**Software Requirements Specification**

**for**

Betrayal Online

**Version 1.0 approved**

**Prepared by Nobody (Caleb Corlett, Ryan Nodarse, Patrick Storer, Ethan Wyman)**

**COS420/520 Intro to Software Engineering**

**2-10-2024**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**3. External Interface Requirements 3**

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

**4. System Features X**

4.1 The System shall be entirely offline with the exception of “Online Play”. (Caleb) x

4.2 The System shall have a volume control slider available. (Caleb) x

4.3 The System shall save game data locally. (Caleb) x

4.4 The System may offer a tutorial. (Caleb) x

4.5 The System may offer support for latest generation controllers. (Caleb) x

4.6 The System may save custom character data locally. (Caleb) x

4.7 The System shall have a spectate feature for eliminated players. (Ryan) x

4.8 The System shall have an exploration System. (Ethan) x

4.9 The System shall have a haunt mechanic. (Ethan) x

4.10 The System shall have a visual representation. (Ethan) x

4.11 The System shall have win/loss conditions. (Ethan) x

4.12 The System shall have a main menu option for eliminated players. (Ryan) x

4.13 The System may allow for in-game communication. (Patrick) x

4.14 The System may offer dynamic notifications to users. (Patrick) x

4.15 The System shall allow the host to select settings before the game. (Patrick) x

4.16 The System shall allow the user to view past changes to their character. (Ryan) x

**5. Other Nonfunctional Requirements 4**

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Nobody | 2-18-24 | Finalized for Deliverable 1 submission | 1.0 |
|  |  |  |  |

# **Introduction (I** think prof said the only sections needed were 3 (UI), 4 (FR), and 5 (NFR), but 1 and 2 seem kind of important. Final call is yours, Ryan, on if you want to do them or not, we will prob be fine either way (You will have to add them back to the Table of Contents if you do). -Caleb**)**

## **Purpose**

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

## **Document Conventions**

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

## **Intended Audience and Reading Suggestions**

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

## **Product Scope**

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

## **References**

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

# **Overall Description**

## **Product Perspective**

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

## **Product Functions**

*<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>*

## **User Classes and Characteristics**

*<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>*

## **Operating Environment**

*<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>*

## **Design and Implementation Constraints**

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

## **User Documentation**

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

## **Assumptions and Dependencies**

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

# **External Interface Requirements**

# 

## **User Interfaces**

*<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>*

*(Also don’t forget the Table of Contents -Caleb)*

## **Hardware Interfaces** (By the way, I really didn’t plan for the ramen noodle mockup to be used in anything as-is -Caleb)

Betrayal Online is a downloadable webapp with support for Windows and MacOS with Linux support a real possibility but overall not a priority.

## **Software Interface**s

Betrayal Online will run on Python 3.12 with Pygame Zero and utilize SQLite for local databases which will store data for local game stats + settings, character info + stats and Haunt phase win conditions + special effects. As of right now, we aren’t sure if we will get to the point of starting development on Online Play so we plan to decide on a backend framework and cloud service when we can decide Online Play is a realistic endeavor with the limited time we have available.

# **System Features**

*<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>*

## The System shall be entirely offline with the exception of “Online Play”. (Caleb)

4.1.1 Description and Priority

The game doesn’t really need to connect to the internet for anything other than online netplay. If everything is utilized correctly, we can achieve this without too much difficulty. Medium-High Priority*.*

4.1.2 Stimulus/Response Sequences

The User should be able to access each of the modes offered (local multiplayer, single player, settings, optional modes (if we get to them)) without requiring a network connection. A network connection cannot be absent for Online Play

4.1.3 Functional Requirements

Haunt rules, character stats, user settings, local gameplay stats (if we get to it), and TBD features stored in a local SQLite database. User should be able to request a reset to default state.

REQ-1: Design With Offline In Mind

## The System shall have a volume control slider available. (Caleb)

4.2.1 Description and Priority

The user should be able to modify the volume of the game’s sounds via sliders and should be retained after closing the game. Medium Priority*.*

4.2.2 Stimulus/Response Sequences

Users should be given volume sliders when they open the settings menu, either via settings option in Main Menu or from settings option from Pause Menu. Modifying these sliders will show instant changes to the game’s volume.

4.2.3 Functional Requirements

Individual variables for Master Volume, Music Volume, System Volume. Master Volume is a multiplier (0.0 - 2.0, default=1.0, hard cap=2.0) that will be multiplied against Music Volume and System Volume values. Music Volume and System Volume (0.0 - 0.5, default=0.25, hard cap=1.0) are actual volume values that will be modified by volume slider. Three slider elements for each of the volume types available in the settings menu to modify each of the three volume types. Upon closing Settings menu, update modified volume values in Settings database.

REQ-2.1: Create Slider Element Class

REQ-2.1.1: Focus On Slider When MouseDown In Screen Region

REQ-2.1.2: Track Mouse Pos Until MouseUp

REQ-2.1.3: Adjust Slider Pointer Position Onscreen

REQ-2.1.4: Adjust Variable Values

REQ-2.2: Create Master Volume Slider

REQ-2.3: Create Music Volume Slider

REQ-2.4: Create System Volume Slider

REQ-2.5: Adjust Volume Sliders To Current State Upon Menu Open

REQ-2.6: Save Volume Data In Settings DB

## The System shall save game data locally. (Caleb)

4.3.1 Description and Priority

High Priority. When the user does something that should have some element of data retained (like modify settings, etc.) the system will save it on the user’s machine to be loaded and applied when the user reopens the game.

4.3.2 Stimulus/Response Sequences

Either the user will finalize their data modification with a confirmation or certain conditions would be satisfied (a local game finishing, window closing, etc.), in response, the system will update a local database on the user’s machine with the modified data. Upon opening the game, the system will check for the existence of a database file, upon finding a database file, the system will take the data from the database and apply them to the applicable areas. If database file is not found, a new database file is created and filled with default data. If error is returned when searching for database file, a new database file is created and filled with default data.

4.3.3 Functional Requirements

Database service that supports saving to local machines (SQLite), variables for any default data that will populate the local database files, function to check for existence of local database file on program launch, function to create local database file, function to handle data retrieval from local database file, function to handle saving data in local database file, function to populate database with default data values. TBD for other situations that require local saving to a database.

REQ-3.1: Check for existing database

REQ-3.2: Create local database file

REQ-3.3: Retrieve data from local database

REQ-3.4: Save data to local database

REQ-3.5: Populate local database with default values

## The System may offer a tutorial. (Caleb)

4.4.1 Description and Priority

Medium Priority. When the user starts the game for the first time, the user may be given the option to run through a quick tutorial that will give the user a rundown on everything they need to know to play the game.

4.4.2 Stimulus/Response Sequences

On game boot, if no local database file found then upon main menu open, offer popup notification giving user yes/no option to play a tutorial. If yes, run tutorial. On tutorial finish, offer popup notification with yes/no option to return to main menu / replay tutorial, if yes, return to main menu, if no, restart tutorial.

4.4.3 Functional Requirements

Bool Flag for first open / new local save database file created. Queue of functions to auto execute upon main menu opening. Function to display yes/no notifications. Function to check if first boot + display yes/no notification for a tutorial. Function to actually run the tutorial. Function to return to main menu from the tutorial (doing any special cleanup if required).

REQ-4.1: New Save Flag

REQ-4.2: Autoplay Main Menu Queue

REQ-4.3: Display YES/NO Notification

REQ-4.4: Ask Tutorial

REQ-4.5: Run Tutorial

REQ-4.6: Return To Main Menu (from tutorial)

## The System may offer support for latest generation controllers. (Caleb)

4.1.1 Description and Priority

Low Priority. Menus and gameplay may be designed to be easily navigated with Xbox Series X and Playstation 5 controllers and offer the same functionality as mouse and keyboard. TBD: may allow rebinding of controller buttons

4.1.2 Stimulus/Response Sequences

When user inputs up/down/left/right either on joystick or directional pad, the ui will offer some visual indication for position the user to select options with ease (hover). Upon user up/down/left/right input, variable pointer will be updated to point to the current ui element in a UI tree. Upon pressing a button designated as confirm, the system will execute the UI function inside the variable pointer.

4.1.3 Functional Requirements

4-ary tree (or cycle) to store UI options for traversal. Variable pointer to whichever UI element is currently being hovered (inside the UI Tree). Function to execute UI func inside variable pointer.

REQ-5.1: Place UI Elements In Tree

REQ-5.2: Define UI Pointer + Confirm And Execute

REQ-OPTIONAL-5.x: Change Controller Keybinds

## The System may save custom character data locally (Caleb).

4.6.1 Description and Priority

Low Priority. When the user creates a custom character, the custom character’s stats and identifying data will be stored in a local database file. If the user accesses the program from another device, they will only be able to access custom character data that is on the new device.

4.6.2 Stimulus/Response Sequences

When a user goes through the character customization menu (currently nonessential, low priority), they will choose to finalize their character. Upon finalization, the system will take the custom character data and use it to populate a local database entry. Upon entering a new game and beginning the character select phase, the system will access the custom character data alongside the default character data and provide the custom characters alongside the default Betrayal characters to the user in a list to choose from.

4.6.3 Functional Requirements

Function to check if Character Data Database (CDDB) file exists to be executed on program start. Function to create local CDDB file. Function to create entry in local CDDB using in-progress character creation data from Character Creation Mode. Function to delete single custom character entry from CDDB. Function to retrieve custom character data from CDDB for selection in the Character Select Phase of a game.

REQ-6.1: Check If CDDB File Exists

REQ-6.2: Create New CDDB File

REQ-6.3: Create New CDDB Entry

REQ-6.4: Delete CDDB Entry

REQ-6.5: Get CDDB Entries For Selection

## The system shall have a spectate feature for eliminated players (Ryan)

4.1.1 Description and Priority

Low-medium priority: the user will be offered a Spectate option after they are eliminated and can no longer take any actions to affect the game. They will still be able to navigate the board, follow other players’ perspectives, and interact with aspects of the game to view more information.

4.1.2 Stimulus/Response Sequences

User dies → System offers a Spectate button → User clicks spectate → System hides death interface, showing the board → User can freely navigate through the board and view other players’ perspectives

4.1.3 Functional Requirements

REQ-7.1: The system shall allow users to select a *Spectate* option after death

REQ-7.2: The system shall allow spectating users to navigate freely around the board

REQ-7.3: The system shall allow spectating users to interact with the board to view information about the rooms and units within

REQ-7.4: The system shall allow spectating users to follow the perspective of the active player’s turn

## The system shall have an Exploration System (Ethan)

4.8.1 Description and Priority

Implementing a robust system for players to explore the haunted house. Priority: High. This is a high priority as it is a main function of the game itself and starting the game.

4.8.2 Stimulus/Response Sequences

* User selects a character and initiates exploration.
* System reveals adjacent rooms and allows movement.

4.8.3 Functional Requirements

Functionality to facilitate player exploration of the haunted house, including room generation, movement, and event handling.

REQ-8.1: Generate Randomized House Layout  
 REQ-8.2: Enable Player Movement Between Rooms  
 REQ-8.3: Display Room Descriptions and Events  
 REQ-8.4: Handle Special Room Features or Events

## The system shall have a Haunt Mechanic (Ethan)

4.9.1 Description and Priority

Developing a compelling system for triggering the Haunt phase and initiating asymmetric gameplay. Priority: High. Another core mechanic of the game is the haunt feature though we may not have to flush it out as clearly as the base board game itself we still will need one.

4.9.2 Stimulus/Response Sequences

* Trigger conditions met based on exploration or event outcomes.
* System dynamically assigns roles and objectives to players.

4.9.3 Functional Requirements

Mechanism to trigger the Haunt phase, assign roles, and introduce asymmetric gameplay elements.

REQ-9.1: Define Haunt Trigger Conditions  
 REQ-9.2: Assign Roles to Players  
 REQ-9.3: Specify Win Conditions and Objectives  
 REQ-9.4: Restructure Gameplay for Haunt Phase  
 REQ-9.5: Enable Communication During Haunt Phase

## The system shall have a Visual Representation (Ethan)

4.10.1 Description and Priority

Providing visual elements to represent the game environment, characters, and events for an immersive experience. Priority: High

4.10.2 Stimulus/Response Sequences

* Players initiate actions such as exploration, interaction, or combat.
* System responds with visual feedback including animations, effects, and changes to the game environment.

4.10.3 Functional Requirements

Functionality to present the game environment, characters, and events visually for an immersive experience.

REQ-10.1: Design and Implement Graphical Representations  
REQ-10.2: Provide Animations for Character Actions and Events  
REQ-10.3: Display Visual Cues for Game State Changes  
REQ-10.4: Allow Customization of Visual Settings

## The system shall have Win/Loss Conditions (Ethan)

4.11.1 Description and Priority

Clearly defining win and loss conditions for all players to provide clear objectives. Priority: Medium

4.11.2 Stimulus/Response Sequences

* Players strive to achieve their respective win conditions while preventing opponents from achieving theirs.

4.11.3 Functional Requirements

Specification of clear win and loss conditions for all players based on the triggered scenario. Whether the player or the villain of the event you should know what is required for you to win or lose so you know what to achieve or what to avoid.

* REQ-11.1: Define win conditions for both traitor and survivors based on the triggered scenario.
* REQ-11.2: Specify loss conditions for all players, including individual and group failures.

## The system shall have a Main Menu option for eliminated players (Ryan)

4.1.1 Description and Priority

Medium-high priority: the user will be offered a Return to Main Menu option after they are eliminated and can no longer take any actions to affect the game. Users will be able to exit from the current game to navigate the main menu to do things such as starting a new game

4.1.2 Stimulus/Response Sequences

User dies → System offers a Return to Main Menu button → User selects button → System loads the Main Menu for the user, eliminating the option to return to spectate the last game → User navigates Main Menu similarly to a fresh start of the application

4.1.3 Functional Requirements

REQ-12.1: The system shall allow users to select a *Return to Main Menu* option after death

REQ-12.2: The system shall allow users to freely navigate the Main Menu, having no limitations from the previous game

## 4.13 The system may allow for in-game communication (Patrick)

4.13.1 Description and Priority

Low Priority: The user may be able to communicate with other players via a chat box using basic text functionality. Users will be able to opt into the communication via a setting.

4.13.2 Stimulus/Response Sequences

Users can type messages into the box and press the ‘send’ button to send a message to the server, which will then be displayed to all users. Other player’s messages will be displayed to the user.

4.13.3 Functional Requirements

REQ-13.1: The system shall allow users to send text messages to other players

REQ-13.2: Allow opening/closing of chat menu

## 4.14 The system may offer dynamic notifications to users (Patrick)

4.14.1 Description and Priority

High Priority: Given the elaborate event system of the game, users must be notified about the results of other player’s turns. The user must be shown information in real time to be aware of the state of the game and other players.

4.14.2 Stimulus/Response Sequences

When other players are completing their turns, popup messages will be displayed to illustrate various changes in the game and/or those player’s standings.

4.14.3 Functional Requirements

REQ-14.1: Indicate types of events that have occured

REQ-14.2: Indicate the result of any actions taken by players

REQ-14.3: Indicate to the player when they are required to take action outside of their “normal” turn.

## 4.15 The system shall allow the host to select settings before the game (Patrick)

4.15.1 Description and Priority

Medium Priority: Before the game starts, the host (or the player in offline mode) will be shown a menu with a variety of customizable options for the game.

4.15.2 Stimulus/Response Sequences

When a game is about to start, the host of the game will be able to toggle many gameplay settings. Upon saving the settings, the game will have the settings applied.

4.15.3 Functional Requirements

REQ-15.1: Show and explain game options

REQ-15.2: Provide tooltips for what each individual option will mean

REQ-15.3:Save user options into a database and adjust game variables

## 4.16 The system shall allow the user to view past changes to their character (Ryan)

4.1.1 Description and Priority

Low priority: the user will be able to hover their cursor over their stats to see what events, items, rooms, and combat have affected their stats and by how much. In this tooltip, the user will be able to confirm any one-time improvements from rooms (e.g. Library, Gym, …)

4.1.2 Stimulus/Response Sequences

User explores and experiences rooms, events etc. → System affects possible stat changes on character → User hovers over their character stats in the HUD → System shows tooltip listing all sources that have changed a specific stat, highlighting one-time affects to notify they can’t be re-used

4.1.3 Functional Requirements

REQ-16.1: The system shall track sources of stat changes on each character

REQ-16.2: The system shall allow users to hover over their stats in the HUD to reveal recent changes to the stat

REQ-16.3 The system shall highlight one-time stat changes to allow the player to easily see if they’ve used special rooms

## 4.17 The system shall allow the user to see recent turns (Ryan)

4.1.1 Description and Priority

Low priority: the user will be able to select a button in the user interface to be able to see every player’s last turn, including movement taken, items gathered, events endured, combat fought, and stats changed. This will be a pop-up interface and will show the order in which a given turn was carried out

4.1.2 Stimulus/Response Sequences

User clicks a Recent Turns button → System opens pop-up interface that shows recent player’s turns → User selects a given player → System shows action-by-action breakdown of that turn

4.1.3 Functional Requirements

REQ-17.1: The system shall keep track of actions that each player performs

REQ-17.2 The system shall keep track of major events (Haunt start, deaths, …)

REQ-17.3: The system shall have a button to open a pop-up Recent Turns interface

REQ-17.4 The system shall chronologically organize each player’s actions and effects during that given turn

## 4.18 System Feature EXAMPLE

*<Don’t really say “System Feature 1.” State the feature name in just a few words.>*

*<\*(We will order these by priority High->Low when we get them all in here. -Caleb)\*>*

4.1.1 Description and Priority

*<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>*

4.1.2 Stimulus/Response Sequences

*<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>*

4.1.3 Functional Requirements

*<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>*

*<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>*

REQ-1:

REQ-2:

# **Other Nonfunctional Requirements**

## **Performance Requirements**

5.1.1 The System shall not crash upon menu interaction 95% of the time (Caleb).

Whenever the user selects an option from a menu, the program should do the

intended behavior, anything unintended should be fixed as early as possible.

5.1.2 The System may not be without music or user input for 3 seconds, 95% of the time

(Caleb). Making sure the user has control over something will help the user

differentiate whenever the program is crashing vs when it is just taking longer than

usual to process.

5.1.3 The System may make any changes of aspect ratios and graphic settings in under 3 seconds 90% of the time (Ryan). This will allow the user to have a satisfying view of the game without causing disruptions to the game state.

5.1.4 The System shall maintain a frame rate of at least 30 frames per second (FPS)

(Ethan). In doing this we can guarantee that smooth and fluid animations are met

making the performance more eye appealing and making the system feel as if it runs smoothly.

*<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>*

## **Safety Requirements**

5.2.1 The system shall provide clear and informative error messages in case of unexpected

errors or invalid user inputs (Ethan). If the user inputs an invalid input or an error

case pops up the system will guide users on how to resolve issues or recover from

errors safely.

5.2.2 The system shall log critical errors for observation and debugging purposes, and may

ask the user for further details about what caused the error.

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>

## **Security Requirements**

5.3.1 The system must implement measures to prevent unauthorized access or tampering

with the game files (Ethan). This will prevent issues that should not occur and will

maintain the integrity of the software.

5.3.2 The system must implement measures to protect the integrity of the host player’s

connection. The system will ask the host to allow connections. If the host

disconnects, the game must end.

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

## **Software Quality Attributes**

5.4.1 The System shall offer the user the option to delete local saved data (Caleb). If the

user wishes, they should have the option to delete their saved local data and reset

their game to a factory default.

5.4.2 The System may have some element on screen moving 95% of the time (Caleb).

Having an onscreen element moving in some way will make the program feel more

appealing and alive.

5.4.3 The System shall allow for at least 4 different aspect ratios (Ryan). This will allow the game to be viewed in the preferred way for several types of computers and monitors.

*<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>*

## **Business Rules**

5.5.1 The system shall enforce role-based access control (Ethan). This turn based controls

only allow the user whose turn it is to perform specific actions or access certain

features based on their assigned roles not allowing players to make moves or

decisions out of turn.

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>