for a higher total. If the draw creates a bust hand by counting the ace as an 11, the player simply counts the ace as a 1 and continues playing by standing or "hitting" (asking the dealer for additional cards, one at a time).

Naturals:

If a player's first two cards are an ace and a "ten-card" (a picture card or 10), giving a count of 21 in two cards, this is a natural or "blackjack". If any player has a natural and the dealer does not, the dealer immediately pays that player one and a half times the amount of their bet. If the dealer has a natural, they immediately collect the wager of all players who do not have naturals, (but no additional amount). If the dealer and another player both have naturals, the bet of that player is a tie), and the player takes back his chips. If the dealer's face-up card is a ten-card or an ace, they look at their face-down card to

see if the two cards make a natural. If the face-up card is not a ten-card or an ace, they do not look at the face-down card until it is the dealer's turn to play.

Help/Info:

Shows a list of all commands and some general information about how to play blackjack.

(Desirable Goals)

Betting:

Betting is done before the dealer deals and is recorded in a denomination of chips. The player must place at least the minimum bet each hand. If the player loses, they forfeit their bet. If the player wins, they are returned their bet plus an amount equal to their bet. If the hand results in a tie, no chips are exchanged.

Bet Settlement:

A bet once paid and collected is never returned. Thus, one key advantage to the dealer is that the player goes first. If the player goes bust, they have already lost their wager, even if the dealer goes bust as well. If the dealer goes over 21, the dealer pays each player who has stood the amount of that player's bet. If the dealer stands at 21 or less, the dealer pays the bet of any player having a higher total (not exceeding 21) and collects the bet of any player having a lower total. If there is a tie (a player having the same total as the dealer), no chips are paid out or collected.

Wallet:

A pool of chips from which the player can hold his or her winnings and make bets from. Starts at a standard amount of chips to be decided later. Could be utilized in a potential save state system.

Splitting Pairs:

If a player's first two cards are of the same denomination, such as two jacks or two sixes, they may choose to treat them as two separate hands when their turn comes around. The amount of the original bet then goes on one of the cards, and an equal amount must be placed as a bet on the other card. The player first plays the hand to their left by standing or hitting one or more times; only then is the hand to the right played. The two hands are thus treated separately, and the dealer settles with each on its own merits. With a pair of aces, the player is given one card for each ace and may not draw again. Also, if a ten-card is dealt to one of these aces, the payoff is equal to the bet (not one and one-half to one, as with a blackjack at any other time).

Doubling Down:

Another option open to the player is doubling their bet when the original two cards dealt total 9, 10, or 11. When the player's turn comes, they place a bet equal to the original bet, and the dealer gives the player just one card, which is placed face down and is not turned up until the bets are settled at the end of the hand. With two fives, the player may split a pair, double down, or just play the hand in the regular way. Note that the dealer does not have the option of splitting or doubling down.

(Optional goals)

Save-States:

A means to save the players wallet balance, and record (scoreboard) for another play session.

Scoreboard:

A means to record Wins, Losses, Ties, and winnings (either positive or negative).

The Scope

The scope of this game will be limited to the dealer and one player. The scope will include splitting pairs, and doubling down, but at a lower priority. Making side bets is not included in the scope as this is a single player game.

Measuring Success

To measure success, we will identify whether the goals of the game are implemented, and if they are implemented well. If the essential goals are not met, or not met to a reasonable quality, we would not measure that as successful. If the desirable goals are met, and to a reasonable quality, that would be considered a success. If all those goals are met and the optional goals are implemented, that would be evaluated as very successful.

Priority	**Goals of the Project**	**Implemented?**	**Quality**
Essential	Dealing		_/5
	The Play		_/5
	Dealer Logic		_/5
	Soft Hands		_/5
	Naturals		_/5
	Help/Info		_/5
Desirable	Betting		_/5
	Bet Settlement		_/5
	Wallet		_/5
	Splitting Pairs		_/5
	Doubling Down		_/5
Optional	Save-States		_/5
	Scoreboard		_/5

2. The Stakeholders

We will use three examples of stakeholders that may be interested/invested in the projects:

Working class:

Jason Mason Alexander is a 31-year-old supervising technician at a factory. Different from the other blue collars, he is an extrovert and enjoys cycling, car racing and spending times with friends. He is energetic, open, and generous, but can also be very hot blooded.

He is an American. He started studying engineering at college but never finished the course.

Physically, Jason is in pretty good shape. He is average height with olive skin, mousey hair, and blue eyes.

He grew up in a working-class neighbourhood. He was raised in an average household, but he has fascination and affinity for many things. He is unusually good at card games which can surprise many people.

He is currently single. His most recent romance was with a lawyer called Robin Lucinda Walters, who is around his age. She is also his best friend. They have been friends since high school until Robin left for higher education, but they stayed in contact. Years later, he proposed to her, but their relationship did not progress so the two broke up and decided to stay as friends.

He also plays with Penelope Hutchinson and Yasmin Bowman. They enjoy playing card games together.

Upper class:

Doris Molly Torrance is a 28-year-old management consultant who enjoys reading, listening to music, and binge-watching boxed sets. She is entertaining and inspiring but can also be very sneaky and a bit greedy.

She is an American Christian who defines herself as straight. She has a degree in business studies.

Physically, Doris is in good shape. She is average height with bronze skin, mouse hair and blue eyes.

She grew up in an upper-class neighbourhood. She was raised by her father; her mother having left when she was young.

She is currently single. Her most recent romance was with a chef called Jeff Kade Simmons, who was 8 years older than her. They broke up because Jeff wanted a quieter life than Doris could provide.

Doris has one child with ex-boyfriend Jeff: Gertrude aged 1.

Doris's best friend is a management consultant called Nell Young. They get on well most of the time. She also hangs around with Hope Rogers and Trinity Taylor. They enjoy ferret racing together.

Middle class:

Hannah Ruth Godfrey is a 35-year-old Croupier who enjoys working on cars, playing card games, and performing precision and challenging work to improve her skills on dealing with cards. She is smart and energetic but can also be very sneaky and sly. Such are the required personalities for the jobs she often claimed.

She is an American Christian who defines herself as straight. She has a post-graduate degree in law. She has a severe phobia of crocodiles.

Physically, Hannah is in surprisingly good shape. She is tall with light skin, copper hair and brown eyes.

She grew up in a middle-class neighbourhood. After her mother died when she was young, she was raised by her father and became independent early.

She is currently single. She used to be in a relationship with an amateur novelist called Montgomery Wesley Baker, but they broke up shortly because both cannot feel this relationship moving forward.

Hannah's best friend is a partner at a law firm called Nadia Thomas who help her with various problems related to court case. They get on well most of the time.

User Priority

1. Hannah Ruth Godfrey - Key user: She is critical to the continued success of the product. As a dealer, she is knowledgeable of the rule and gives advice on the project. Thus, greater importance to requirements should be given to this user.

2. Jason Mason Alexander - Secondary users: He will use the product and can be invested in the process, but his opinion has no effect on the project's long-term success. The opinion of this users' requirements will be noted but the key user's idea takes precedence.

3. Doris Molly Torrance - Unimportant users: This category of user is given the lowest priority. She is not a person who will be involved in the program, and she is not guaranteed to be a skilled user. Making mistakes in usage will not be uncommon.

Development Team (or Other Stakeholders)

Possible roles for the development team (or other stakeholders) are required for the development of the program.

1. Designers
2. Developers
3. Testers
4. Maintenance operator
5. Clients
6. Product managers

3. Constraints

Solution Constraints

Code in C++

Rationale: Grading depends on solution being in C++.

Fit Criterion: Source code in C++.

Runs on U of L Computer Science Lab Computers

Rationale: Execution of program happens at the U of L CS lab computers.

Fit Criterion: Game compiles and runs on CS lab computers.

Command-line application

Rationale: Using GUI libraries would take too long and slow down development process.

Fit Criterion: There is no GUI component to the application.

Code Hosted on U of L GitLab instance

Rationale: Grading depends on solution being in C++.

Fit Criterion: Source code in C++.

Process Constraints

Must be accomplished within 2 weeks

Rationale: Grading requires it to be finished in two weeks.

Fit Criterion: The application is finished within 2 weeks.

Each person on team must work minimum 4-5 per week on project

Rationale: 4-5 hours for each member of the team should enable meeting the deadline.

Fit Criterion: The workload is balanced across the team members.

Code Hosted on U of L GitLab instance

Rationale: Grading depends on solution being in C++.

Fit Criterion: Source code in C++.

4. Naming Conventions and Terminology

Glossary

Ace: An ace card. The player can count the ace as either worth 1 or 11. If the dealer has an ace and counting it as 11 would bring the total to 17 or more (but not over 21), the dealer must count the ace as 11 and stand.

Bet: The wager placed on a hand (in chips) by the player. If the player loses, they forfeit their bet. If the player ties, they are returned exactly what they bet. If the player wins, they are returned their bet and winnings which are equal to their bet.

Blackjack:

1. The game.

2. Either the player or the dealer having a total hand of 21.

Bust: Happens when the total of either the player or dealer exceeds 21. If the dealer busts but the player does not the player wins. If the player busts but the dealer does not the player loses. If both the player and the dealer bust the outcome is not a tie, the player loses and forfeits their bet.

Chip: The single standard unit of a bet. We will shy away from using terms like “dollars,” “cash” or “money.”

Dealer: The entity known as the dealer is an automatic process that runs every hand. In contrast to the player who always has the option to hit or stand, the dealer's decisions, are automatic on all plays.

Doubling Down: An option available to the player when the original two cards dealt a total 9, 10, or 11. When the player's turn comes, to double down, they place a bet equal to the original bet, and the dealer gives the player just one card, which is placed face down and is not turned up until the bets are settled at the end of the hand. With two fives, the player may either double down or split a pair, but not both. The dealer does not have the option of doubling down.

Hand:

1. The data structure used to contain the player and dealer's cards. For the player both are displayed to see so that the player can make informed decisions.

2. One round of play.

Help: A command perpetually available to the player to display useful information such as game controls, and basic game information.

Hit: The act of a player taking another card from the dealer.

Lose/Loss: An outcome of a hand where the dealer beats the player by either having a blackjack when the player does not, or simply by outcounting the player.

Natural: The player or dealer's first two cards which by the combination of an ace worth 11 and a card worth 10, make a blackjack. If the player has a natural and the dealer does not, the dealer immediately pays that player one and a half times the amount of their bet. If the dealer has a natural and the player does not, they immediately collect the bet of the player. If the dealer and the player both have naturals, the outcome of that hand is a tie. If the dealer's face-up card is a ten-card or an ace, they reveal their face-down card to see if the two cards make a natural.

Player: The player. Has the ability to hit or stand, double down, split pairs, make bets, and see the cards on the table.

Scoreboard: A displayed arrangement of statistics such as Wins, Losses, Ties, and winnings (either positive or negative).

Stand: The act of a player choosing to not take another card so as not to bust, and potentially lose the hand. The dealer will stand on a total of 17 or higher.

Soft Hand: The combination of an ace with a card other than a ten-card is known as a "soft hand," because the player can count the ace as a 1 or 11, and either draw cards or not. For example, with a "soft 17" (an ace and a 6), the total is 7 or 17. While a count of 17 is a good hand, the player may wish to draw for a higher total. If the draw creates a bust hand by counting the ace as an 11, the player simply counts the ace as a 1 and continues playing by standing or "hitting" (asking the dealer for additional cards, one at a time).

Splitting Pairs: If a player's first two cards are a pair, such as two jacks or two sixes, they may choose to treat them as two separate hands when their turn comes around. The amount of the original bet then goes on one of the cards, and an equal amount must be placed as a bet on the other card. The player first plays the hand to their left by standing or hitting one or more times; only then is the hand to the right played. The two hands are thus treated separately, and the dealer settles with each on its own merits. With a pair of aces, the player is given one card for each ace and may not draw again. Also, if a ten-card is dealt to one of these aces, the payoff is equal to the bet (not one and one-half to one, as with a blackjack at any other time).

Shuffle: The ability to randomize the deck.

Ten/Ten Card: Any face card or 10 card is worth 10 points. Can combine with a singular ace worth 11 to make blackjack.

Tie: An outcome of a hand where both the player and the dealer stand on the same total. Sometimes referred to as a "push," but we will refrain from labelling it that in the interest of simplicity.

Wallet: The account or pool which holds the players chips from which they can make bets. Starts at a predetermined amount and will fluctuate based on the outcome of the players performance.

Win: An outcome of a hand where the player beats the dealer by either having a higher total at the end of the hand (without busting) or by getting a natural when the dealer does not.

5. Assumptions

Assumptions List

What the developers expect to be defined in time for them to use:

1. Atomic requirements.
2. Project goals.
3. Stakeholder identification.
4. Solution and process constraints.
5. Naming conventions.
6. Assumptions.
7. Product scope and use case(s).
8. Functional requirements.
9. Non-Functional requirements.
10. Look and Feel requirements.
11. Risks and challenges.

The technological environment in which the product will operate:

1. U of L Computer lab computers.
2. Unix based systems.
3. Will take user interaction by means of the command line.

The software components that will be available to the developers:

1. Source Code will be written in C++.
2. Git will be used for version control. Gitlab will host the repository.
3. The program will be run in the command line.
4. Code coverage will be checked by gcov/lcov.
5. Memory management will be done with valgrind.
6. Style checking will be done with cpplint.

7. Testing will be done with gtest.
8. Documentation will be done with doxygen.

Dependencies on computer systems or people external to this project:

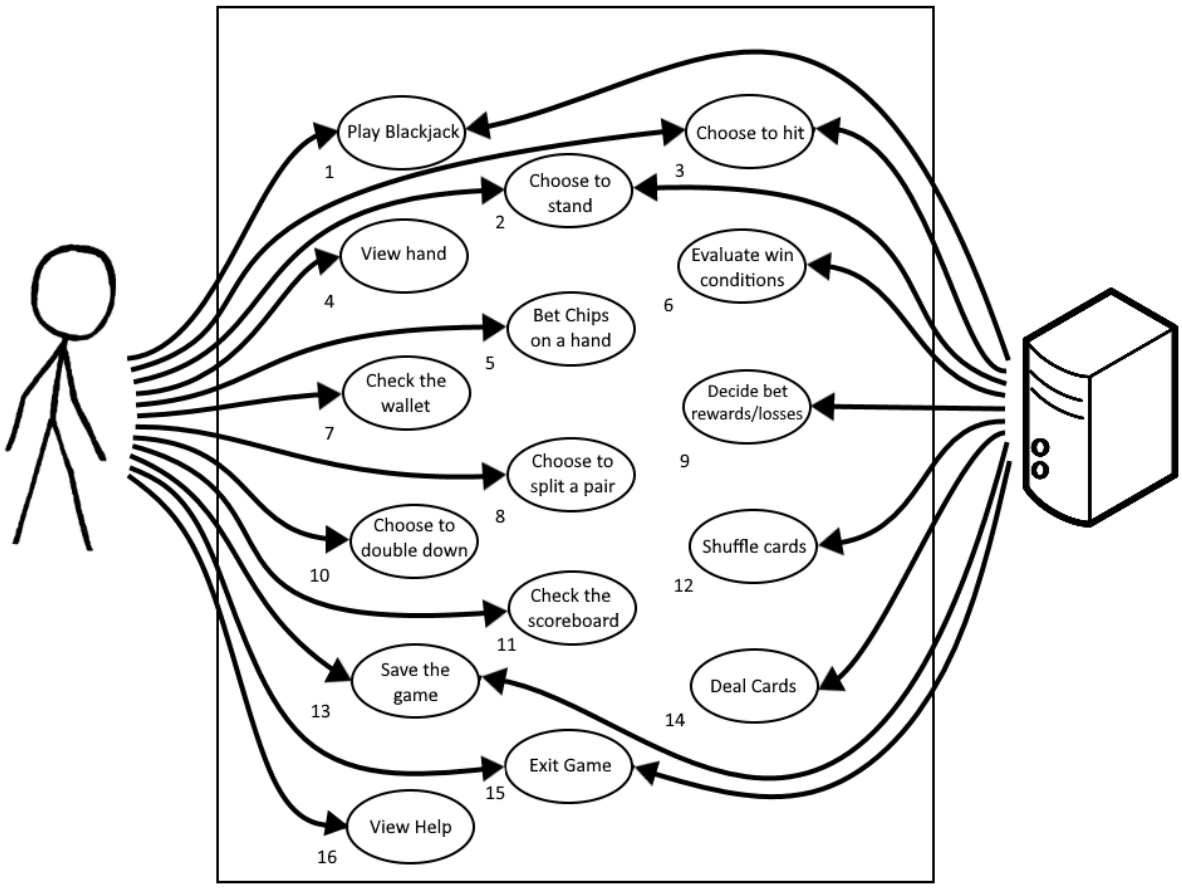
1. The product will not be dependent upon any external computer systems or people.

Requirements that will specifically not be carried out by the product (Everything listed below is still required, just will specifically not be carried out by the end project, as they are non-functional requirements):

1. Must have code coverage of 90% or higher.
2. Must have no memory leaks.
3. Code Hosted on U of L GitLab instance.
4. Uses Model/View/Controller (MVC) Architectural Pattern.
5. Code will be styled with Cpplint.

6. The Scope of the Product

Use Case Diagram



Use Cases

1. **Play blackjack**

Scenario: The player tells the computer to start the program.

Rationale: The player wants to play blackjack.

2. **Choose to stand**

Scenario: The player has a 3, a 7 and receives a king, and is therefore counting 20.

Rationale: The player chooses to stand.

3. **Choose to hit**

Scenario: The player has a 3 and a 7 and is therefore counting 10.

Rationale: The player chooses to hit.

4. **View hand**

Scenario: A new hand has started being played.

Rationale: The player must be able to view what cards they and the dealer have in order to make good decisions.

5. **Bet chips on a hand**

Scenario: The player bets 5 chips.

Rationale: The player must make a minimum bet on each hand.

6. **Evaluate win conditions**

Scenario: The dealer is standing on 19, and the player has stood on 20, therefore the player wins.

Rationale: The dealer must conclude that the player has won the hand.

7. **Check the wallet**

Scenario: The player has lost quite a few hands and wants to see how many chips are left in their wallet.

Rationale: The player must know the balance of their wallet.

8. **Choose to split a pair**

Scenario: The player receives a pair of 5's from the dealer, they choose to split, and play two hands.

Rationale: The rules of blackjack allow for the splitting of pairs.

9. **Decide bet rewards/losses**

Scenario: The player has won the hand and the dealer must pay out winnings equal to their bet.

Rationale: The dealer must conclude all hands by using the outcome of the hand to administer winnings/losses accordingly.

10. **Choose to double down**

Scenario: The player receives a 6 and a 5 and is therefore counting 11.

Rationale: The player chooses to double down.

11. **Check the scoreboard**

Scenario: The player has remembered how many wins they have but they do not remember how many times they have lost.

Rationale: The player wants to check the scoreboard.

12. **Shuffle cards**

Scenario: A new hand is starting.

Rationale: The dealer must shuffle the deck.

13. **Save the game**

Scenario: The player wants to exit the game, and they want to return later where they left off in terms of wins/losses.

Rationale: The program must save their wallet balance and all-time record.

14. **Deal Cards**

Scenario: A new hand has started, and the game commences.

Rationale: The dealer must deal the cards.

15. **Exit Game**

Scenario: The player wants to stop playing the game.

Rationale: The program must terminate.

16. **View Help**

Scenario: The player wants to view the help screen because they forgot a command.

Rationale: The program must be able to communicate to the user it is controls.

7. Functional Requirements

Points – Hours

1 0-1

2 1-2

3 3-4

4 5-6

5 7-8

The wallet tracker tracks all money spent by the players

Priority or Implementation Order	Essential
Use Cases	7, 13
Rationale:	The player can only participate in the game while chips are still available in their wallet
Fit Criterion	The amount of chips in the player's wallet will change with each game round (increase or decrease)
Story Points	3

The dealers shuffle the card

Priority or Implementation Order	Essential
Use Cases	12
Rationale:	The dealer must shuffle the deck before every new game and the card must exist in the deck for the dealer to shuffle
Fit Criterion	Every time the game start, the dealer must shuffle the deck and the deck must not be in any form of sorted order
Story Points	3

The dealer makes a deck of cards

Priority or Implementation Order	Essential
Rationale:	You cannot play the game without a deck of cards
Fit Criterion	The deck exists for the entire length of the game
Story Points	1

The players use the betting mechanism

Priority or Implementation Order	Desirable
Use Cases	5,9
Rationale:	You cannot play the game without using the betting mechanism
Fit Criterion	User must make any amount of betting to determine the amount they will get and lose before starting the game
Story Points	3

The player wins the game against the dealer

Priority or Implementation Order	Desirable
Use Cases	6
Rationale:	One of many definite ends to the game and incentive to keep playing the game
Fit Criterion	The win condition is available when the card value of player is closer to 21 than the dealer
Story Points	1

The player loses the game against the dealer

Priority or Implementation Order	Desirable
Use Cases	6
Rationale:	One of many definite ends to the game and incentive to keep playing the game
Fit Criterion	The lose condition is met when the card value of player is farer from 21 than the dealer or the value of the card is greater than 21
Story Points	1

The player and the dealer tie at the game

Priority or Implementation Order	Desirable
Use Cases	6
Rationale:	One of many definite ends to the game and incentive to keep playing the game
Fit Criterion	The tie condition is met when the card value of player is equal that of the dealer
Story Points	3

The player chooses to hit

Priority or Implementation Order	Desirable
Use Cases	3
Rationale:	One of many choices a player can make to become victorious. To gain cards and add to your score
Fit Criterion	Actor's hand will increase, and their score increase according to the card's value
Story Points	1

The player chooses to stand

Priority or Implementation Order	Desirable
Use Cases	2
Rationale:	One of many choices a player can make to become victorious. To stay the card and maintain the score
Fit Criterion	Actor's hand and score will not increase
Story Points	1

The player uses hold card ability

Priority or Implementation Order	Essential
Use Cases	4
Rationale:	The player needs to look at their cards they own to progress in the game, whether to hit or stand
Fit Criterion	The player must have access to separate hands of cards
Story Points	1

The dealer uses hold card ability

Priority or Implementation Order	Essential
Rationale:	The dealer needs to look at their cards they own to progress in the game, whether to response to player decision to hit or stand
Fit Criterion	The dealer must have access to separate hands of cards
Story Points	3

The ability to make a card

Priority or Implementation Order	Essential
Rationale:	The actor must have a card to use it
Fit Criterion	The actor must have a card list so they can add more card on their hand
Story Points	1

The player decides to double down

Priority or Implementation Order	Desirable
Use Cases	10
Rationale:	More ways to increase score to come close to the value of 21
Fit Criterion	User can choose to double down after the first two cards equal 9,10, or 11
Story Points	2

The player decides to split pair

Priority or Implementation Order	Desirable
Use Cases	8
Rationale:	One of many ways to give the user ability to increase their score to get close to the value of 21
Fit Criterion	User can split pairs and assign bets to each hand
Story Points	2

The player can exit the game at any time

Priority or Implementation Order	Essential
Use Cases	15
Rationale:	Need to allow the user to end the game gracefully
Fit Criterion	The user can type ‘exit’ to end the game
Story Points	1

The player can view help screen at any time

Priority or Implementation Order	Optional
Use Cases	16
Rationale:	Allow the user to get all the help they need during a game
Fit Criterion	The user can type ‘help’ to open the game screen.
Story Points	1

8. Non-functional Requirements

The program must be testable using GTEST

Priority or Implementation Order	Essential
Rationale:	All methods should have unit tests to ensure proper function using the GTEST library on school computer
Fit Criterion	Can be run with GTEST and everything passes
Story Points	1

The program must have code coverage of 90% of higher

Priority or Implementation Order	Essential
Rationale:	Dead code is wasted code so do not include it.
Fit Criterion	Gcov reports code coverage of 90+%.
Story Points	1

The program must have no memory leaks

Priority or Implementation Order	Essential
Rationale:	we respect the user's property and seek to use the resources appropriately.
Fit Criterion	the memory checker valgrind reports no leaks.
Story Points	1

The program code must be styled with Cpplint

Priority or Implementation Order	Essential
Rationale:	make a game that is easy to understand and extend.
Fit Criterion	Gcov reports code coverage of 90+%.
Story Points	1

The program documentation generated with Doxygen

Priority or Implementation Order	Essential
Rationale:	Documentation of the code makes it easier to work on for long-term
Fit Criterion	There is a generated html interface for your code based on doxy strings
Story Points	1

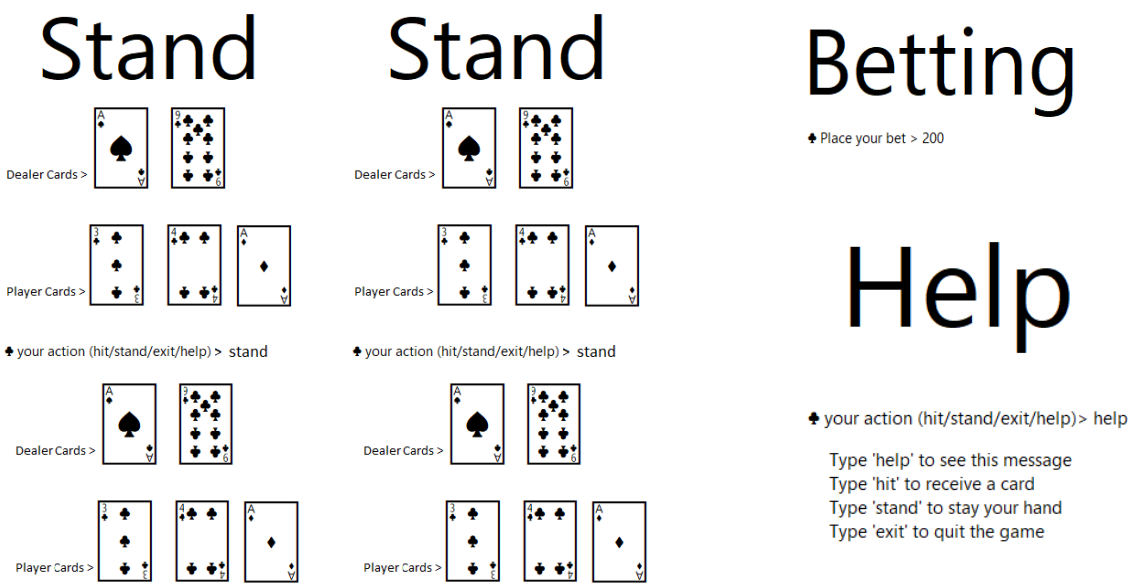
The program scoreboard

Priority or Implementation Order	Optional
Rationale:	Can see wins, losses, and ties. Can check on the dealer or player's performance
Fit Criterion	User can enter a name and add it to a database of previous playthroughs
Story Points	1

The permanent wallet tracker

Priority or Implementation Order	Optional
Rationale:	If you play for a long time you may want to maintain your earnings across play sessions.
Fit Criterion	User's wallet earnings will be the same compared to the last play session
Story Points	1

9. Look and Feel Requirements



If the school computers can display the characters correctly, it would greatly improve the user experience if we could display the cards. The card Unicode characters can be found at here.

The UI should display both dealer and player cards and then ask for player action hit, stand, split pair, double down, etc.

At any point the player should have the ability to submit commands such as help, and exit, scoreboard, wallet, etc.

The help information should be laid out logically and inform the player of what commands are available and give some background to the game of blackjack.

At the end of a hand when the win outcome and bet settlement is needed, the appropriate information should be displayed. Such as: the hand totals for the dealer and player, who won, who lost, and how many betting chips were exchanged.

10. Risks

Probability = High, medium, low

1. Requirements/Design/Estimates
    a. Project cannot be implemented in the allotted time.
        i. **Probability of occurrence**: Medium
        ii. **Contingency**: Re-evaluate the essential requirements and determine if they are essential. Check with authors and implement only those.
    b. Redesign major components of the game during implementation.
        i. **Probability of occurrence:** Medium-Low
        ii. **Contingency:** Branch repository to save original work and start a new one ready for the redesign
    c. Requirements are ambiguous
        i. **Probability of occurrence:** Medium-low
        ii. **Contingency:** Check with authors to clarify and if no clarification made then make assumption that fits best with current timeline and resource levels.
2. People
    a. Member contracts COVID-19:
        i. **Probability of occurrence**: High-Medium
        ii. **Contingency:** If possible, member works from home using the VPN (Virtual Private Network) to access GitLab.
    b. Member drops the course
        i. **Probability of occurrence:** low
        ii. **Contingency:** Meet team redistribute workload and re-evaluate essential requirements to see which need to be completed next.
    c. Member does not have a C++ background
        i. **Probability of occurrence:** low
        ii. **Contingency:** Give member low story point items to get him/her familiar with C++.
    d. Member is hard to contact
        i. **Probability of occurrence:** Medium

ii. **Contingency:** Contact Dr. Anvik about the situation.

e. Members GitLab account suffer an error.

i. **Probability of occurrence:** Low

ii. **Contingency:** Use “Technical Help” channel in MS Teams for help and then if all else fails contact Sys. Admin.

f. Member does not perform as needed for the project.

i. **Probability of occurrence:** High-Medium

ii. **Contingency:** Encourage person to contribute, if all else fails contact Dr. Anvik about the issue.

g. New member added to the team

i. **Probability of occurrence:** Low

ii. **Contingency:** Initially give him/her any of the easiest work items and then redistribute workload once time allows.

3. Learning/Tools

a. Tough time understanding MVC design

i. **Probability of occurrence:** Medium

ii. **Contingency:** Find other online learning resources and go to Dr. Anvik’s office hours.

b. Git is not understood

i. **Probability of occurrence:** Low

ii. **Contingency:** Get a git cheat sheet from the internet and consult the manual and further learning materials.

c. School systems or online servers shuts down and team members cannot access GitLab

i. **Probability of occurrence:** Low

ii. **Contingency:** Consult the technical help of MS Teams. Check with Dr. Anvik about problem.