Software Requirements Specification for SFWRENG 4G06: Dice Duels: Duel of the Eights

Team 9, dice_devs
John Popovici
Nigel Moses
Naishan Guo
Hemraj Bhatt
Isaac Giles

October 10, 2024

Contents

Revision History

Table 1: Revision History

Date	Developer(s)	Change
2024-09-27	John Popovici	Set up formatting
2024-09-30	Nigel Moses	Added three blank sections (Commonalities related sections)
2024-10-04	Nigel Moses	Added content to section 11, 12, 13
2024-10-05	Hemraj Bhatt	Added content to section 2 and 3
2024-10-05	John Popovici	Added content to section 4.1 and 4.1.1
2024-10-05	John Popovici	Fixed compiling issues and formatting in section 3
2024-10-07	John Popovici	Added goals and stretch goals to section 4.1.3
2024-10-07	Isaac Giles	Added functional and non-functional requirements to section 5
2024-10-08	John Popovici	Removed sections based on physical description
2024-10-08	John Popovici	Added stakeholders section template
2024-10-09	Hemraj Bhatt	Added content to section 9
2024-10-09	Naishan Guo	Added Likely and Unlikely changes to section 6
2024-10-09	John Popovici	Added content to stakeholder section
2024-10-09	John Popovici	Added content to reflection and removed comments
2024-10-10	Isaac Giles	Updated non-functional requirements and added to reflection
2024-10-10	Hemraj Bhatt	Updated content on section 2 and 3
...	...	...

1 Reference Material

This section records information for easy reference.

1.1 Table of Units

Throughout this document SI (Système International d'Unités) is employed as the unit system. In addition to the basic units, several derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

symbol	unit	SI
m	length	metre
kg	mass	kilogram
s	time	second
°C	temperature	centigrade
J	energy	joule
W	power	watt ($\mathrm{W} = \mathrm{J\,s^{-1}}$)

[Only include the units that your SRS actually uses. —TPLT]

[Derived units, like newtons, pascal, etc, should show their derivation (the units they are derived from) if their constituent units are in the table of units (that is, if the units they are derived from are used in the document). For instance, the derivation of pascals as $\mathrm{Pa} = \mathrm{N\,m^{-2}}$ is shown if newtons and m are both in the table. The derivations of newtons would not be shown if kg and s are not both in the table. —TPLT]

[The symbol for units named after people use capital letters, but the name of the unit itself uses lower case. For instance, pascals use the symbol Pa, watts use the symbol W, teslas use the symbol T, newtons use the symbol N, etc. The one exception to this is degree Celsius. Details on writing metric units can be found on the NIST web-page. —TPLT]

1.2 Table of Symbols

The table that follows summarizes the symbols used in this document along with their units. The choice of symbols was made to be consistent with the heat transfer literature and with existing documentation for solar water heating systems. The symbols are listed in alphabetical order.

symbol	unit	description
A_C	$\mathrm{m^2}$	coil surface area
A_{in}	$\mathrm{m^2}$	surface area over which heat is transferred in

[Use your problems actual symbols. The si package is a good idea to use for units. —TPLT]

1.3 Abbreviations and Acronyms

symbol	description
A	Assumption
DD	Data Definition
GD	General Definition
GS	Goal Statement
IM	Instance Model
LC	Likely Change
PS	Physical System Description
R	Requirement
SRS	Software Requirements Specification
SFWRENG 4G06	[put an expanded version of your program name here (as appropriate) —TPLT]
TM	Theoretical Model

[Add any other abbreviations or acronyms that you add —TPLT]

1.4 Mathematical Notation

[This section is optional, but should be included for projects that make use of notation to convey mathematical information. For instance, if typographic conventions (like bold face font) are used to distinguish matrices, this should be stated here. If symbols are used to show mathematical operations, these should be summarized here. In some cases the easiest way to summarize the notation is to point to a text or other source that explains the notation. —TPLT]

[This section was added to the template because some students use very domain specific notation. This notation will not be readily understandable to people outside of your domain. It should be explained. —TPLT]

2 Introduction

Do you know the game Yahtzee? Played with dice, it's like Poker but more varied, requiring different skills, without betting. It uses 5 dice, with the usual dots representing the numbers 1 to 6. In rolling the dice, players try to create specific formations, thereby scoring points. At the end of a set number of rounds of dice rolls, the player with the higher score wins. But the cube isn't the only possibility for dice as the octahedron, with 8 sides could be used. Scoring could likewise be done each round. The number of dice could be changed. These suggest an expanded version of Yahtzee.

This project creates an online multiplayer game platform that allows for the creation of custom Yahtzee-like games and variants. This family of games will come with some presets such as classic Yahtzee, Dr. Paul's octahedron version, and more, but will allow for the users to set their own variables such as number of dice, what kind of dice, and some elements of scoring, and then play that game. Kinds of dice would include cubes and octahedrons, among other multi-sided dice, and scoring could be calculated at the end of the game, as in classic Yahtzee, or on a per-round basis, where hands go in a head-to-head matchup.

This section includes a general overview of the entire SRS document providing descriptions of all sections. It also outlines of the areas of knowledge needed to grasp the documentation accurately.

2.1 Purpose of Document

The Software Requirements Specification (SRS) document aims to clearly define the functional and non-functional needs of the project. So all parties involved have a common understanding of the objectives and requirements of the project. Throughout the whole software lifecycle, the SRS will serve as the development team's fundamental guide, aiding in the phases of design, implementation, and testing. In addition, it will facilitate effective communication between stakeholders and the development team while creating a framework that ensures that the final result satisfies user requirements and is in line with the project's pre-determined goals.

2.2 Scope of Requirements

The scope of the project involves developing a modified Yahtzee game that features adjustable numbers of playable dice and various game attribute variations. This game will be available for both offline and online play, offering single-player and multiplayer. The scope does not include the implementation of advanced graphics, advanced computer opponents and other board games.

2.3 Characteristics of Intended Reader

The intended readers of this SRS document are mainly software developers and game designers with expertise in game development and design using game engines, in this case, Godot. They should possess a solid understanding of game mechanics, particularly those related to dice games, in this case, Yahtzee, and have experience in understanding how multiplayer games work. Additionally, experience with C# and .NET is necessary for readers due to the development being done using C# on a .NET framework. A high school level of probability theory understanding is also recommended for comprehending the game's underlying mechanics and probabilities due to the differing number of playable dice. It is assumed that readers are well-versed in these areas as it will allow readers to understand certain decisions made by the team.

2.4 Organization of Document

This SRS document is structured to provide a clear roadmap for readers. The sections are as follows:

- **Introduction**
  - Outlines the purpose and scope of the SRS.
- **General System Description**
  - Outlines an overview of the project, along with user characterstics, system constraint and the interfaces between the system and its environment.
- **Specific System Description**
  - Indepth system description containg high-level probelem and goal description, along with solutions characteristics, assumptions, definiations and instance models detailed functionalities and features of the system.
- **Requirements**
  - Outlines both functional and non-functional requirements of the system.
- **Likely Changes**
  - Outlines anticipated modifications to the system.
- **Unlikely Changes**
  - Outlines ascpects of the system that will remain static.
- **Traceability Matrices and Graphs**
  - Tracking of requirements throughout the development process.

- **Development Plan**
  - Outlines the general timeline and milestones for the project.
- **Values of Auxiliary Constants**
  - Outlines constant parameters used in report.
- **Commonalities**
  - Outlines commonalities between differnt aspects of the system.
- **Variabilities**
  - Outlines variables between differnt aspects of the system.
- **Parameters of Variations**
  - Outlines the different variations of aspects of the system.
- **Appendix — Reflection**
  - Indepth insights into the development process and lessons learned.

3 General System Description

This section provides general information about the system. It identifies the interfaces between the system and its environment, describes the user characteristics and lists the system constraints.

3.1 System Context

System context includes an abstract view of the software following the design pattern of Inputs → Calculations → Outputs. The user provides inputs, such as game settings and preferences, the system performs the necessary calculations (e.g., score tracking, dice roll simulation), and then outputs the results to the user. The system will also interact with external entities, such as players in a multiplayer environment.

The following high-level requirements are relevant to this context:

- The game must support online multiplayer functionality, allowing two players to play against each other.
- The system must calculate scores based on standard Yahtzee rules.
- The game will feature realistic dice rolling physics and random outcomes.
- The game will provide a user interface that displays essential game information (scores, player names, dice states).
- Players will be able to customize game settings and access different game modes.

Inputs:

- Game settings (number of dice, player names, game modes).
- Player actions (dice selection, roll decisions).

Outputs:

- Game results (scores, dice outcomes).
- Feedback on game progress (turns, current scores, remaining rolls).

External Entities:

- Other players (for online multiplayer functionality).
- Peer-to-peer networking services for connecting players.

User Responsibilities:

- Provide valid game input data, such as names and settings.
- Make strategic decisions regarding dice rolls and game modes.

System Responsibilities:

- Ensure that all input data is properly validated (e.g., detect type mismatches).
- Perform calculations for scorekeeping and game logic.
- Provide feedback and results to the user in an intuitive interface.

Additionally, the system is intended for casual use, primarily focused on entertainment and social interaction. It is not a mission-critical or safety-critical system, which will influence the degree of formality in the system design.

3.2 User Characteristics

This section summarizes the knowledge and skills expected of the user:

- The end user of the Yahtzee 3D game should have an understanding of the Yahtzee game, including its rules and strategies.
- Users should have a rudimentary (elementary school level) understanding of probability theory, as it is important for comprehending the game's mechanics and making informed decisions during gameplay.
- Familiarity with casual gaming and turn-based gameplay will be beneficial for the user's overall enjoyment and understanding of the pacing of the game.
- Basic digital literacy is expected, particularly in understanding how to adjust game settings, access help menus, or troubleshoot simple connectivity issues.

3.3 System Constraints

The following constraints must be adhered to during the development of the system:

- The game must use the Godot engine exclusively for its development.
- The codebase must be written in C# and use .NET framework exclusively.
- The system must support peer-to-peer and multiplayer capabilities.

4 Specific System Description

This section presents the problem description, which gives a high-level view of the problem to be solved. This is followed by the solution characteristics specification, which presents the assumptions, theories, definitions and finally the instance models.

4.1 Problem Description

Games are a staple of entertainment and are used to bring people together for both competition and fun. There can often be a desire to share the experience of playing a game with someone but meeting can be hard or impossible in-person and so having an online version of popular games allows for such opportunities. One such game is the game of Yahtzee, which while it does have online versions, are limited to the classic rule-set and do not allow for variants to be designed and played. *Dice Duels: Duel of the Eights* looks to solve the issue of not having online access to Yahtzee and the ability to create custom versions of the game and play them.

4.1.1 Terminology and Definitions

This subsection provides a list of terms that are used in the subsequent sections and their meaning, with the purpose of reducing ambiguity and making it easier to correctly understand the requirements:

- Dice rolls can be given in the form AdX where A and X are variables representing the number of dice and number of sides, respectively. When A is 1, its inclusion is optional. For example 4d6 represents rolling 4 6-sided dice, and a d12 represent 1 die with 12 sides. These terms can also be used in contexts such as "we are playing with d8s".
- Since a fundamental mechanic of *Dice Duels: Duel of the Eights* is that of re-rolling dice, an additional variable will be added to the notation to denote the number of rolls, AdXrB. For example, typical Yahtzee with 3 rolls would be denoted as 5d6r3.
- Dice typically take the forms of platonic solids, meaning where the dice faces are congruent regular polygons. The five such polyhedra are:
  - A tetrahedron has four faces (d4)
  - A cube has six faces (d6)
  - An octahedron has eight faces (d8)
  - A dodecahedron has twelve faces (d12)
  - An icosahedron has twenty faces (d20)

Other die shapes can be used such as a pentagonal trapezohedron with ten faces (d10) or even dice meant to be rolled lengthwise such as a triangular prism which despite having five total faces, is used as having three (d3), or has caps to prevent rolling an unintended face.

- Dice rolls indicate a value integer when rolled. These can be represented by the numeric value of the integer or by representing the integer value as dots, called pips.
- In typical Yahtzee some patterns for scoring have terms. These will have to be added to and abstracted for a game with a different number of dice, but for a 5d6r3 these would be the scoring opportunities:
  - Rolling for aces, twos, threes, fours, fives, or sixes is rolling for as many of the number.
  - Chance is any combination of dice as a sum of all dice values.
  - A yahtzee is rolling all five of five dice with matching faces.
  - A three of a kind is having at least three of five dice match. For four of a kind, it is having at least four dice the same.
  - A straight is a set of sequential dice values. This can come in the small straight variant with four sequential values, or a large straight where all five dice are part of a sequence.
  - A full house is having two dice of a kind and three of another.

4.1.2 Goal Statements

These primary goals should be achieved in the development of our system, providing criteria for completeness. We have additionally organized stretch goals for further development, but they are not to be a metric for system completeness.

The goal statements are:

GS1: Enjoyable game. The project is more than just a capstone, and we need the game to be an enjoyable experience.

- User feedback collection. Implement a simple feedback system such as rating or comments for use in interviews or surveys to gather player insight.
- Testing iteration. Conduct at least two rounds of user testing to identify and iteratively improve the system based on user experience.
- Quality assets. Ensure that graphics, animations, and sounds are of a quality to add positively to the overall enjoyment.

Measurement: Based on user feedback, a minimum of 75% consider the experience as enjoyable.

GS2: Online multiplayer functionality. We need to be able to connect two concurrent players to play the game together.

- Connection setup. Develop a server-client connection system where two players can connect.
- Game state synchronization. Each player's actions are to be reflected to both user outputs using real-time synchronization.
- Disconnection handling. Design a method to handle disconnections.

Measurement: Two players can connect such that both players can affect the game state and both players are notified of the updates.

GS3: Customizable game settings. Core game elements must be modifiable to create custom Yahtzee variants. As a goal we would need these options to be implemented:

- Dice quantity option. Create an interface element for users to select from at least three options for number of dice.
- Dice type option. Create an interface element for users to select from at least three options for type of dice. These different types of dice will have different number of faces.
- Scoring method option. Offer at least two scoring systems that can be used.
- Timer feature. Implement a timer that can be turned on or off, providing a countdown for turn time.

Measurement: The above options are implemented and are compatible with each another.

GS4: Preset game settings. By having some preset game configurations, it would allow players to more quickly learn the game or jump into an environment that has been tested.

- Create presets. Develop at least three preset configurations and test them for gameplay balance.
- Preset selection menu. Create a simple way to select a preset.
- Preset names and description. Give each available preset a name and a brief description to help players understand them.

Measurement: At least three preset game configurations would be available for players to load up and play.

GS5: 3D dice rolling. Rolling the dice will need to be or look to be three dimensional to recreate the tactile feel of the original game.

- Dice 3D models. Develop 3D dice models, or the appearance of such.

- Dice interaction. Allow for user interaction with rolling dice through clicks or drags.

Measurement: Dice will have the appearance of the preset die shape, and of being rolled, based on a minimum of 75% of user feedback considering it so.

Additional stretch goal statements are the following, and can additionally be considered for when looking to add to the system's complexity and to better fulfill the intended goal of being an enjoyable game.
The stretch goals are:

SG6: Local multiplayer. This would allow for players to play together on a single computer, but would require a different user interface and allow for different user interactions.

- Player input methods. Develop a system to handle inputs from multiple players on the same device.
- User interface adjustments. Design a UI layout suitable for two players sharing a single screen.

Measurement: The ability for two players to play together using a single interface and game instance without an internet connection.

SG7: Singleplayer variants. A singleplayer game could be achieved either through a computer-run opponent in a game, or through a custom designed experience that could leverage the different environment.

- AI opponent development. Create an AI that can play against a human player.
- Difficulties. Design and allow players to adjust the AI opponent difficulty.
- Solo game modes. Design at least one unique solo challenge or mode that provides a self-contained experience.

Measurement: A single person can play at least one variant made specifically to be singleplayer without requiring a second human player to update the game state.

SG8: Online matchmaking. The game would provide users with the option to connect to another concurrent user based on a matchmaking score.

- Player rating system. Develop a rating system to categorize players by experience and skill.
- Matchmaking algorithm. Implement a system to match players.

Measurement: A player can connect to another unknown concurrent player who was selected as a compatible opponent.

SG9: Saving custom game setting. Having this ability would allow for a user who created a custom game variant to save them for the ability to replay it without the need to recreate those specific settings.

- Save and load system. Build a system for players to save custom setting to a file or local storage.
- Edit and delete options. Allow players to edit or delete saved custom settings for better management.

Measurement: A custom game variant, as per the "Customizable game settings" goal and "More game setting customization" stretch goals, can be saved locally and loaded up to be played.

SG10: More game setting customization. Besides the options in the goals section, some additional game customization options would include:

- Scorecard customization. Provide methods for players to adjust what options and hands appear on the scorecards.
- Scoring points options. Offer players the ability to modify the scores of scoring options.
- Additional scoring mechanisms. Include options for different methods of round or game scoring.
- Gambling mechanism. Add additional ways to act on probabilities such as wagering on specific rolls.
- Feedback-based features. Gather ideas through user feedback and testing to further expand available customization options.

Measurement: Additional game options outside the ones listed in the "Customizable game settings" goal would be available.

SG11: Dice customization. Dice could be made to appear differently, either as a means for personalization or for aiding with different impairments. An example could be a dice with pips versus a dice with a numbered faces.

- Dice colour options. Create at least three different dice appearances.
- Dice number representation. Create different ways to represent dice face values such as traditional pips, numbers, and symbols.
- Personalization menu. Implement a menu to allow for easy selection of dice appearance.

Measurement: At least five different dice appearance variants players can choose from, that would appear in the game.

SG12: Post game statistics. This could allow for players to analyze a game after completion in a more quantitative manner, aiding in better understanding statistical probabilities.

- Key statistic tracking. Track important game stats during play.
- Post-game summary screen. Present collected stats on a summary screen after a game.

Measurement: A post-game summary showing at least three key game stats, available after each game.

SG13: Multi-platform support. While most gaming experiences are for windows, this would allow for the game to be run on more than just the Windows operating system, allowing for a wider audience.

- Compile for systems. Compile the created system for other operating systems.
- Platform testing. Test the game on multiple operating systems.
- Cross-platform functionality. Verify the features such as online multiplayer work across different operating systems.

Measurement: The game can be run on operating systems other than Windows.

SG14: Dice highlighting. This would aid in determining what dice are used when scoring.

- Automatic dice highlighting. Implement a system to automatically highlight the dice that contribute to a player's score.
- Optional setting. Allow players to enable or disable the feature.

Measurement: Dice used in scoring will be highlighted when appropriate.

4.2 Solution Characteristics Specification

This section, along with Physical System Description are not included within our document and have been removed from the template. The purpose of this section is to reduce the problem into one expressed in mathematical terms. Mathematical expertise is used to extract the essentials from the underlying physical description of the problem, and to collect and substantiate all physical data pertinent to the problem. Important elements that might otherwise have been in this section can be found in other sections where they may be more pertinent. Given the focus of this section on the physical description, and the fact there is no physical description as we have a software based project that is mostly hardware agnostic where the operating system would be more pertinent to the execution of the program.

5 Stakeholders

The stakeholders for this game project are those with vested interests in its development, release, and continued use. While there may be individuals with vested interest in *Dice Duels: Duel of the Eights*, they all fall within the broader categories below and generally act as a member of the whole rather than an individual stakeholder. Stakeholders for the capstone but not for the project itself are included in some sections below, but are not considered a stakeholder of the project due to not influencing the direction and development of the game outside of requiring a certain challenge level and documentation.

5.1 Traditional Yahtzee enthusiasts

- **Demographics:** Generally older players who have nostalgia for the classic Yahtzee game. They may have experience with the physical dice game playing with family and friends and seek a familiar experience to bring back memories and connect with people.
- **Technical Comfort Level:** May be lacking in experience with technology and be more comfortable with straightforward, user-friendly interfaces and are likely to appreciated clear instructions.
- **Motivations:**
  - *Nostalgia.* They seek to relive the classic Yahtzee experience in a more accessible format.
  - *Social connection.* They are interested in playing with friends they may not be able to meet in person.
  - *Relaxed gameplay.* They generally prefer low-stakes slower-paced gameplay that doesn't require complex strategies or fast decision making. They are more likely to appreciate a more relaxed experience.
- **Preferences:**
  - *Classic game mode.* They'll likely be drawn to a game mode that replicates the original Yahtzee experience.
  - *Simplicity over customization.* While they may be willing to try out some customization, they may prefer options that don't stray too far from the classic.
  - *Minimalistic design.* An interface that's easy on the eyes with intuitive navigation will help make the experience enjoyable for them.

5.2 Video Game enthusiasts

- **Demographics:** A diverse group spanning casual players to more experienced players focused on optimization, they are familiar with online multiplayer games and are comfortable with technology and video games.

- **Technical Comfort Level:** Most typical gamers are comfortable with technology, online multiplayer, and faster-paced games. They enjoy exploring game mechanics and may be open to more complex overlapping game mechanics. Are comfortable downloading a new video game with straightforward procedures.

- **Motivation:**

  - *Challenge and skill expression.* They're interested in games that allow for strategy and competitive play where skill and quick decision-making have an impact on the outcome.
  - *Engagement.* Video game enthusiasts seek a fast-paced experience with more action and reward cycles that provide instant feedback and keep them engaged.
  - *Replayability.* They generally enjoy experiences with unique challenges every time they play and can develop strategies over multiple runs.

- **Preferences:**

  - *Customization and variability.* They are likely to appreciate the ability to customize gameplay to explore possibilities and develop a game system that is challenging but enjoyable for them.
  - *Faster game modes.* They may enjoy options that speed up gameplay, such as round timers or different scoring methods that add intensity and variety.
  - *Reactive feedback.* Faster-paced action and instant feedback to keep them engaged would be preferred over slower gameplay.

5.3 Personas

- **Joan, 55 years old (The Nostalgic Yahtzee Player)** Joan enjoys simple board games that remind her of family gatherings. She values a straightforward interface, classic gameplay, and the option to play casually with friends over the internet when they cannot meet in person.

- **James, 24 years old (The Casual Gamer)** James plays games on the weekends with friends. He enjoys the flexibility of customizable game modes and fast-paced rounds. Alex values multiplayer gameplay with friends and occasional solo play.

- **Julian, 19 years old (The Competitive Gamer)** Julian enjoys strategic games that involve skill, competition, and learning over time. He prefers playing with friends that are similarly competitive, customization options, and post-game statistics to analyze his performance.

5.4 Priorities Assigned to Users

- **Primary Priority:** Casual and competitive video game enthusiasts would be the primary priority as they are the most plentiful and would be most likely to come across the game to play it. There is currently no game that provides customizable Yahtzee-like mechanics, and with feedback and testing, the game can be made to tailor to their preferences and be engaging.
- **Secondary Priority:** Traditional Yahtzee enthusiasts are a secondary priority in that using customization mechanics their preferred classic game can be recreated, but the game system overall would be more than just that. Their needs for a simple and intuitive user interface must be taken into consideration, as must options be available for their experience to likewise be enjoyable.
- **Tertiary Priority:** Outside of the context of stakeholders for the program, there are stakeholders for the capstone project. Their considerations do not shape the game itself as they are unlikely to play the game, but rather require a specific challenge level and documentation to be produced. As such, they can be considered tertiary despite not being stakeholders to the game unless also included in one of the above stakeholder groups.

5.5 User Participation

- **Development Phase:** Selected playtesters and the development team will regularly engage with the game throughout the development cycle to identify bugs and shape the user experience.
- **Testing Phase:** Both playtesters and target user groups will participate in beta testing to validate usability, customization options, and multiplayer functionality.
- **Feedback Phase:** In the feedback phase, when functionality is mostly validated, users will provide feedback through surveys and interviews to refine the game before a final release.

5.6 Ongoing Support

This project will be released with a final build that is complete and of quality. Since this game will also be released under a permissive license, the codebase will be publicly available on GitHub, allowing others to further develop it and add features they themselves may wish to have available, or fix any bugs that may be present. The current developers may also be available in their free time if this project or a spin-off is something that will continue past the capstone project timeframe.

6 Requirements

6.1 Functional Requirements

R1 The game shall support an online player vs player mode, where 2 players can play against each other.

Rationale: Player vs player support is a central functionality for this game and one that enhances the overall enjoyment of the game by fostering a more social game dynamic.

R2 The game shall handle score calculations using similar rules to standard Yahtzee under default game settings.

Rationale: Score calculations that follow from a well established, and well balanced game will help users jump into the game faster and foster a more fair game experience.

R3 The game shall simulate realistic physics for 3D dice rolls with true randomness on every roll that replicates accurately to all users.

Rationale: It is important that players get to visually see the dice roll and feel that the outcome of the roll is not clearly predetermined in order to enhance the user experience and maintain continuity between the game and real life.

R4 The game shall use the real outcome of a roll to get the values from the dice.

Rationale: In order to get true randomness off a simulated dice roll, it makes the most sense to actually just simulate the roll and read the result rather than precalculating the results and forcing the roll to match the output.

R5 The game shall support both regular six-sided dice and dice with more sides, such as octahedral dice.

Rationale: By providing support for dice with more sides, we can increase the level of complexity of the game for users that are looking for a more interesting variation of the game.

R6 The game shall implement a turn based mechanism such that each player (or computer) takes a turn and then their opponent takes a turn, and this sequence repeats.

Rationale: This game makes the most sense in a turn based setup so that both players finish all of their rolls at the same time, and so that the back and forth nature of the gameplay fosters better social engagement.

R7 The game shall allow players to pick which dice they would like to use for each roll, and which dice they would like to omit.

Rationale: This is an important part of standard game rules, and also increases the range of strategic decision making that players get to have.

R8 The game shall display some sort of user interface to display scores, number of rolls, time limits, state of dice, and player names.

Rationale: Some form of user interface is necessary to convey important game information to the player.

R9 The game shall provide controls for the player to modify the game settings to access unique variants of the game.

Rationale: A core part of our team's vision for this game is that players should be able to customize their playing experience by altering things like; number of dice, type of dice used, time settings, and scoring methods.

R10 The game shall provide presets for different game modes.

Rationale: This should help new players get used to the game before exploring more unique setting configurations.

***** The following requirements relate to our stretch goals *****

R11 The game shall support a local player vs player mode, where 2 players can play against each other on the same computer.

Rationale: Given that this is a turn based game it is perfectly possible for this game to be setup with 2 player functionality on one device. This simply increases the number of ways in which the game can be played.

R12 The game shall support a player vs computer mode, where 1 player can play against another entity without needing to find a human match.

Rationale: Player vs computer support is important for instances where a user may not be able to find a human match to play against.

R13 The game shall implement some sort of artificial intelligence for computer players in the player vs computer gameplay option.

Rationale: For player vs computer to function, some sort of artificial intelligence will be necessary to implement the computer player or else there will be no challenge for the human player.

R14 The game shall implement online matchmaking.

Rationale: For players that want to play the game against a real opponent but don't know anyone who is available to play against them, this is a great way for that player to find someone to play against.

R15 The game shall provide the option to save specific game settings to be reused in future sessions.

Rationale: For frequent players of the game, they may have a preferred custom variation of the game settings that they may wish to save rather than having to input the game settings on every new game.

R16 The game shall display round statistics to each player at the end of every round.

Rationale: Players will likely want to see some statistics at the end of the round to quantify how well they played.

6.2 Non-Functional Requirements

NFR1 **Performance** The game shall maintain a frame rate of at least 30 FPS at all times.

Rationale: Players need the frame rate of the game to be high enough at all times to see what's going on.

NFR2 **Usability** The game shall implement a clear and easy to use/understand user interface.

Rationale: Users should not struggle to figure out where controls are or how certain features work. This would create a barrier to entry for new players, and it would be best if players could pick up this game and its controls with little difficulty or time required.

NFR3 **Portability** The game shall be supported on systems running Windows 10 or later.

Rationale: Most PC users use Windows 10 or 11, so ensuring that the game ports well to these operating systems is an important requirement if we want to reach a large demographic of users.

NFR4 **Reliability** Multiplayer games of Yahtzee should crash less than 1% of the time.

Rationale: Players need to feel that the game is reliable and should not need to be concerned about the game crashing in the middle of a round.

NFR5 **Responsiveness** The game shall respond to inputs from the user within 500 milliseconds in the worst case.

Rationale: This figure represents a minimum acceptable response. If a user needs to wait longer than this to see the result of their input, they may become frustrated and even try spamming the control.

NFR6 **Modularity** The game's codebase shall be modular, such that it is easily extendable and reusable, and allows for quick fixes to bugs that may occur.

Rationale: Our team has minimum goals in mind for what this game must be upon release, but to be able to continue updating the game to meet stretch goals, our codebase should be designed with modularity in mind to streamline the development process.

NFR7 **Efficiency** The game shall be optimized to run on lower end systems which might have minimal CPU and GPU resources.

Rationale: The game that our team is developing is not such a graphically or computationally intensive game that it should require high-end PCs to run. Additionally, being able to run the game on lower-end systems will allow our game to reach a broader audience.

NFR8 **Enjoyability** The game shall be found enjoyable by at least 75% of users.

Rationale: The game should appeal to the majority of its users, because if it doesn't then the game serves no particular purpose.

NFR9 **Appearance** The game shall maintain a consistent UI style and 3D visual style throughout all in game views.

Rationale: A style that evokes some sense of continuity throughout the game is important to help users get used to the layout faster and to give the appearance of a more professionally developed gaming experience.

***** The following requirements relate to our stretch goals *****

NFR10 **Portability** The game shall be supported on MacOS devices.

Rationale: There are many users who use MacOS devices and would be a good target audience for this game.

7 Likely Changes

LC1: Adding more ways to find opponents to play with (single player, local multiplayer on same device, etc)

- After setting up our initial on-line multiplayer and making sure that the online multiplayer is stable, we can expand our projects scope by adding other ways that users can find opponents to play against, such as a singleplayer gamemode against Ai, or a local multiplayer mode that can be played on the same device.

LC2: the amount and variability of dice options may increase and decrease based on the scope of the project

- As the Scope of our project increase and decreases, we will consider experimenting with different amounts and types of dice. and should circumstances allow, we may choose to add or remove these different options for our game.

LC3: We may expand comparability to other operating systems depending of the scope of the project (Mac OS, etc).

- While we will initially design our system to work on Windows 10/11 devices, should the scope of our project expand, we are likely to adapt our system to work on other OS systems too such as Mac OS, or Linux, allowing our product to reach more players.

LC4: We may change how the scores are calculated.

- As we change and add various options for the players, such as changing the number and types of the dice, we may need to account for the different probabilities that these new options may create, which in turn would require us to modify our scoring system to account for these radically different probabilities.

LC5: We may expand the amount of players that can play the same game from just the initial 2 depending on scope.

- While initialized for two players, it is known that Yatzee can easily support more players, given that the fundamental mechanics of the game don't change when you add them. Thus, when we stabilize our two-player multiplayer, we could expand upon it to allow more then the initial two players to play the same game, allowing more users to enjoy our product.

8 Unlikely Changes

UC1: We will not remove the multiplayer component of the game.

- Yatzee is a social game of chance that involves chance and strategy in an appempt to get the highest score compared to other players, As such Multiplayer is a core component of the game and it must be included to ensure that an important aspect of the game isn't lost.

UC2: We will not remove the dice and it's probabilities

- Dice are a critical component of Yhatzee and it's variations, and the probabilities that the dices rolls provide are a core component in the way that score is calculated in game. Thus, our game will always involve the usage of dice and the calculation of the probabilities involving them.

UC3: We will not switch from our engine Godot for the duration of the project.

- After conducting research on other game engines, and comparing our options, we have concluded that the Godot engine is the game engine best suited for our project.

UC4: We will always have customization between game variants, and not just presets.

- One of the selling points for this project is to have customised settings for our game, allowing the user to tailor their experience to the way they want it. Thus, it is unlikely we will alter our plans to include this feature.

UC5: We will not change the 3D format of our game to 2D

- When playing a physical game like Yahtzee, one of the most engaging aspects is the action of rolling the dice. We wish to recreate the feel of playing the physical game as closely as possible by allowing our player to be able to visually see the dice roll in a way that mirrors the physical experience.

9 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an "X" may have to be modified as well. Table 5 shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table 6 shows the dependencies of instance models, requirements, and data constraints on each other. Table 7 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

[You will have to modify these tables for your problem. —TPLT]

[The traceability matrix is not generally symmetric. If GD1 uses A1, that means that GD1's derivation or presentation requires invocation of A1. A1 does not use GD1. A1 is "used by" GD1. —TPLT]

[The traceability matrix is challenging to maintain manually. Please do your best. In the future tools (like Drasil) will make this much easier. —TPLT]

	TM??	TM??	TM??	GD1	GD??	DD1	DD??	DD??	DD??	IM1	IM??	IM??
TM??												
TM??			X									
TM??												
GD1												
GD??	X											
DD1				X								
DD??				X								
DD??												
DD??								X				
IM1					X	X	X				X	
IM??					X		X		X	X		
IM??		X										
IM??		X	X				X	X	X		X	

Table 2: Traceability Matrix Showing the Connections Between Items of Different Sections

The purpose of the traceability graphs is also to provide easy references on what has to be additionally modified if a certain component is changed. The arrows in the graphs represent dependencies. The component at the tail of an arrow is depended on by the component at the head of that arrow. Therefore, if a component is changed, the components that it points to should also be changed. Figure **??** shows the dependencies of theoretical models, general definitions, data definitions, instance models, likely changes, and assumptions on each other.

	IM1	IM??	IM??	IM??	4.2.10	R??	R??
IM1		X				X	X
IM??	X			X		X	X
IM??						X	X
IM??		X				X	X
R??							
R??						X	
R??					X		
R??	X	X				X	X
R??	X						
R??		X					
R??			X				
R??				X			
R??			X	X			
R??		X					
R??		X					

Table 3: Traceability Matrix Showing the Connections Between Requirements and Instance Models

	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??
TM??	X																		
TM??																			
TM??																			
GD1		X																	
GD??			X	X	X	X													
DD1							X	X	X										
DD??			X	X						X									
DD??																			
DD??																			
IM1											X	X		X	X	X			X
IM??												X	X			X	X	X	
IM??														X					X
IM??													X					X	
LC??				X															
LC??								X											
LC??									X										
LC??											X								
LC??												X							
LC??															X				

Table 4: Traceability Matrix Showing the Connections Between Assumptions and Other Items

Figure **??** shows the dependencies of instance models, requirements, and data constraints on each other.

10 Development Plan

Please refer to separate development plan located in docs: link

11 Values of Auxiliary Constants

This section, titled "Auxiliary Constants," has been intentionally omitted as it is more appropriate for documents that require the definition and organization of numerous constants and parameters. In the context of our project, which focuses on the game variants, the need for such a section is minimal. The constants and parameters relevant to the game family are already comprehensively addressed in other parts of this document, particularly within the sections dealing with Commonalities, Variabilities, and Parameters of Variations. These sections provide all necessary explanations and definitions in a self-contained manner. Given that these parameters are specific to certain features of the game and are clearly outlined where applicable, adding an additional section on auxiliary constants would be redundant and not add any meaningful value to the document. Therefore, we have chosen to skip this section for the sake of clarity and to avoid unnecessary duplication.

[Show the values of the symbolic parameters introduced in the report. —TPLT]

[The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance. —TPLT]

[The value of FRACTION, for the Maintainability NFR would be given here. —TPLT]

12 Commonalities

This section outlines the shared elements across the different games in the product family, including core mechanics, UI components, and gameplay goals.

12.1 Dice Rolls

A fundamental aspect of the game is its reliance on dice rolls. All variants will incorporate dice rolling as a primary mechanic, with corresponding UI elements to visually represent the action, ensuring consistency across game versions.

12.2 Yahtzee Hands

The scoring system in each game variant will be based on Yahtzee-style hands. While the specific scoring mechanics and available hands may vary slightly, the core objective of rolling a hand that aligns with predefined scoring categories remains consistent throughout all versions.

12.3 Score Sheet Structure

Similar to the Yahtzee Hands mechanic, the structure of the score sheet will be uniform across all game variants. This provides players with a familiar interface and ensures consistency in tracking scores.

12.4 Simultaneous Play

The game is designed for real-time play rather than a turn-based system. Players will always be engaged in the same stage of rolling, preventing any strategic advantage in head-to-head game variants.

12.5 2-Player or Single-Player Only

None of the game variants will support more than two human players. The focus will remain on single-player modes or head-to-head matchups with only two participants.

12.6 Always Playing Against an Opponent

Players will never play in isolation to achieve the highest possible score. Instead, each game will involve competing either against another player or against a predetermined computer score criterion, reinforcing the competitive nature of the gameplay.

13 Variabilities

This section details the variabilities between the different games in the product family, including changes in UI elements, game goals, and core mechanics.

13.1 Number of Dice

The number of dice used in the game can vary between different versions. Some variants may increase or decrease the number of dice to adjust the complexity and strategy of the game.

13.2 Sides on Dice

The number of sides on each dice can differ across game variants. This allows for customization of gameplay, where different variants may feature dice with more or fewer sides, influencing the probability and outcomes of each roll.

13.3 Individual Sides

The specific values or symbols on the sides of each dice can vary between versions. This variability provides an opportunity to introduce different themes or scoring dynamics depending on the game's rules.

13.4 Scoring Calculation

How points are calculated for different hands and the influence of dice rolls on the final score can be customized in different game variants. This allows for changes in how specific hands are valued and how scoring impacts the overall gameplay strategy.

13.5 Time Per Turn

The amount of time a player has to roll the dice and make their selection can be adjusted between variants. Time limits can vary, creating different levels of pressure and pacing in the game.

13.6 Hand Restrictions

Some game variants may restrict certain hands from being scored or may allow hands to be scored multiple times. These variations provide flexibility in game strategy and scoring dynamics.

14 Parameters of Variations

This section details the specific values that the variabilities between the different games in the product family can take, corresponding to the variabilities outlined in the previous section.

14.1 Number of Dice

The number of dice used in the game can range from 4 to 8. This allows for variations in gameplay complexity, depending on the specific variant.

14.2 Sides on Dice

The number of sides on each die can range from 4 to 9. This flexibility in dice sides introduces variability in the probability and potential outcomes for each roll.

14.3 Individual Sides

The values displayed on the individual sides of each die can range from 1 to 9. This variability allows for different numerical or symbolic configurations to match specific game variants.

14.4 Scoring Calculation

Each hand can be assigned any integer score value or modified by factors based on the dice rolls. This provides flexibility in defining how each hand is valued in the different game variants.

14.5 Time Per Turn

The time allotted for each turn can range from 5 seconds to 2 minutes, or be set to unlimited. This parameter affects the pacing of the game, allowing for both fast-paced and more thoughtful gameplay.

14.6 Hand Restrictions

Hands can be restricted by removing up to all but one from play. Additionally, hands can be allowed to be repeated between 1 to 10 times or an unlimited number of times, giving options for different scoring strategies.

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning.

1. What went well while writing this deliverable?

    - The SRS document had a lot of work to be done in a lot of sections. In an effort to mitigate merge conflicts and better divide the workload, a main SRS.tex file was created which would reference and include other .tex files which contain the information for the sections. - John P.
    - All team members were able to contribute to the document. Sections were divided up well so that each team member could focus on their own specific part. - Isaac G.
    - Team members communicated often and effectively. - Isaac G.
    - Code changes were handled well as described in our workflow plan such that we were able to keep track of changes and avoid merge conflicts. - Isaac G.

2. What pain points did you experience during this deliverable, and how did you resolve them?

    - Since out project is solely a software without any hardware components or dependencies, the template was a little difficult to navigate. While we used the default SRS template as guidance, we made use of inspiration from the Volere SRS template for sections that would be more relevant to our own and removed sections that were not relevant from the default SRS template. - John P.
    - Some sections for this SRS were not entirely clear and some sections were not very applicable to our specific project. We were however able to meet with our TA to iron out some of these issues and find a clear path forward. - Isaac G.
    - Our project was able to benefit from a commonalities section but this is not something that any of us knew how to do. But through some discussion with the prof and some research, Nigel was then able to look into this issue and ultimately did a great job building out the commonalities section for the team. - Isaac G.
    - With many busy shedules finding time to meet and work was a challenge. We were however able to compare shedules and find meeting times that we could all make, and everyone communicated well to let team members know when they would have time to get to their part(s) of the deliverable done. - Isaac G.

3. How many of your requirements were inspired by speaking to your client(s) or their proxies (e.g. your peers, stakeholders, potential users)?

- Our primary stakeholders are those who would be interested in playing the game, so indirectly we are members of the stakeholder group as well as our supervisor. As such, through consensus and group discussion, game mechanics that would be interesting and enjoyable were drawn up which determine most of our functional requirements. - John P.
- Our look and feel requirements are all inspired by speaking with potential users. - Isaac G.
- Our functional requirement for being able to customize the dimensionality of our dice is inspired by our supervisor who is a stakeholder. - Isaac G.

4. Which of the courses you have taken, or are currently taking, will help your team to be successful with your capstone project.

    - For the calculations of probabilities, SFWRENG 4E03 proved to be helpful and discrete time markov chains were used to determine probabilities in rolls and how many of a kind. - John P.
    - In thinking how we are to organize the program in terms of architecture styles, SFWRENG 3A04 is helpful in providing examples and information on different architectures and their applications. - John P.
    - Since our project is a game, there will be constant player input and output and interaction through an interface. As such, SFWRENG 4HC3 provided helpful information on the design of user interfaces and principles of good interface design. - John P.
    - All of our software design courses will be crucial for the initial planning phases of this project. And these courses were key to the development of our skillsets relating to gitflow and large project setups. - Isaac G.
    - Our software testing and requirements course is proving to be useful now and will likely continue to be useful as the project continues. - Isaac G.
    - Our object oriented programming course will be useful when coding the project. - Isaac G.
    - Our engineering project courses (1P13, 2PX3, 3PX3) are also helpful for many of the planning schemes that we use and for the familiarity that they have provided us with group work. - Isaac G.

5. What knowledge and skills will the team collectively need to acquire to successfully complete this capstone project? Examples of possible knowledge to acquire include domain specific knowledge from the domain of your application, or software engineering knowledge, mechatronics knowledge or computer science knowledge. Skills may be related to technology, or writing, or presentation, or team management, etc. You should look to identify at least one item for each team member.

- I will need to learn to develop in Godot, as that is the game engine we will be using, as well as learning C# as a language. I have also already learnt more GitHub that before with branched, merging, issues, etc. Finally, I hope to further explore probability throughout this project and more advanced calculations such as DTMC. - John P.
- All team members will need to develop a strong working understanding of the Godot game engine and associated coding language. - Isaac G.
- 2D design skills should be developed by at least 1 team member to be able to create visually appealing user interfaces and graphics. - Isaac G.
- 3D design skills should be developed by at least 1 team member to be able to create visually appealing 3D assets for the game. - Isaac G.

6. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?

    - In terms of learning Godot, there are many online tutorials that can be found and followed along, and all of us should be making use of this resource before we contribute to the project to become accustomed to the process. A second way to acquire experience in Godot is through making a small project on our own once we have followed the online tutorials. This would also be the case for C# and the scripting that is available in Godot. - John P.
    - To learn about the Godot game engine and its coding language, team members can read the documentation, watch tutorials, and follow along in making sample Godot projects. This will be the primary method of learning for all team members. - Isaac G.
    - I(Isaac) will focus on Godot because I have game development experience and feel very comfortable with coding. I also do not feel that artistically inclined which is most beneficial for 2D and 3D design. - Isaac G.
    - To learn about 2D design, team members can watch tutorials and practice using software like Photoshop, GIMP, or PAINT.NET. - Isaac G.
    - To learn about 3D design, team members can watch tutorials and practice using software like Blender. - Isaac G.