Metroidvania Game:

Subterra

Department of Computer Science and Engineering

California State University, Fullerton

Project Advisor: Lidia Morrison

Submitted By: Andrew Ghaly, Jason Zhu, Matthew Iversen, James Pham

Course: CPSC 491-04

Spring 2023

Table of Contents

Table of Figures

Abstract:

Entertainment takes shape in various forms for all individuals, but it all serves the same purpose of providing a light-hearted environment for that partaking. One form of entertainment is the gaming industry where there are a plethora of game types to please all and any who wish to partake. Subterra is of a genre type called "Metroidvania", these types of games often revolve around fast-paced movement, linear traversal through the game world, fighting various enemies, progression through item unlocks, and of course boss fights. The aim of this project is to delve into the world of game design – which is to say, to create an appealing and playable world, an intuitive and iterative state machine, competent algorithms usage, and a sense of entertainment that appeals to the player.

1.0 Introduction:

The gaming industry is wide and varied in order to appeal to as many individuals as possible, as any good form of entertainment aims to catch the attention of all near and wide. Composing a game is a rather involved process that entails varying degrees of detail depending on the depth wanted by the developer. Everything from the minute aspects of gameplay that are crafted with the player's enjoyment in mind, to considering the environment of the game to avoid complicated interactions during play. The right game blends a competent code design that allows for flexibility and adjustments, while also focusing on features that enhance the enjoyment a player might have while playing the game.

2.0 Problem Statement:

Gaming, just like any other form of entertainment, is rather involved in its production, there are many aspects to it that divide the good games from the bad. The core of any form of entertainment is to bring satisfaction and enjoyment to the user but to accomplish this in a game, one must focus on the interaction a user will have with your product. Production of a game revolves around three critical criteria – the gameplay elements presented to the player and their unique interaction with said gameplay elements, the environment of the crafted world presented to the player and the interactions with the said world, and finally the interactions and behaviors of non-player characters that enrich the gaming experience.

2.1 Problem Definition:

There are three major problems that we are focusing on for this project. The first is a method to universalize the operation of all the tasks the game undergoes while running. This means that a system that acts as a general hub or controller that automates various tasks at any time would be highly beneficial. The second is a method that addresses the world design and the complexity that is required to make adequate traversal entertaining but not tedious. Lastly, the third is a method that involves the pathing and understanding that all non-playable characters in the game will follow, we'll need something that is smart enough to track targets but also do so in a reasonable amount of time.

2.2 Problem Rationale:

By addressing these problems, we'll be implementing systems that allow for upgradability and expansion during any iteration of the project. This is important because in the gaming design world, a product will undergo various amounts of play testing while under review by shareholders that hold stake in the product. Thus being able to make quick and refined adjustments to isolated systems is critical in developing a good game. Furthermore, by having this iterative design, the costs of game design would drastically be reduced due to designing a system that considers the possibility of frequent changes or additions.

3.0 Proposed Method of Solving the Problem:

The proposed methods that we've found to be ideal for our issues is to have a state machine to control the operation of the entire game. This method will allow for a central control hub that will reference various other scripts that handle various other tasks in order to manage multiple operations. Simply put, this means that whenever a certain gameobject requests a task to be done, regardless of what game object type it is, it will be fulfilled by the state machine. This addresses the issue of needing a universal and versatile center of operations. The second issue of creating adequate traversal that's entertaining but not tedious will be addressed by creating various iterations of the map through diagrams, once imported, the map will be play tested for tolerances, and redesigned where seen fit. The third issue of competent non-player characters present in the game will be addressed by implementing an A* algorithm to their pathing and decision making.

4.0 Project:

The purpose of Subterra is to delve into the world of game development and learn what it takes to make a good and desirable game. The many facets of game development involve a trial-and-error approach that includes the attempt to develop a system to handle certain tasks, putting that system out for testing, and then receiving feedback and adjusting anything that needs fixing. This is the ideal format for any software development and will teach what it takes to make a successful product that consumers readily enjoy; furthermore, Subterra will provide a form of entertainment to the users interested in it. Entertainment is a huge part of everyday life and is researched to be critical to a long life due to its stress-relieving properties, therefore, this game hopes to provide another avenue for recreational activity.

4.1 Significance & Future:

This project is planned to be hosted on a indie game focused website called Itchi.io where the product page will be available to anyone and the game will be downloadable any time. The significance of this project involves the entertainment value that it provides to anyone who is willing to play it, and encourages indie game development to be more widespread.

5.0 Objectives:

The objective of Subterra is to understand the elements that make a game, which includes everything from the UI layout to the expected interactions and familiarities the user will expect when using the software. The game will run as an executable in a Windows environment. The genre requires a significant amount of planning and design, including creating a world map, designing abilities and power-ups, and developing enemy types and boss battles. This process can teach developers the importance of careful planning and iteration, as well as the value of feedback from playtesting.

Developing a game like Subterra requires knowledge of a wide range of game development tools, including game engines, art and sound creation software, and programming languages. Working on a project like Subterra can teach developers how to utilize these tools effectively and efficiently, improving their skills and knowledge in the process. Finally, creating a game like Subterra requires teamwork and collaboration. Developers must work together to create a cohesive and engaging experience for players, with each team member contributing their unique skills and perspectives. This process can teach developers the importance of communication, collaboration, and project management, which are essential skills in any game development project.

6.0 Activities:

6.1 Assets, Resources, Research:

The initial steps of this project will involve the research of an ideal map and art style for the game, since visually, most of the game will be represented through its assets. This will involve creating various prototypes and samples of character design, world design, and various tokens for the world population. Furthermore, there must be work done on the sound design for all aspects of the game – movement, jumping, background, successes, failures, etc. Models for all the resources must be made and tested to make sure their proportions are appropriate for the game's design. And finally, world development entails a base outline that will serve as a template for various levels.

6.2 Initial Development:

The phase of test fitting all the researched elements to ensure their compatibility with one another and making any corrections that need to be done. This also includes the implementation of a rough code design for all the operational elements and assuring a certain degree of compliance.

Subterra would need to go through an initial development phase before it could be released. During this phase, developers would focus on planning and design, creating a concept for the game, and determining the game's core mechanics and features. This phase would likely involve creating a detailed game design document, which outlines the game's world, characters, abilities, enemies, and progression.

Once the initial planning and design phase is complete, developers would move on to prototyping and iteration. This phase involves creating a playable version of the game, which allows developers to test and refine the game's mechanics and features. This process can involve multiple rounds of iteration, as developers fine-tune the game's mechanics and incorporate feedback from playtesting.

6.3 Testing of features:

Bug testing is critical for game development because functions and features that seem to be implemented well may fail due to unforeseen issues; therefore, feature testing mid-development will ease final development issues. During this phase, developers would focus on ensuring that all of the game's features are functioning as intended and are free from bugs and glitches. This process involves a combination of automated testing, where developers use software to identify potential issues, and manual testing, where testers play through the game to identify any issues that may have been missed by automated testing.

Testing of features is essential for ensuring that the game provides a high-quality experience for players. By identifying and fixing any bugs or glitches, developers can ensure that the game is stable and that players can progress through the game without encountering any issues that might detract from the overall experience. Additionally, testing of features can help developers to identify any areas of the game that may need further refinement or iteration, allowing them to make changes before the game is released to the public.

6.4 Final Development:

During this phase, developers focus on polishing the game to ensure that it is as enjoyable and bug-free as possible. This phase involves extensive testing, optimization, and bug fixing to ensure that the game is stable and that players have a smooth and immersive experience. Developers may also focus on improving the game's graphics, sound, and music during the final development phase. This can involve refining character designs, improving the game's audio quality, and creating a soundtrack that fits the game's mood and atmosphere. Additionally, developers may work on improving the game's user interface, making it easier and more intuitive for players to navigate the game's menus and systems.

7.0 Development Environment:

For this project, we plan to mostly be using C# as the primary coding language since the development of the game will be done using Unity. For map-level design Figma will be used to do the initial design and unity for the rest of the drawing out assets into the actual game. Unity is a popular game engine that provides developers with a robust and flexible set of tools for creating games. Its 2D tools allow developers to create high-quality graphics and animations, while its physics engine makes it easy to implement complex game mechanics. In terms of IDEs, the built-in IDE for Unity will be used as well as Visual Studio.

Type	Software
Programming language	C#
Element Design	Unity
IDE	Unity, Visual Studio

Figure 1: Development Req. Table

8.0 Operational Environment:

Subterra will be an executable that can run on any Windows or Mac enabled device.

Minimum System Requirements
CPU: Dual Core Processor
GPU: Integrated Graphics
RAM: 4GB
STORAGE: 300MB

Figure 2: System Req. Table

9.0 Overall Description

Subterra is designed to be easily downloaded from an itch.io page and played from the browser. It is designed to offer a 2D exploration experience with a creative map layout and custom handmade assets. The game is designed to offer you a calming and relaxing experience, where you can immerse yourself in a visually stunning environment and let your mind wander. In Subterra, you'll explore vast underground caverns, winding tunnels, and dark caves, all while encountering friendly creatures and collecting treasures along the way. You'll solve quick puzzles, overcome challenges, and uncover secrets as you navigate through the various levels of the game.

10.0 Reports and Products:

The final iteration of this project will be a full-fledged game that has multiple levels and a degree of intuitive game design to keep player retention. The product should be fun and engaging and fully operational while being bug-free; the goal of this product is to demonstrate well-implemented game elements working in unison. The product is meant to work on all Windows machines and should be a simple executable program that the user double-clicks to launch.

11.0 Schedule:

This is the ideal schedule for Spring of 2023 at CSUF in order to start and complete this project.

2023	January		February				March				April				May				Summary	
Tasks:	1	2	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	Hours:	Percentage:
Research	8	8	6																22	12%
Design		2	10	10	6	8													36	20%
Development					4	5	8	10	10										37	20%
Testing									2	5	8	5	5						25	14%
Modification										2	5	8	5	5					25	14%
Final Report													2	10	5	5			22	12%
Demonstration																	8	8	16	9%
Hours:	8	10	16	10	10	13	8	10	12	7	13	13	12	15	5	5	8	8	183	100%

Figure 3: Projected Schedule

12.0 UML Diagrams

12.1 Use Case Diagram:

The case diagram illustrates the user interactions with the various menus in the game, as well as the actual gameplay the player can have. The user will be able to change aspects of different features through a settings menu, be able to traverse elements through the main menu and pause menu elements, and finally be able to interact with the gameworld elements.

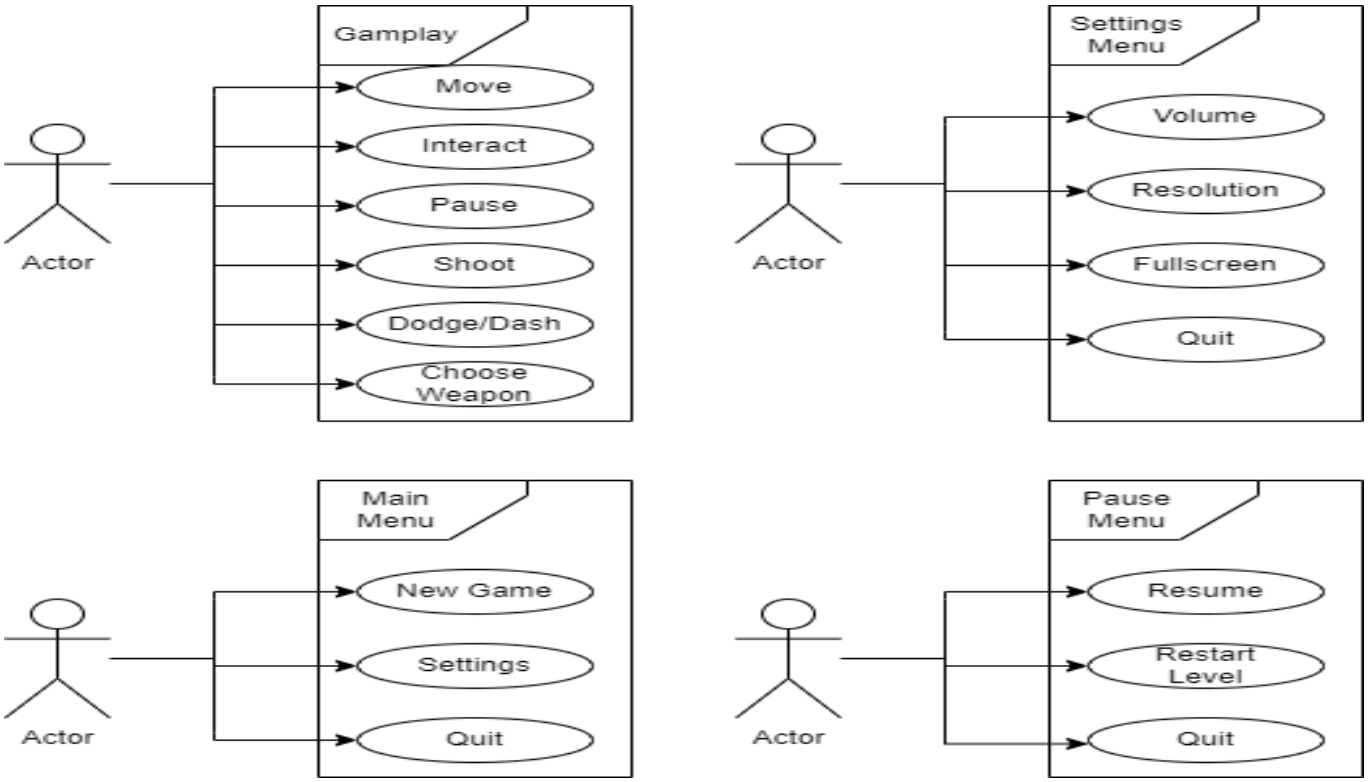

Figure 4: Use Case Diagram

12.2 Sequence Diagram:

This sequence diagram conveys the flow of the user's options through the game's system. The user shall start off by launching the game and seeing the game's UI that displays a menu for user settings. It then transitions into one of the game's levels chosen by the user with the additional option to pause the game to quit or resume.

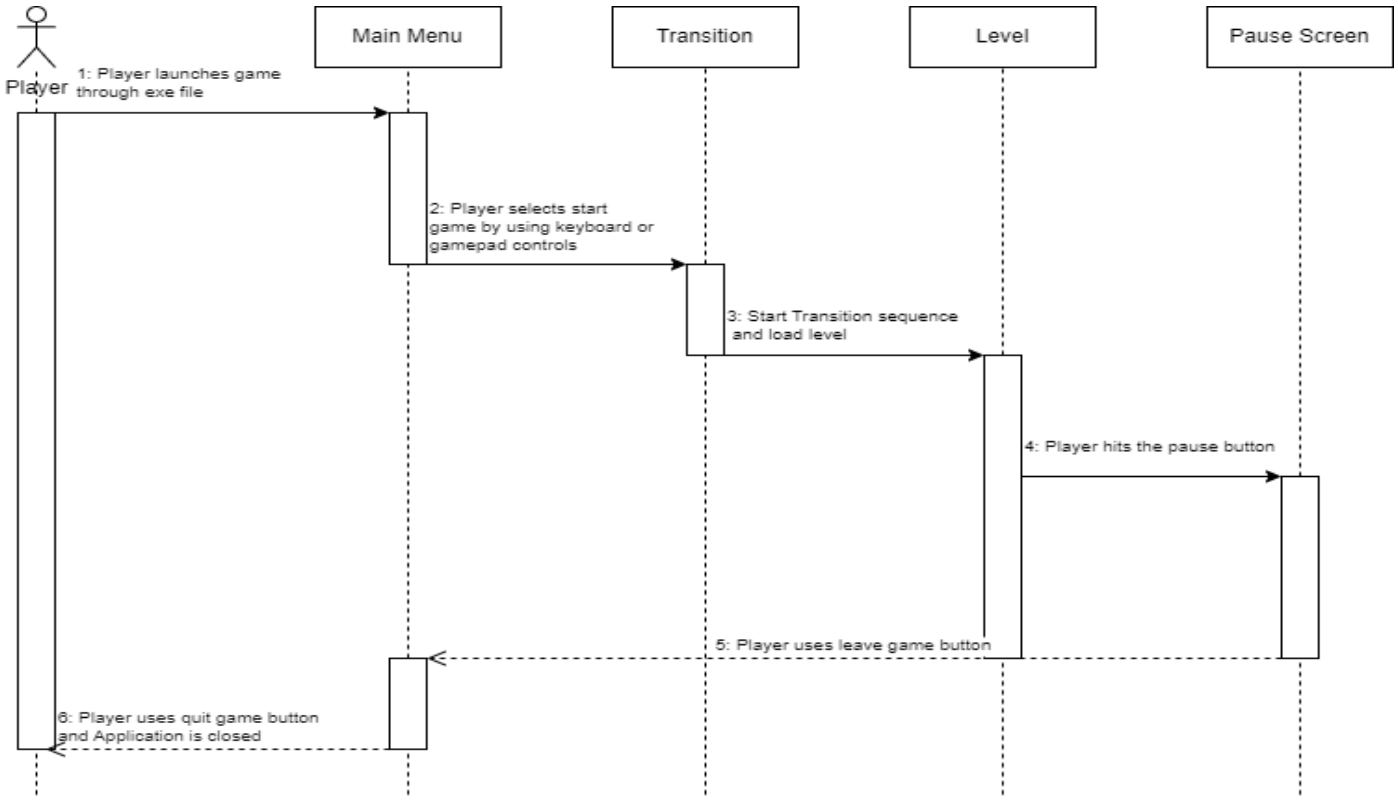

Figure 5: Sequence Diagram

12.3 Activity Diagram:

The activity diagram displays the flow between actions during the game's launch and ending execution. It shows the parallel or concurrent flows and alternate flows that a user comes by through the game. Starting from the main menu options we go into two options between starting the game and settings. Settings will allow users to modify their gameplay and then put them back into the main menu. Starting the game will load the level the user chose with the current settings saved.

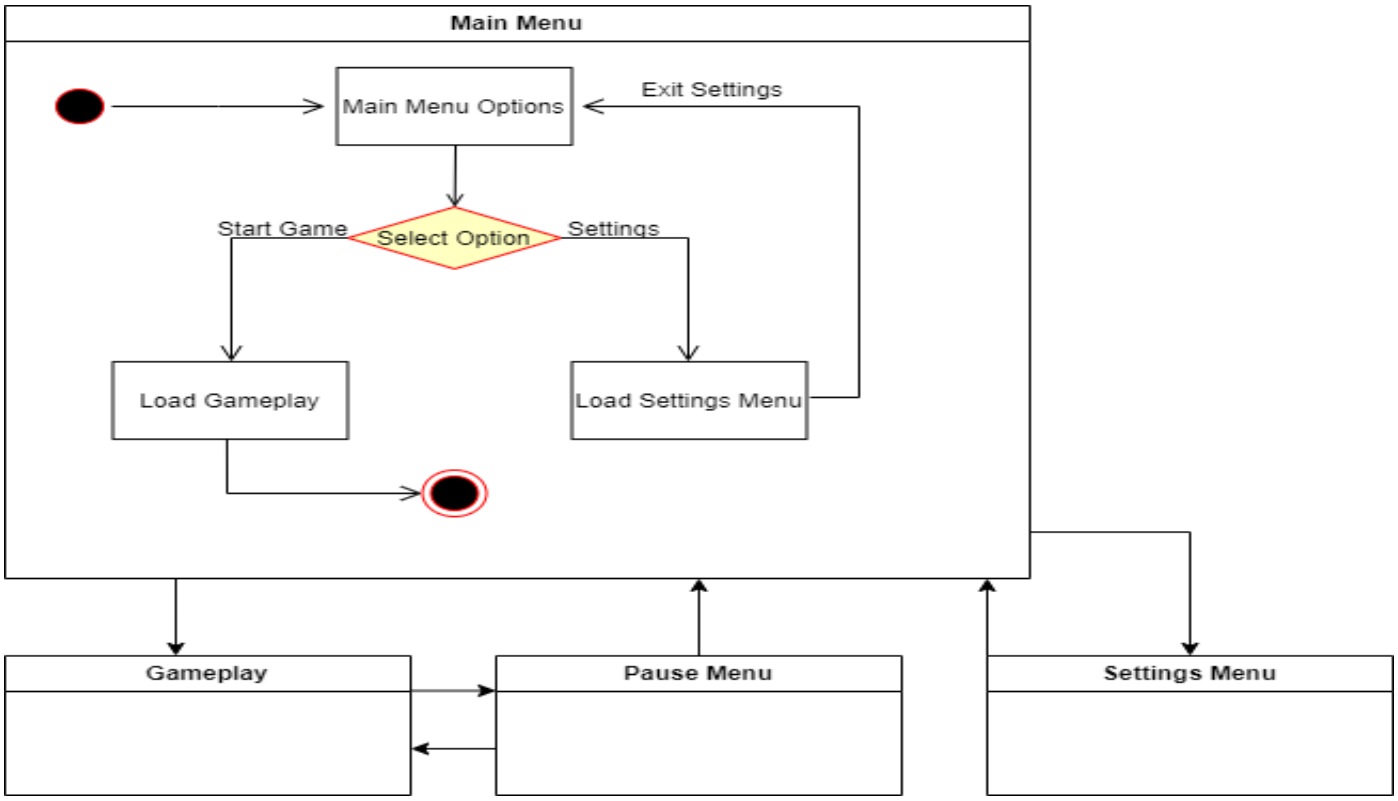

Figure 6: Activity Diagram

12.4 Class Diagram:

The class diagram breaks down the interactions between various elements in the game design's code. It shows how they depend on each other and the hierarchy of their command within the code. In this representation, the main menu, game, and setting elements are all communicating with one another, ultimately resulting in the game function operating the rest of the program. The player, enemies, music, and other elements will be dependent on the game executing.

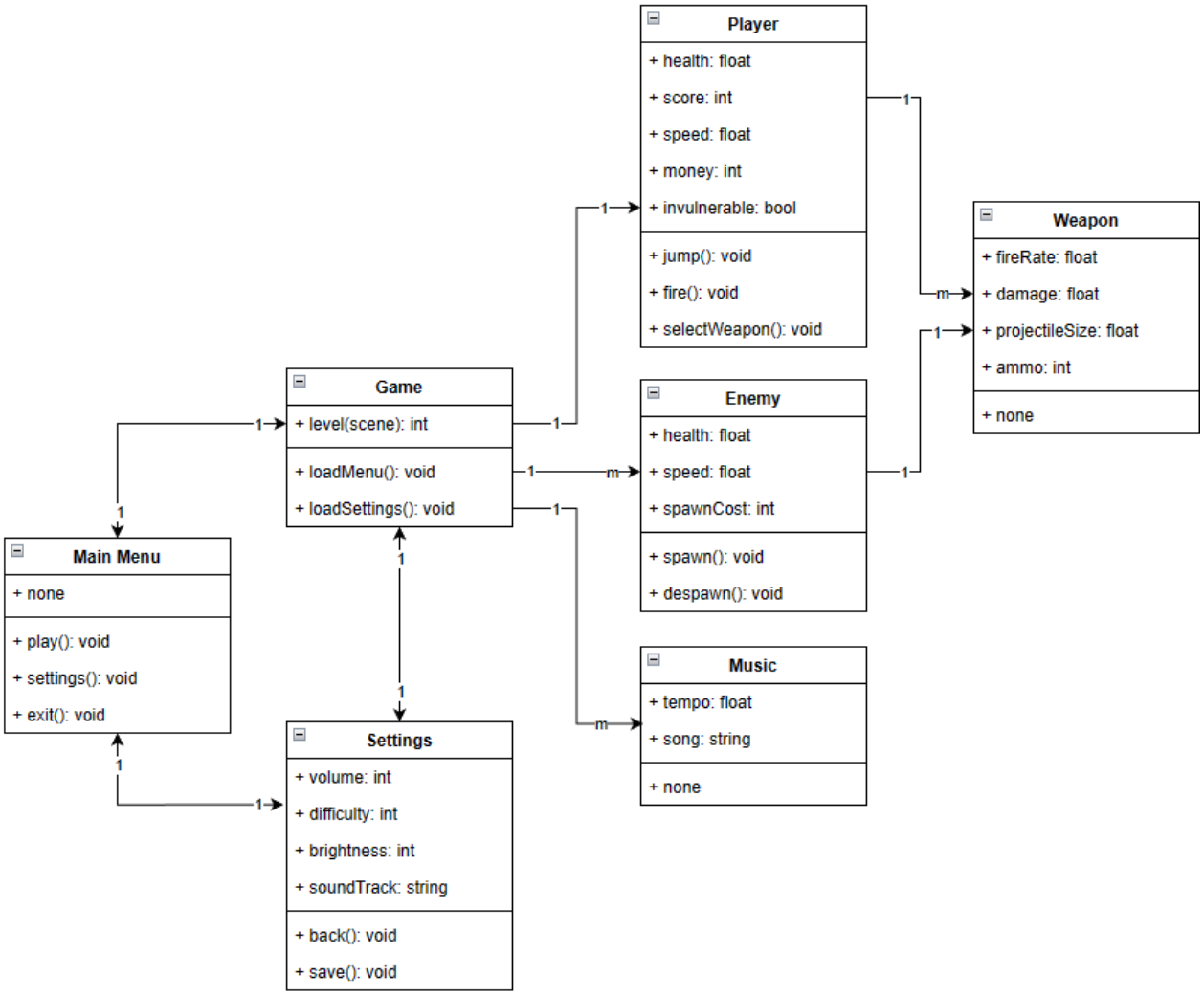

Figure 7: Class Diagram

13.0 Implementation:

13.1 Main Menu Page:

Below is the main menu displaying the user's options to either start a new game, load an old game, change the user's settings in options, or exit the program which saves all user data before exiting out. A new game will launch a fresh start to a user's data and will save as the game continues. The options menu will allow users to look at their controls.

Figure 8: Main Menu Page

13.2 Load Game:

The load option will allow users to go back to past saved games and continue where they left off. It will show the health count of the player, time amount played, total coins collected and a delete option to remove a saved data file.

Figure 9: Load Page

13.3 Options:

Displays a settings menu where users are able to change the brightness, resolution of the game, a fullscreen option, quality level, and buttons to either apply or reset the settings to default. There will also be a volume setting to change the sound level of the game and a controls menu to change the controls based on the user's preference.

Figure 10: Game Settings Page pt. 1

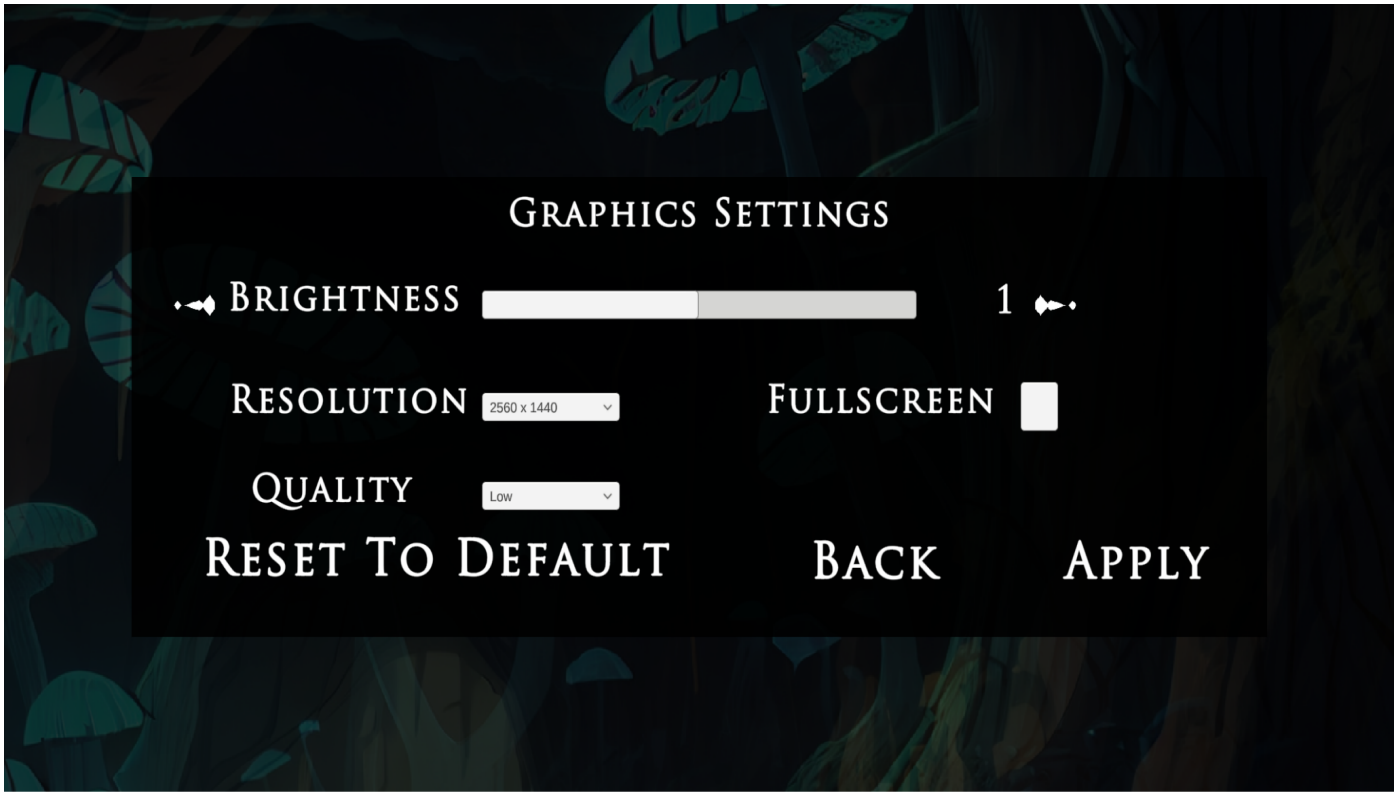

Figure 11: Game Settings Page pt. 2

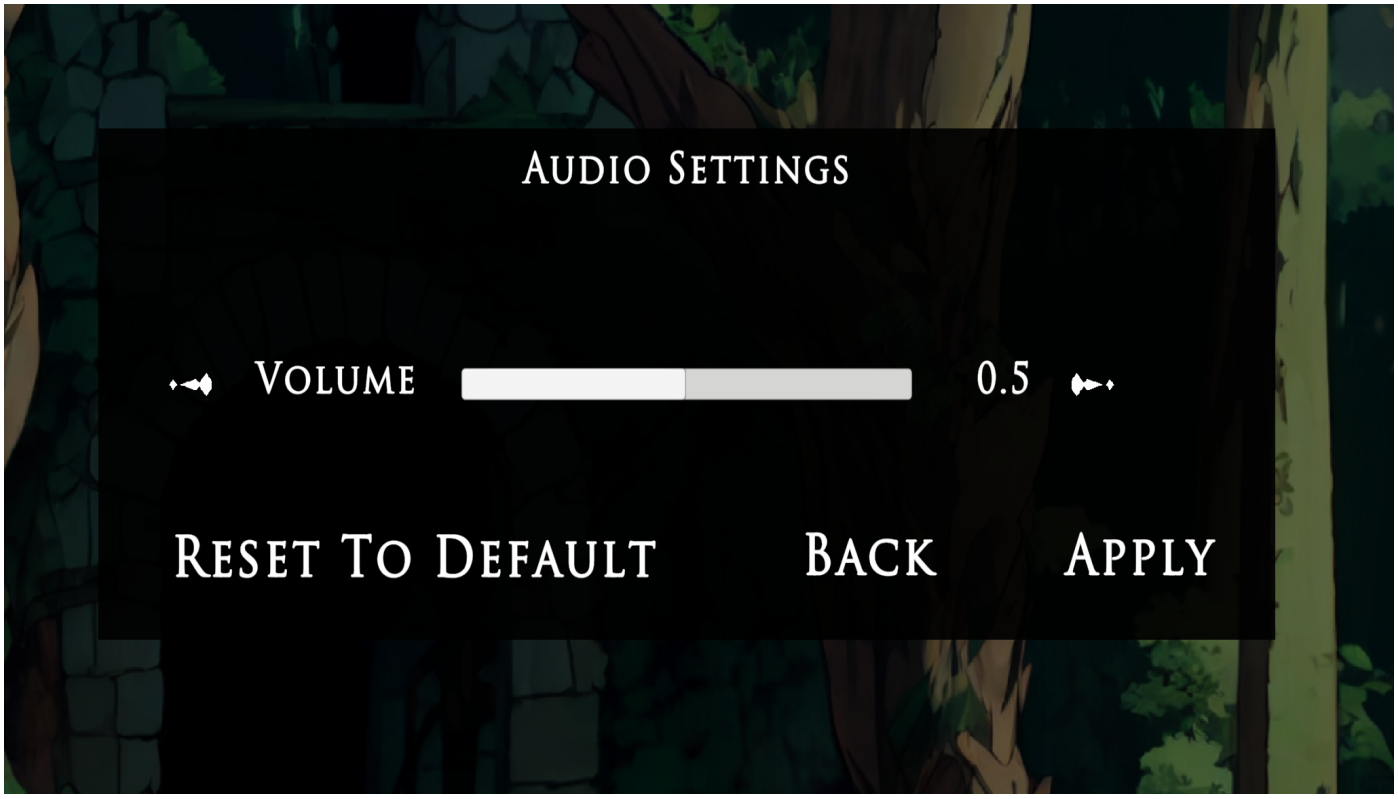

Figure 12: Game Settings Page pt. 3

Figure 13: Game Settings Page pt. 4

13.4 In-Game:

The player model is represented with custom art, surrounded in a custom built world with custom assets. There is a UI that displays the player's current health at all times, as well as another section that displays the current amount of coins the player currently holds.

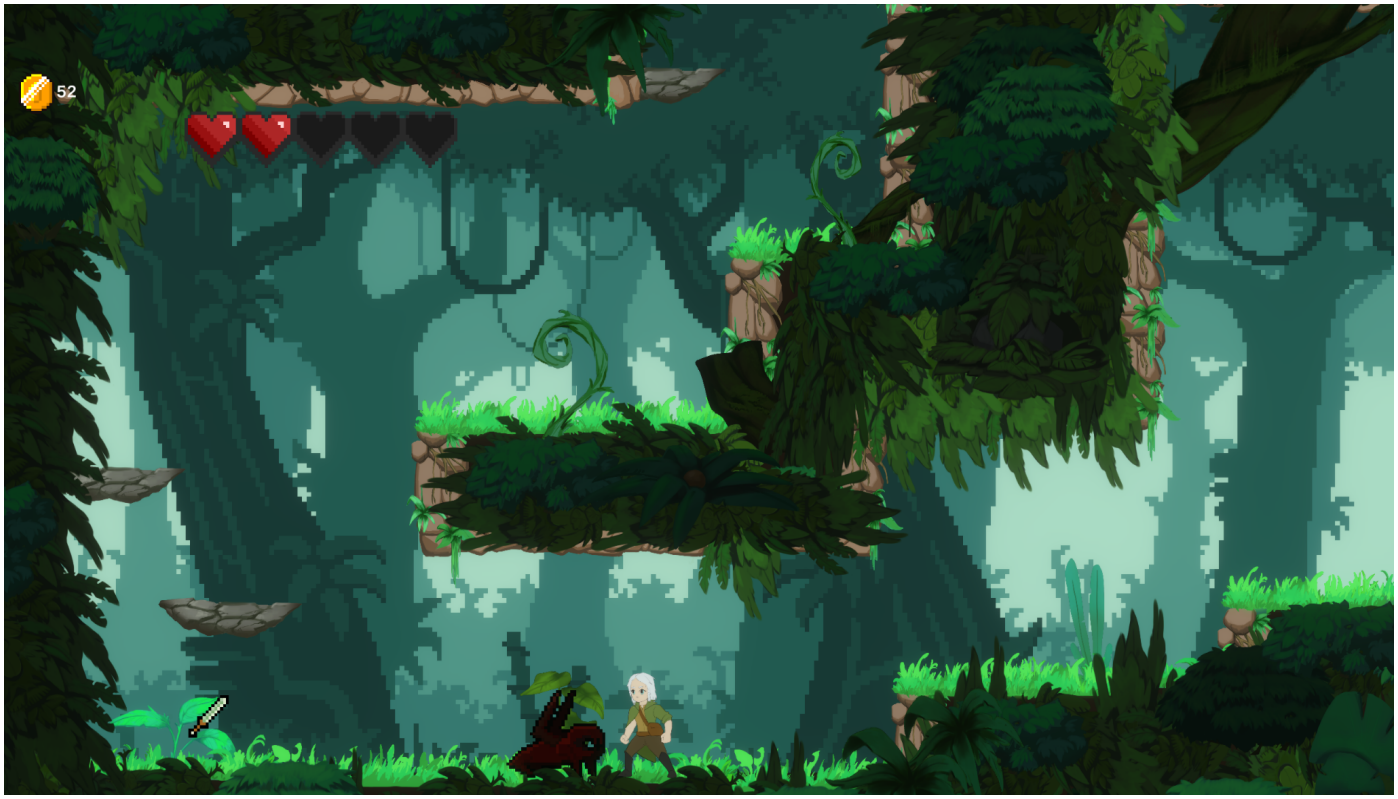

Figure 14: Gameplay Scene

13.5 Inventory:

This UI element will store the player's pickups in a personal inventory. Players may equip different weapons or use consumables from their inventory. Items are handled with a base custom data struct. Items are then classified depending on their usage. When right-clicked or selected through button input, a menu with specific actions for that item is opened up.

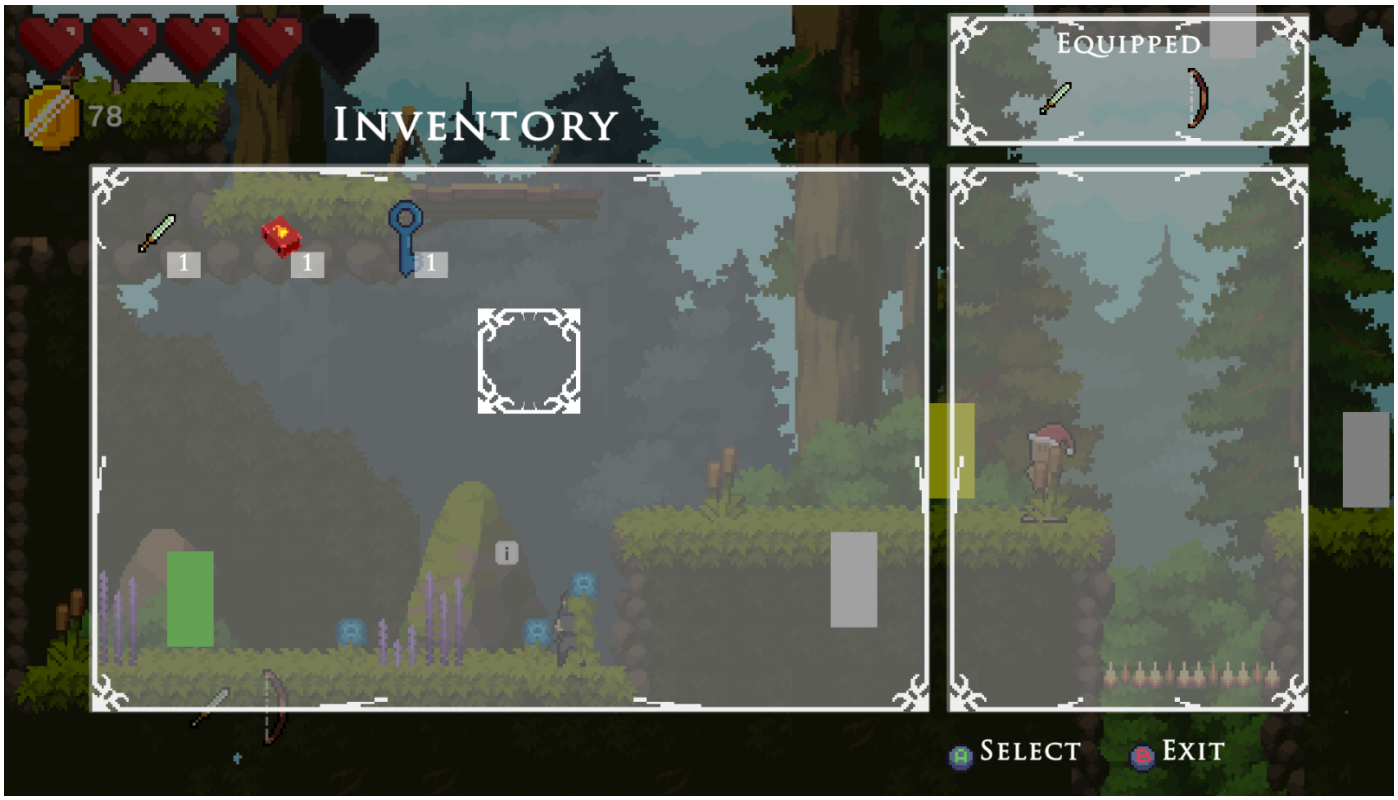

Figure 15: Gameplay Inventory

13.6 Shop System:

The player is now able to purchase items from a nearby NPC. Pressing the Interact Button opens up the shop UI and items will be added to the player's inventory if they have the necessary funds.

Figure 16: Gameplay Shopkeeper

13.7 Pause Menu:

Player is now able to bring up a pause menu that will redirect the player to quit or return to the main menu. While this pause menu is active, the game is paused in the foreground and the mouse becomes focused on the transparent window.

Figure 17: In-Game Pause Menu

13.8 Dialogue System:

The game now incorporates a dialogue system using the Inky Text Package. NPCs are sent an Inky Text File which will parse certain strings to play certain dialogue animations and audio. This will allow for multiple characters to be played using the same script.

The Dialogue System in Subterra is an integral part of the game that allows players to interact with the characters and creatures they encounter during their underground adventures. The system is designed to be intuitive, easy to use, and engaging so that players can immerse themselves in the story and the world of the game.

Figure 18: In-Game Dialogue System

13.9 World Design:

There was great importance placed on the design of the levels before implementing them into Unity to avoid confusion with character spacing, map changes, and weapon and skill unlocks. Below were some of the iterations that lead to a final design. Map environments include a Forest biome, an Underground biome, and a Temple biome.

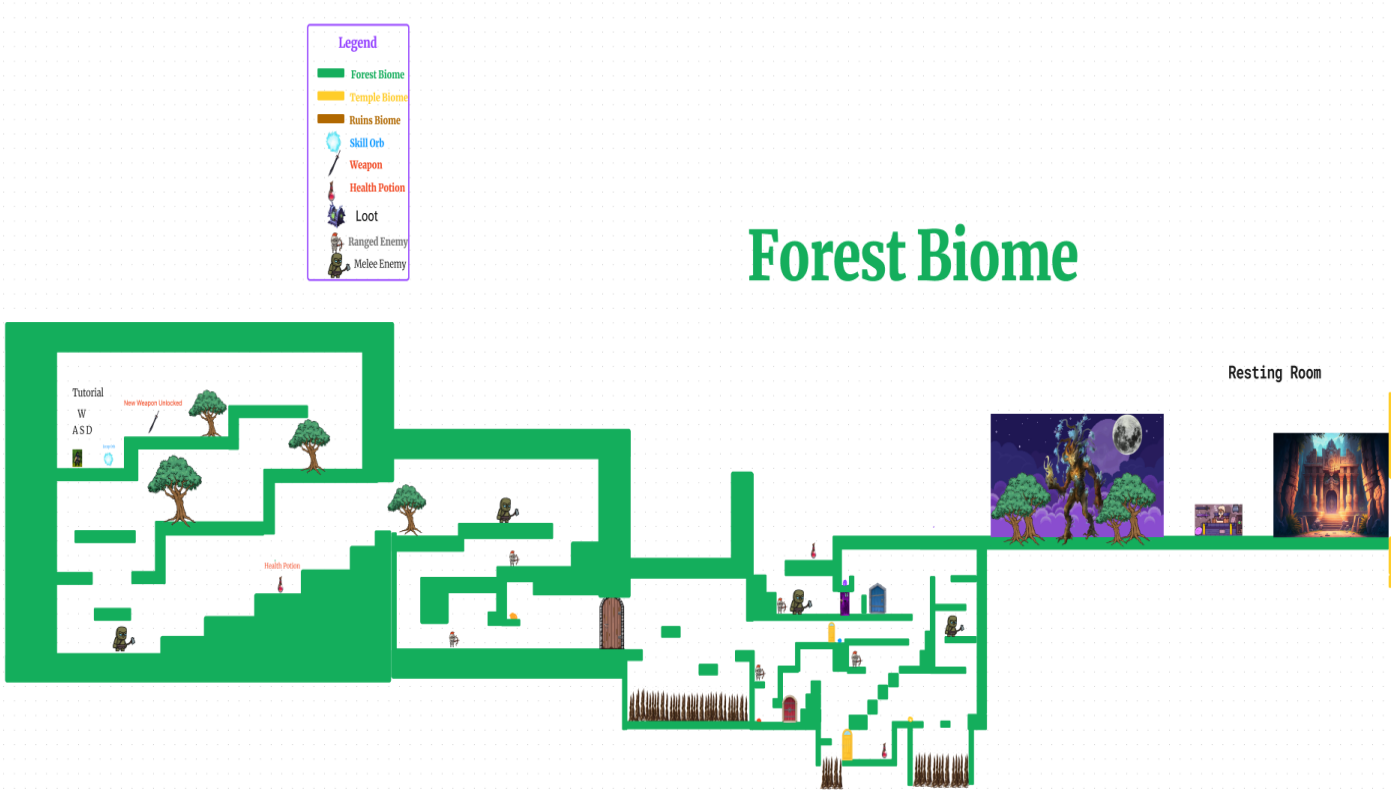

Figure 19: Forest Biome Concept

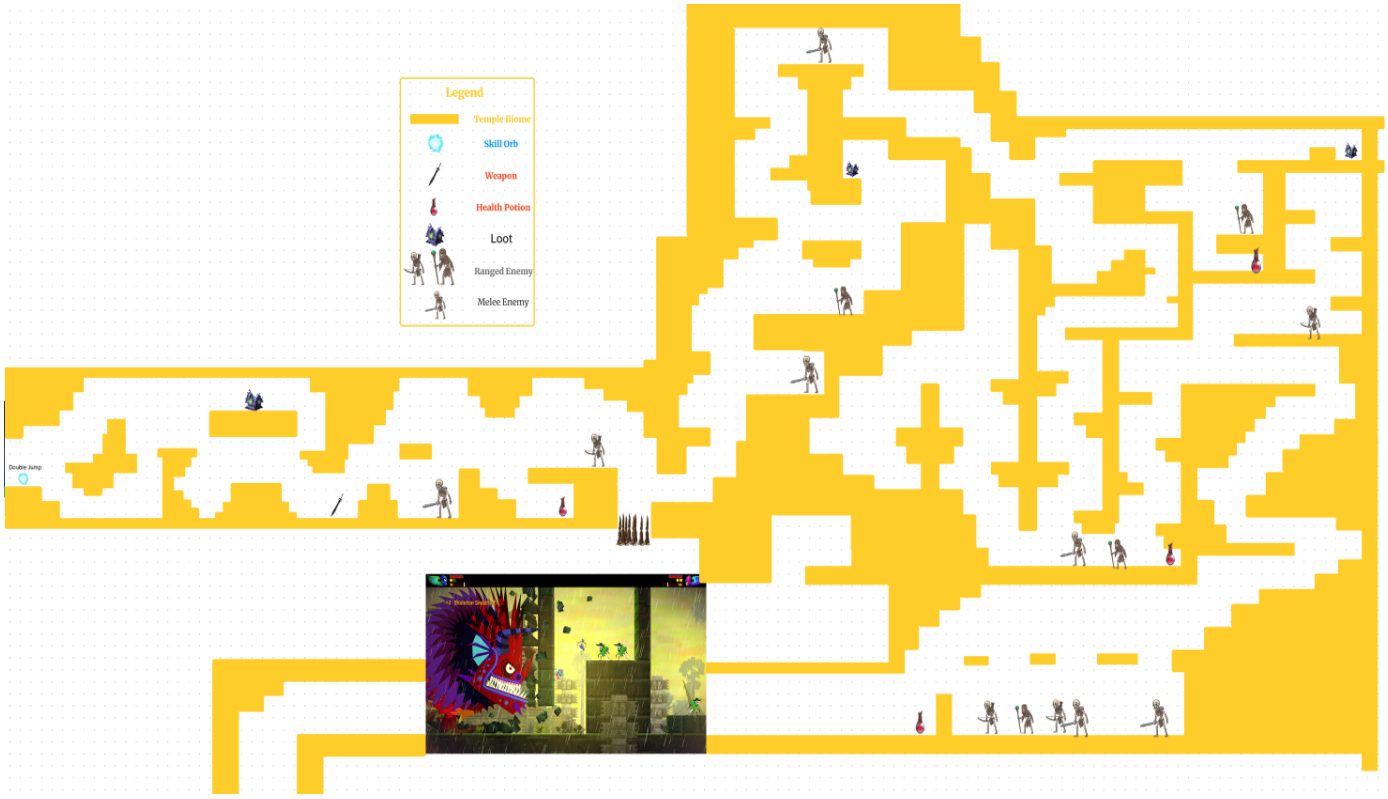

Figure 20: Temple Biome Concept

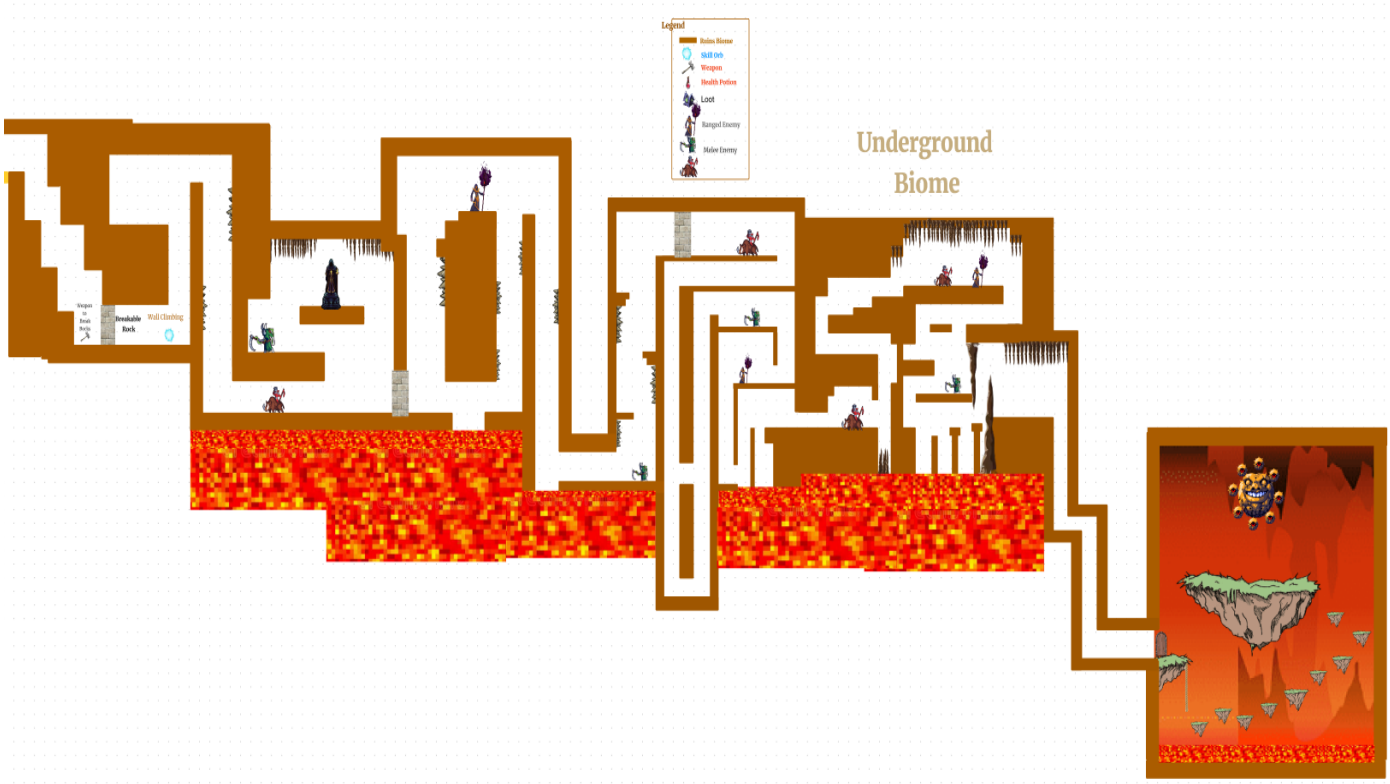

Figure 21:Underground Biome Temple

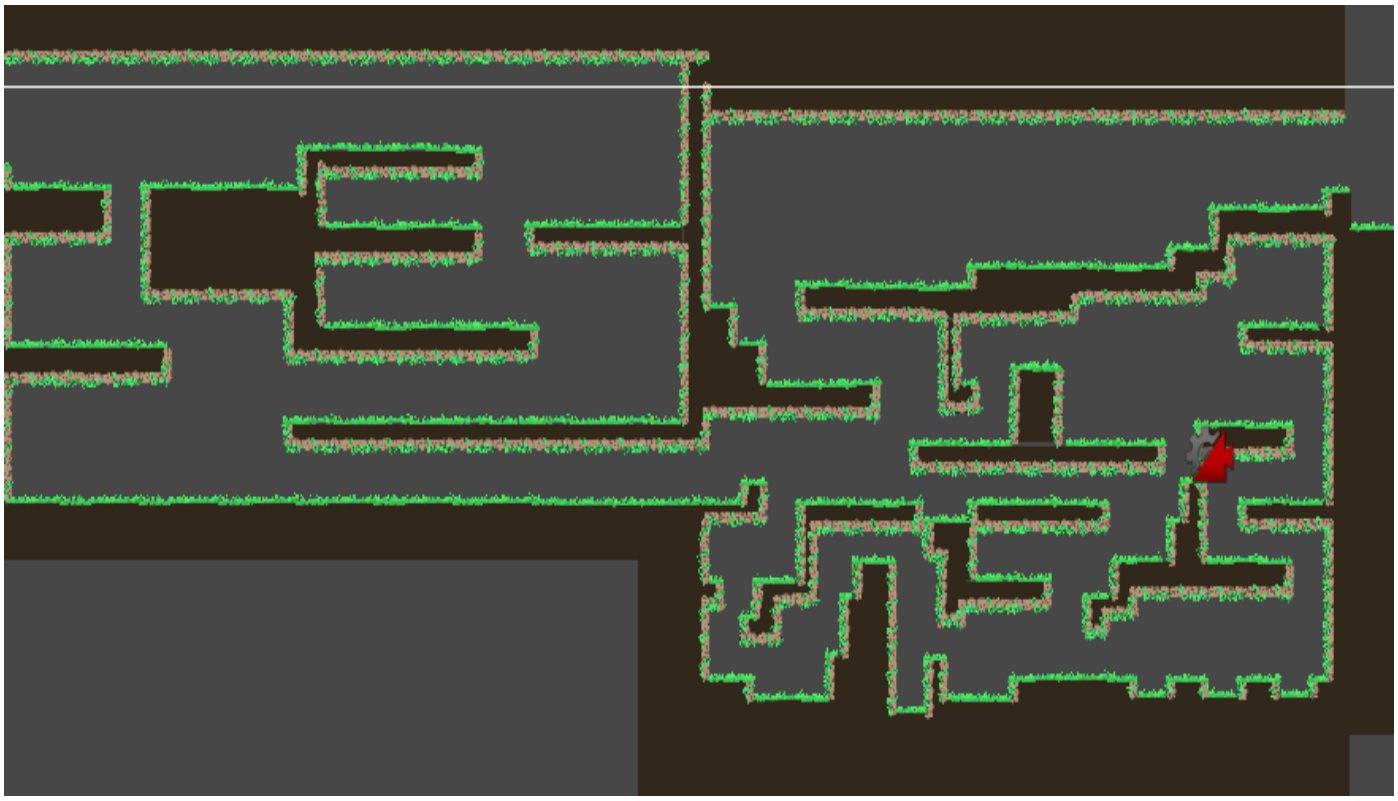

Figure 22: Map Iteration in Complexity

14.0 Installation Instructions:

Subterra will be hosted on Itch.io with a game page that gives a short description of the game and some screenshots to show off aspects of the game. The user will download the game files from the host site, unzip the package, and double click the executable file. The executable file will start up the software, landing the user at the splash screen where they can start playing.

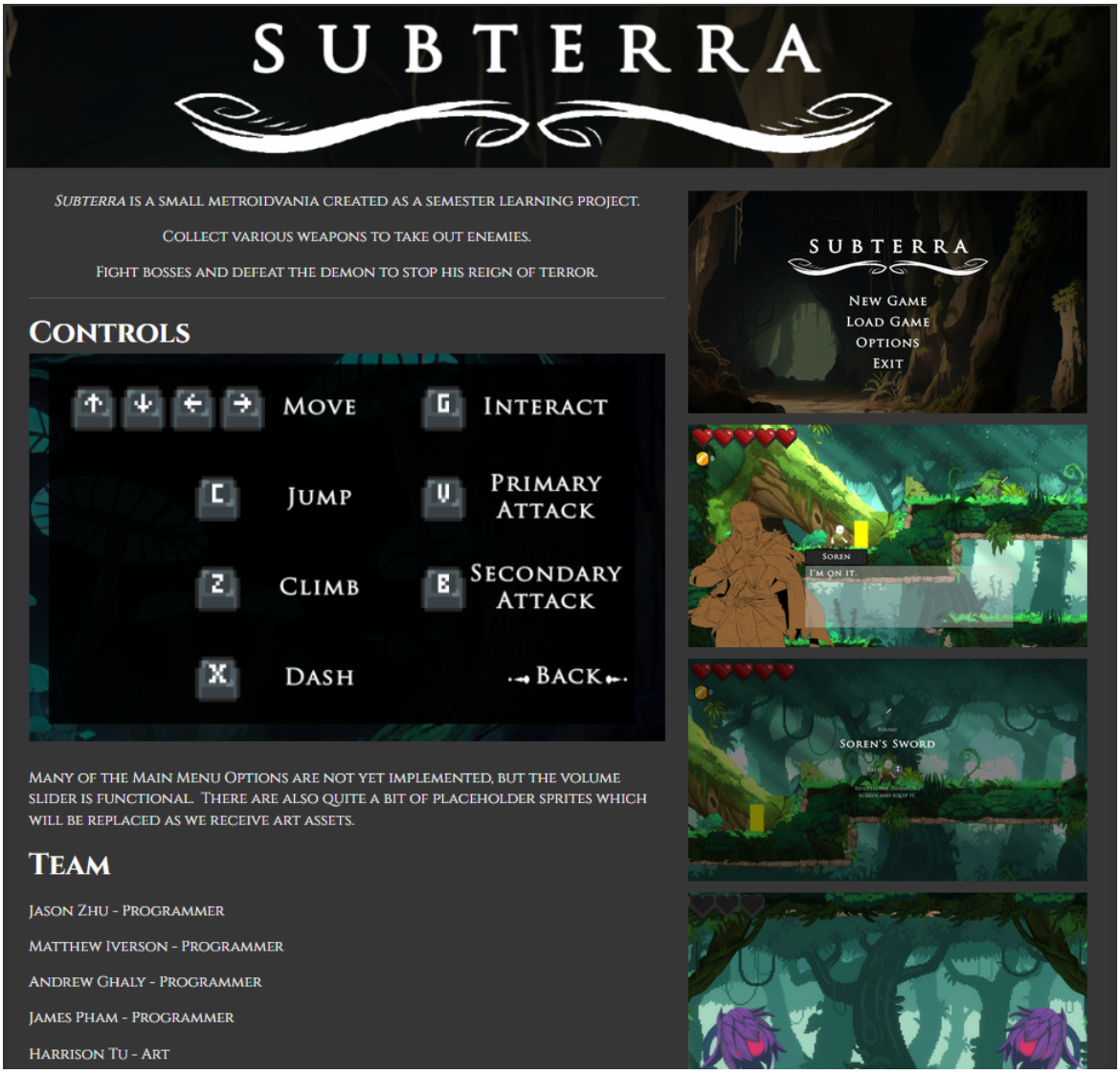

Figure 23: [Itchi.io Page]

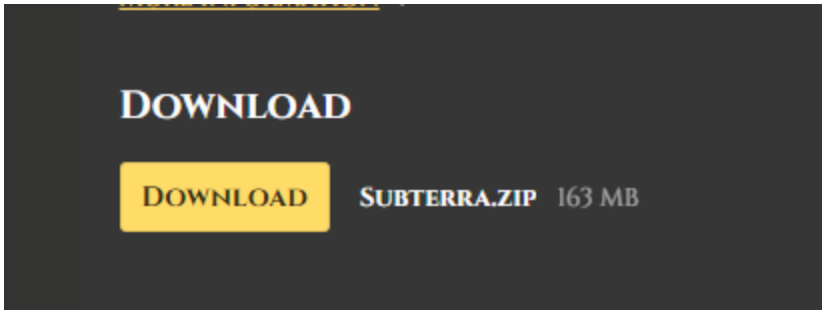

Figure 24: Itchi.io Download Button

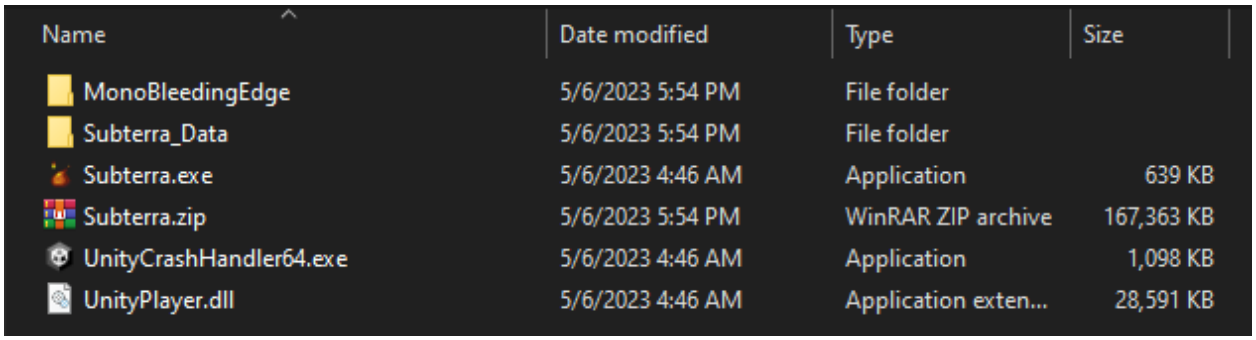

Figure 25: Unzipped Package with Executable

15.0 Recommendations for Enhancement:

Future improvements will include more add-ons to the game including items, enemies, map levels, and quality improvement. Cleaning up the integration of the design would be great as well for less clutter. Custom settings for user's controls and gameplay difficulty would be implemented as well. Due to time constraints, styling and perfecting each level of design would be at a satisfactory level.

Expanding the game's world: Subterra's world is already quite vast, but adding new areas, hidden secrets, and paths to explore would provide players with even more to do. Increase the variety of enemies and bosses: While Subterra's enemies are diverse, there is always room for more types of foes. Adding new bosses with unique mechanics and attack patterns could also add to the game's overall challenge. Add more power-ups and abilities: As with any Metroidvania game, Subterra's power-ups and abilities are key to exploration and progression. Adjusting former upgrades and abilities, such as wall-running or dashing, would enhance the player's sense of discovery and accomplishment.

16.0 References:

- Moll, Philipp, et al. “How Players Play Games: Observing the Influences of Game Mechanics.” *MMVE '20: Proceedings of the 12th ACM International Workshop on Immersive Mixed and Virtual Environment Systems*, 8 June 2020, pp. 7–12. *ACM Digital Library*, doi-org.lib-proxy.fullerton.edu/10.1145/3386293.3397113.

- Toups, Zo, et al. “Making Maps Available for Play: Analyzing the Design of Game Cartography Interfaces.” *ACM Transactions on Computer-Human Interaction*, vol. 26, no. 5, 19 July 2019, pp. 1–43. *ACM Digital Library*, doi-org.lib-proxy.fullerton.edu/10.1145/3336144.

- I. Sagredo-Olivenza, P. P. Gómez-Martín, M. A. Gómez-Martín and P. A. González-Calero, "Trained Behavior Trees: Programming by Demonstration to Support AI Game Designers," in IEEE Transactions on Games, vol. 11, no. 1, pp. 5-14, March 2019, doi-org.lib-proxy.fullerton.edu/10.1109/TG.2017.2771831.