**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**

**3-24-2024**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1. External Interface Requirements 3**

1.1 User Interfaces 4

1.2 Hardware Interfaces 5

1.3 Software Interfaces 5

**2. System Features 6**

2.1 Offline functionality with the exception of “Online Play” (Caleb) 6

2.2 Volume controls (Caleb) 6

2.3 Game data will be saved locally (Caleb) 7

2.4 Tutorial (Caleb) 7

2.5 Support for newer controllers (Caleb) 8

2.6 Custom characters will be saved locally. (Caleb) 9

2.7 Spectate feature for eliminated players (Ryan) 9

2.8 Exploration system (Ethan) 10

2.9 The Haunt (Ethan) 10

2.10 Visual representation (Ethan) 11

2.11 Win/loss conditions (Ethan) 11

2.12 Returning to main menu after being eliminated (Ryan) 12

2.13 In-game communication. (Patrick) 13

2.14 Dynamic notifications (Patrick) 13

2.15 Select settings before game start (Patrick) 14

2.16 Viewing changes to character (Ryan) 14

2.17 Viewing recent turns (Ryan) 15

**3. Other Nonfunctional Requirements 16**

3.1 Performance Requirements 16

3.2 Safety Requirements 16

3.3 Security Requirements 16

3.4 Software Quality Attributes 17

3.5 Business Rules 17

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Nobody | 2-18-24 | Finalized for Deliverable 1 submission | 1.0 |
| Caleb Corlett | 2-25-24 | Implemented Feedback from Deliverable 1 | 2.0 |
| Ryan Nodarse | 3-24-24 | Implemented all changes specified by professor | 2.1 |

# Overall Description

**1.1 Product Perspective**

Betrayal Online is a follow-up product that offers a new video game adaptation of an existing board game. This game will not be a component of any larger system, nor will it have any other systems rely on it. This app aims to contain all features detailed in this document, the many of which will be necessary features. While there are more features that will be useful for additional QoL and game content, the requirements specified in this document detail the guidelines for the necessary base game we are developing, and all the systems included with it.

**1.2 Product Functions**

The major functions this game must perform to enable normal gameplay including menus that allow for game start, settings, rules, and exiting the application. During a game start, a character select will take place, and once begun the game will start its first phase: the exploration phase. During the exploration phase, tiles will be placed onto the board, character models will traverse through the haunted house, and cards will be drawn for events, items and omens. After enough omens are found, the next major gameplay event occurs: the haunt phase. During the haunt, a traitor emerges and turns one player against the rest, enabling combat between units, player-inflicted status effects, monsters, and unit death. Once the game ends, players will be able to exit to the main menu.

**1.3 Design and Implementation Constraints**

One of the major constraints that will occur will be enabling online play. This widens the scope from just local single player or multiplayer to requiring a network to access a shared server, and opens up a new realm for issues caused by players including exploits and cheating. Another major constraint is that this application is a direct video game adaptation of an existing board game, so without receiving permission to partner and work with the original developers of the board game, this video game will never be allowed to be monetized and earn revenue. Further constraints include that this game will be a web application only, only allowing up-to-date Windows operating systems and being unable to be accessed on mobile devices. At this time, no other languages other than English will be added to the game during development.

**1.4 User Documentation**

The game will include user manuals for all the rules, as well as a Grimoire that will contain all the details about different game scenarios, monsters, and traitors. The game will also include a basic tutorial for new players.

**1.4 User Documentation**

The game will include user manuals for all the rules, as well as a Grimoire that will contain all the details about different game scenarios, monsters, and traitors. The game will also include a basic tutorial for new players.

**1.5 Assumptions and Dependencies**

This game assumes the player has some basic knowledge of video game navigation. This game assumes the player has a wifi network when looking to install the game, play in online multiplayer, or change their public profile and settings. Currently no other assumptions and dependencies exist, although further down the line in development it’s more than likely more will arise for the user and application.

# **External Interface Requirements**

# 

## **User Interface**s

1. The logical characteristics of the first Main Menu user interface are a welcoming, mystical background to draw the player in. In this screen, the user can start an Online or Local multiplayer game. The user can also view the Grimoire which contains the Rules of the game and detailed descriptions of each available Event, Item or Omen. The user can access the settings, with more options for Audio, Graphics, and Controls. The player can also exit the game to the desktop. This uses an aspect ratio of 16:9 and resolution of 1920x1080.

1. The logical characteristics of the second Main Menu user interface include a thematic, realism representation of the haunted house to allow for a more serious and immersive quality. The major gameplay options are listed (see above). This uses an aspect ratio of 16:9 and resolution of 1920x1080.

1. The third image is an example in-game user interface. This uses an aspect ratio of 16:9 and resolution of 1920x1080. This interface shows:
   1. The main board, including all explored rooms and present players (center)
   2. The player’s character board with current values of each stat circled (bottom right)
   3. The player’s inventory including all gathered items and omens (bottom center)
   4. The player’s current status effects, such as penalties from certain events and nearby enemies (bottom left)
   5. The menu button to view the Grimoire, access the settings, or leave the game (top left)
   6. The player’s current floor, with buttons to change which floor is being viewed (top center)
   7. The next player’s turn in queue (top right)

## **Hardware Interfaces**

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**

## Offline Functionalities

2.1.1 Description and Priority

The game doesn’t 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*.*

2.1.2 Stimulus/Response Sequences

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

2.1.3 Functional Requirements

REQ-1.1: The system shall offer Local Multiplayer when offline

REQ-1.2: The system shall offer Single Player when offline

REQ-1.3: The system shall offer Settings changes when offline

REQ-1.4: The system shall store rules, stats, and settings locally

REQ-1.5 The system shall offer a tutorial when offline

REQ-1.6 The system shall save custom character data locally

## Spectating

2.7.1 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.

2.7.2 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

2.7.3 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

REQ-7.5: The system shall allow spectating with less than 3 seconds of delay 95% of the time

## Exploration System (Ethan)

2.8.1 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.

2.8.2 Stimulus/Response Sequences

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

2.8.3 Functional Requirements

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

REQ-8.1: The system shall procedurally generate the layout of the house 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)

2.9.1 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 it might not have every single feature from the base board game itself itself

2.9.2 Stimulus/Response Sequences

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

2.9.3 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)

2.10.1 Description and Priority

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

2.10.2 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.

2.10.3 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)

2.11.1 Description and Priority

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

2.11.2 Stimulus/Response Sequences

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

2.11.3 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)

2.12.1 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

2.12.2 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

2.12.3 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

## 

## 2.13 The system shall allow for in-game communication (Patrick)

2.13.1 Description and Priority

Low Priority: The user shall 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.

2.13.2 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.

2.13.3 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

## 2.14 The system shall offer dynamic notifications to users (Patrick)

2.14.1 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.

2.14.2 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.

2.14.3 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.

## 

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

2.15.1 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.

2.15.2 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.

2.15.3 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

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

2.16.1 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, …)

2.16.2 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

2.16.3 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

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

2.17.1 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

2.17.2 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

2.17.3 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

# 

# **Other Nonfunctional Requirements**

## **Performance Requirements**

3.1.1 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.

3.1.2 The System shall 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.

3.1.3 The System shall 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.

3.1.4 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.

## **Safety Requirements**

3.2.1 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.

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

potentially will prompt the user for further details that caused the error (Patrick).

## **Security Requirements**

3.3.1 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.

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

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

host disconnects, the game must end.

## **Software Quality Attributes**

3.4.1 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.

3.4.2 The System shall 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.

3.4.3 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.

## **Business Rules**

3.5.1 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.