Escape Fate

(Horror Game)

Software Requirement Specification

Fall 2022
Version 2.0

Nicholas Montalvo (PM)
Daniel Alcala (APM)
Daniel Calderon
Logan Ashbaugh
Valerie Mora
Peter Leiva

Table Of Contents

1. **Introduction** 3
    - 1.1 Purpose 3
    - 1.2 Scope 3
    - 1.3 Definitions, Acronyms, Abbreviation 4
    - 1.4 References 4-5
    - 1.5 Overview 5
2. **Overall Description** 5
    - 2.1 Product Perspectives 5
        - **2.1.1 System Interfaces** 5-6
        - **2.1.2 User Interfaces** 6
        - **2.1.3 Software Interfaces** 6
        - **2.1.4 Communication Interfaces** 6
        - **2.1.5 Memory** 7
        - **2.1.6 Run-Time** 7
    - 2.2 Product Functions (Use Case Diagram) 7
    - 2.3 User Characteristics 8
    - 2.4 Constraints 8
    - 2.5 Assumptions and Dependencies 8
    - 2.6 Class Diagram 8
3. **Specific Requirements** 9
    - 3.1 External Interface Requirements 9
        - **3.1.1 User Interfaces** 9
            - 3.1.1.1 Main Menu 9
            - 3.1.1.2 Settings Menu 10
            - 3.1.1.3 Save/Load Menu 10
            - 3.1.1.4 Game Screen 11
            - 3.1.1.5 Pause Menu 11
        - **3.1.2 Hardware Interfaces** 12
        - **3.1.3 Software Interfaces** 12
        - **3.1.4 Communication Interfaces** 12
    - 3.2 Functional Requirements 12
        - **3.2.1 Bug Fixes** 12
        - **3.2.2 Interactable Objects** 13
        - **3.2.3 Enemy AI** 13
        - **3.2.4 Cutscenes** 13
        - **3.2.5 Animations** 13
    - 3.3 Performance Requirements 13
    - 3.4 Design Constraints 13
    - 3.5 Software System Attributes 13
        - **3.5.1 Reliability** 13
        - **3.5.2 Maintainability** 14
        - **3.5.3 Portability** 14

Name	**Date**	**Reason for changes**	**Version**
Project Fate: Horror Game	9/2/2022	Creation of SRS document	1
Project Fate: Horror Game	10/2/2022	● Expanded on the purpose of the document so that it was more clear on who it was for. ● Added the discussion of the ability to save the game in the scope. ● Provided a little more information on each class for the use case diagram. ● Expanded more on the non functional requirements. ● Added a class diagram to the document	2

1. Introduction

1.1 Purpose

The purpose of this software Requirement Specification is to outline the goals of the Project Fate Horror Video game experience. This specification will discuss the requirements and Goals discussed with our team creating a unique storytelling experience. These Requirements and Goals are to be met and tested by the end of the Fall 2022 semester for the CSE-4550 software engineering class.

The purpose of this game will give the player a thrilling horror game experience that gets their heart racing. Players will be dropped into a house where they are being stalked by a dangerous monster that has found a home in this luxurious mansion. They will have to collect several items before leaving the house in order to open the exit door and make their escape. These documents are proof that the monster lives inside the house and are necessary for the player to unlock the exit door, where they will have completed the game escaping treacherous fate.

1.2 Scope

The Project Fate Horror game will take players on a unique story driven adventure. Where they will be able to interact with the world around them while doing their best to escape fate. Players will experience an interactive video game with fully developed UI such as a Main Menu and Pause Menu. The interactive menu will also give players the ability to save game states and pick up where they left off if they so choose to exit the current game they are playing. Players will also be able to pick up and use items in their environment to solve simple puzzles and escape the enemy AI without being caught.

Prototype 1: The first prototype will only be playable through the unreal engine. There will be an interactive main menu screen, pause screen, as well as the base environment of the game done. The main level design as well as interactive objects in the game will be finished and ready to play.

Prototype 2: The final prototype of the game will be a finished game that will have a clickable shortcut icon for PC. The game will also start up like any other playable video game as well as showing you the main menu screen. The final cutscenes will be in the final prototype as well. The enemy AI will be complete along with all key elements for creating a scary monster for the player to run and hide from. The sound design for the game will also be done to simulate real world noises that create a very real horror experience for the user.

1.3 Definitions, Acronyms, Abbreviations

- **C++** - is an object-oriented programming (OOP) language.
- **Computer** - A digital piece of hardware that can carry out mathematical calculations, functions, and operations.
- **Game Screen** - The page where the app is played, accessible from Main Menu.
- **Git** - An open source software that helps with version control, tracking files during development.
- **Github** - A internet hosting software that integrates Git to allow collaborative development and programming across multiple devices.
- **Launch Page** - The first page of the app that the user can interact with.
- **Main Menu** - The launch page for the app to access Settings, Game Screen, and more.
- **Mixamo** - A website that develops and sells services for 3D applications, including animations.
- **Pause Screen** - A function in a game that stops the game in the current state and allows the user to change Settings, quit, and potentially more.
- **Software -** A set of computer programs, documents, and data that the user can interact with easily.
- **SRS -** This Document, a Software Requirement Specification.
- **UI** - User Interface, the different elements the user can see and interact with.

- **UML** - Unified Modeling Language (UML), a standardized modeling language enabling developers to specify, visualize, construct, and document artifacts of a software system.
- **Unreal Engine** - Video game engine used for developing games on multiple platforms, especially PC.
- **Use Case Diagram** - A representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.
- **User -** Someone who uses the application or software specified in this Software Requirement Specification.
- **Video Game** - An electronic game that displays generated visual feedback when a user creates input through a device, such as a controller or joystick.
- **Visual Studio** - An integrated development environment developed by Microsoft that is used to develop programs in different languages.
- **Website** - A collection of web pages and similar content that can be accessed by a user under a domain.

1.4 References

1. IEEE Software Engineering Standard Committee, "IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications", October 20, 1998.

2. Epic Developer Community. 2022. *Epic Developer Community Learning | Tutorials, Courses, Demos & More – Epic Developer Community*. [online] Available at: <https://dev.epicgames.com/community/learning>.

3. Awesome Tuts, 2021. *Monster Chase Game Tutorial Unreal Engine C++*. [video] Available at: <https://youtu.be/DwjehZh5YQ0>.

4. Isocpp.org. n.d. *Standard C++*. [online] Available at: <https://isocpp.org/>.

5. Visual Studio- Microsoft. "Visual Studio Code - Code Editing. Redefined." *RSS*, Microsoft, 3 Nov. 2021, https://code.visualstudio.com/.

1.5 Overview

The following information provided in this document is used to describe in detail the type of video game we will be creating. As well as provide any further information on the UI, gameplay, and design of the game.

2. Overall Description

2.1 Product Perspectives

This desktop application will be a first person horror game that will allow the player to save and resume their progress when they so choose to exit the game. The user will also be able to access most main menu settings through the pause screen in the game.

2.1.1 System Interface

The games interface will be developed using the Unreal engines Blueprint feature & if needed be edited in C++. The game is expected to have at least five menus:

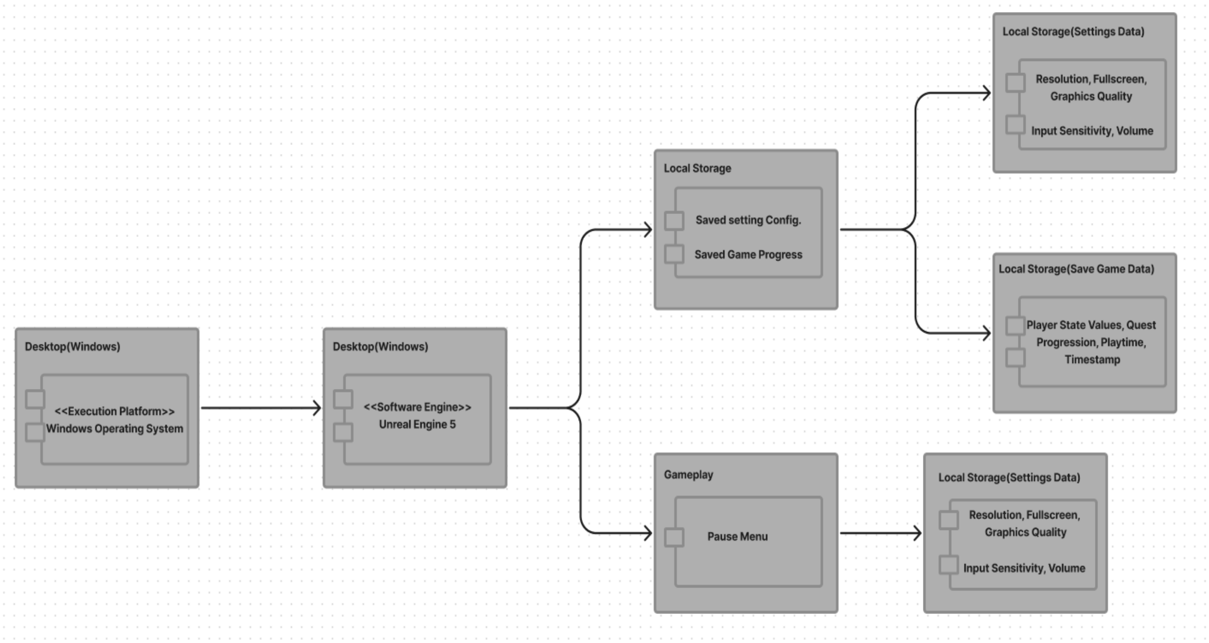
1. **The Main Menu -** It is the launchpage where the user can access other menus.
2. **The Load/Save Menu -** allows the user to load a previous save to resume, or delete an existing save to create a new one.
3. **The Settings Menu -** allows the user to adjust the game's volume or graphic quality.
4. **The Game Screen -** this is where the game is played.
5. **The Pause Menu -** this allows the user to have access to the load/save menu and the settings menu. The user can also choose to return to the main menu of the game as well as quit the current game running.

The Deployment Diagram shown above illustrates the interface and information flow between software and hardware components in the app.

2.1.2 User Interface

When first starting the desktop application the user will see a wash screen introducing the engine used to develop this horror game as well as our studio name. The user will then see the main menu screen where they will have the options to select a new game, load a previous game, adjust the games settings, or exit the games application. After the user accesses a new game or load previous game the user will have access to the game screen where they can play the game using a mouse and keyboard. They can access the pause menu by pressing the Esc. key, the pause menu will give the user access to main menu functions as well as allow them to exit the game.

2.1.3 Software Interfaces


Desktop(Windows)
<<Execution Platform>>
Windows Operating System
Desktop(Windows)
<<Software Engine>>
Unreal Engine 5
Local Storage
Saved setting Config.
Saved Game Progress
Gameplay
Pause Menu
Local Storage(Settings Data)
Resolution, Fullscreen, Graphics Quality
Input Sensitivity, Volume
Local Storage(Save Game Data)
Player State Values, Quest Progression, Playtime, Timestamp
Local Storage(Settings Data)
Resolution, Fullscreen, Graphics Quality
Input Sensitivity, Volume

The software that will be used for this game design will be the Unreal Engine. This engine will be used to create the game for desktop release.

2.1.4 Communication Interfaces

This desktop application will not feature any communication interfaces. It is a single-player game.

2.1.5 Memory

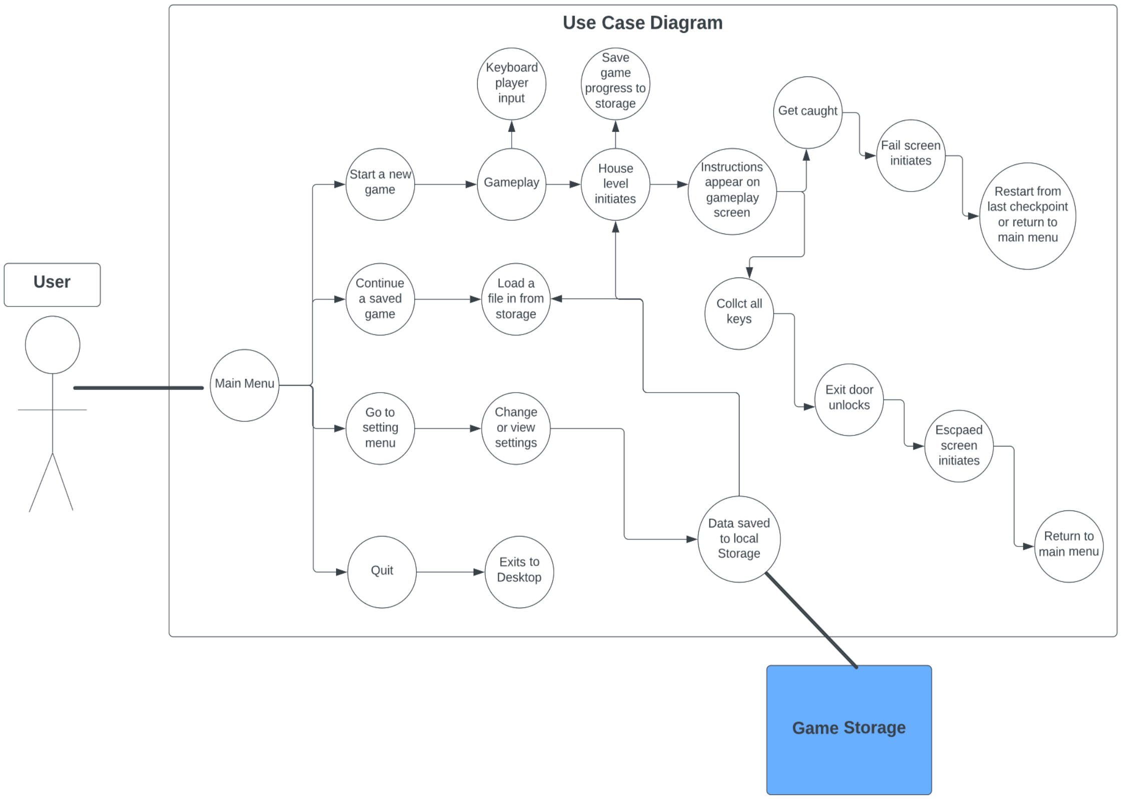
Based on previous projects developed by our individual developers. The game is expected to take up about 25 - 30GB of storage memory. This could increase or decrease depending on the game's optimization.

2.1.6 Run-Time

After the game is downloaded to the user's desktop, it should run when the user opens it and starts it until the user decides to close the desktop application.

2.2 Product Functions

This game is designed and created to bring a unique horror game experience to users looking for a fun gaming experience that will get their heart racing. It will be played by those who enjoy playing unique first person horror games. Consult the use case diagram below for the product functions that the user can interact with.



2.3 User Characteristics

The average user that will be playing this game is anyone who enjoys playing desktop games. This game's goal is to give the user a fun first person horror experience that they can walk away from enjoying and wanting to play again and again.

2.4 Constraints

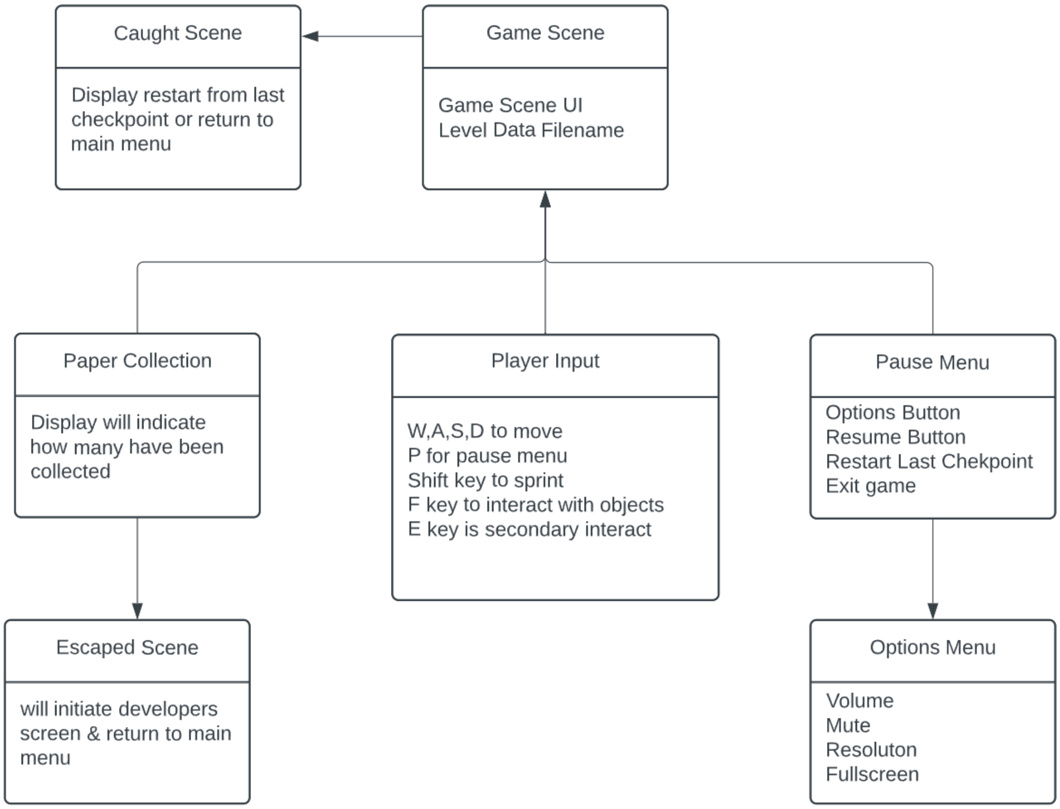
The first publication of this game will only be available for desktop download. Specifically this version of the game will only be playable on windows. No current plans to adapt the game to other platforms have been as that is beyond the scope of this project.

2.5 Assumptions and Dependencies

We expect an average desktop to have enough processing power, memory and power supply to properly run the game. This version of the game might require more powerful desktop equipment to operate properly.

2.6 Class Diagram

This diagram illustrates the basic structure of the main gameplay scene.



3. Specific Requirements

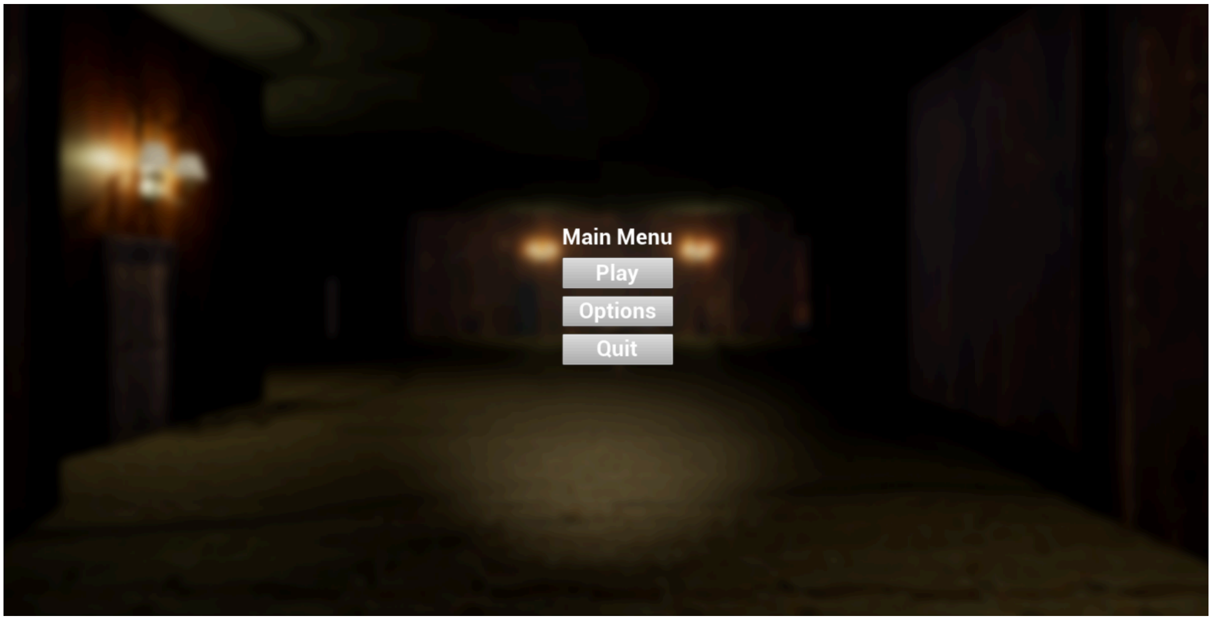
3.1 External Interface Requirements:

3.1.1 User Interfaces

This section of the SRS will show the current layout of the game that we have developed. This is our prototype version of the main menu screens as they are subject to change as the details of the game aesthetic are roughed out. However the current layout of the settings of the game will not be changed. The current layouts are shown below.

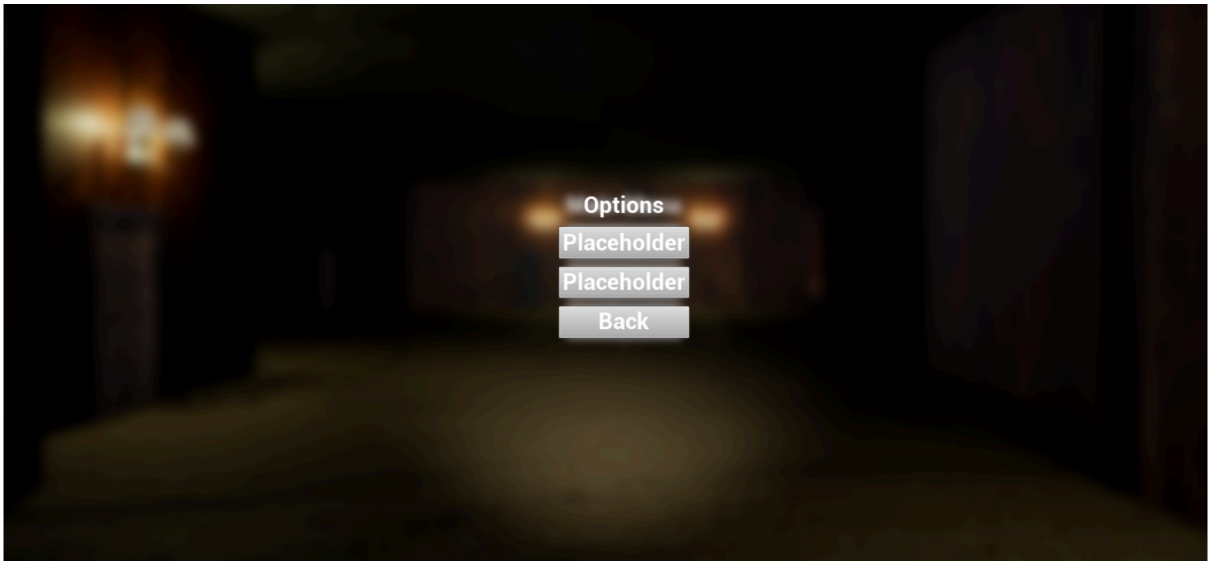
3.1.1.1 Main Menu

After the user has clicked the desktop icon and seen the wash screens introducing the engine and studio they will be shown the main menu. From here the user can choose to start a new game/ load a previous game. They have access to the games settings menu where they can change the game volume or change the resolution of the game. Also if they so choose they can exit the game from the main menu.



3.1.1.2 Settings Menu

If the user decides to access the settings from the main menu screen they will have access to game details that they can change if they so choose. Such as the resolution of the game, look sensitivity, in game volume, as well as different screen size modes.

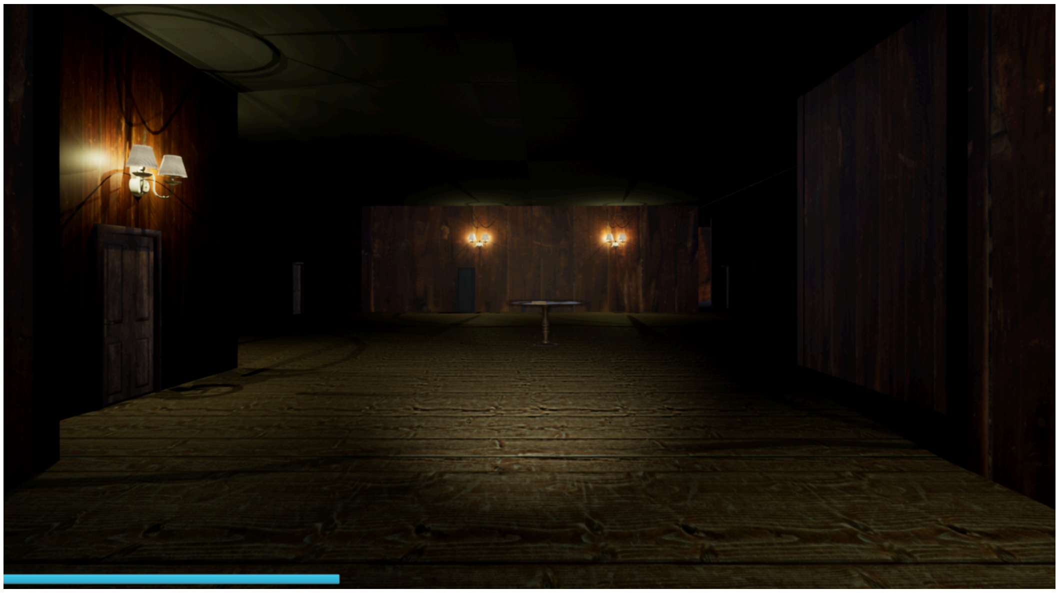


3.1.1.3 Save/Load Menu

For now we will implement a simple menu that the user can access to load previously saved games or if they so choose they can also overwrite or create a new saved game. However we plan to change this so that the user will have the option to actually click a continue button straight from the main menu.

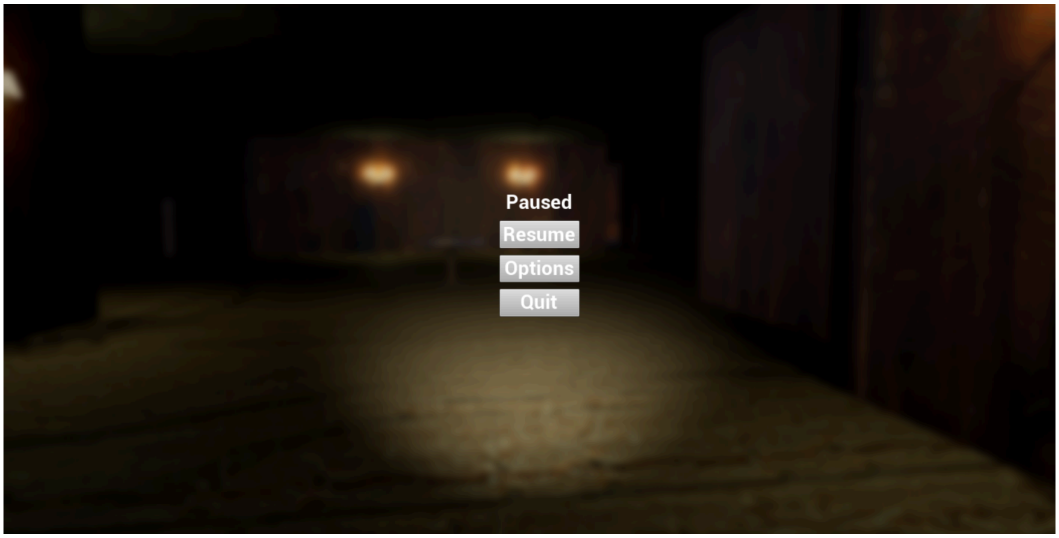
3.1.1.4 Game Screen

This will be the game screen that the user sees when they jump into a game file of their choosing. As you can see there is not much design to our game yet as we are fleshing out key details for playability first. But the design will be simple and have little to none clutter in the user's face.



3.1.1.5 Pause Menu

We decided to implement a pause menu for our user so that they can decide to pause the game instead of having to exit the game, if they need to step away for a minute but want to jump right back into playing. From here the user can also access certain features from the main menu screen as well. Such as the setting menu, and the exit option.



3.1.2 Hardware Interfaces

The user will need a couple of external hardware devices for the game to function properly. The user will first need a mouse and keyboard to be able to move and look around in the environment. As well as some speakers or headphones to be able to listen in game.

3.1.3 Software Interfaces

The game will store the saved data on the users desktop locally. There will be no external software required. When the user decides to save it will store it locally.

3.1.4 Communication Interfaces

The game will have no need to communicate with an external system after it has been downloaded. It will only be on the users desktop.

3.2 Functional Requirements

This section will be on the clients requirements on the final prototype of the game. These requirements will be the minimum requirements needed to make a fully functional horror game.

3.2.1 Bug Fixes

We currently do not have a first prototype of the game so we do not have a list of bug fixes to work with. But we do expect some to pop up and are prepared to handle them as they come at us.

3.2.2 Interactable Objects

We expect to have interactable objects in the game such as keys and information to help the player to progress. As well as help to immerse the player in a more interactive storytelling environment.

3.2.3 Enemy AI

We are currently developing an enemy AI that will be able to track the player as it moves through the environment. We expect to develop an AI that could also interact with the environment such as opening doors and seeing light emitted from the player's flashlight. We also want the AI to be able to miss seeing the player, the player will be using the environment to hide from the enemy AI.

3.2.4 Cutscenes

We are currently developing the game so that it contains cutscenes that help transition the story of the game smoothly from one scene to the next or from one environment to another.

3.2.5 Animations

We will also be trying to implement animations such as the character opening the door that will be happening at the beginning of the game. However this is our last priority because it falls a little far from our scope of this project.

3.3 Performance Requirements

This desktop application will be compatible with Windows version 10 and higher. We have no place to publish this game as of yet as it is beyond our scope of this project to do so.

3.4 Design Constraints

The user must have space to download the game otherwise it will not download properly.

3.5 Software System Attributes:

3.5.1 Reliability

We predict confidently that this desktop application will be able to run on a compatible desktop with minimal issues.

3.5.2 Maintainability

This desktop application of Project Fate the horror game is made to specifications by the team and the client. After further discussion with the team there have been talks of possibly further improving the game by staying in touch through discord. We plan to make customized assets so that we may be able to publish this game on steam one day. There will always be an open conversation with the team to further improve this game.

3.5.3 Portability

We do not have any further plans for the game to be ported to another gaming platform as of late.

Client Signature:

Signature: ___________________________________

Date: _______________________________________