Game Design Document (Exam Game)

Ricardo Costa-Tré (2351979), Andrew Kapp (2329108), Ashley Jurisich (1946550)

October 19, 2022

Chapter 1

Introduction

Forbidden Labyrinth (FL) is a rogue-lite game, drawing aspects and inspiration from games such as *Hades* and *Monster Hunter*.

The game aims to focus on the use of a combat system through a series of dungeons, where the player will kill enemies, loot them and create better equipment, with the focus on getting stronger to face stronger enemies.

The main purpose of choosing this topic was for the investigation of the creation and design process of the mechanics and aspects of games that the members of the group were familiar with and enjoyed.

Chapter 2

Game Inspirations

As mentioned previously, games that provided inspiration for this project were games such as *Hades* and *Monster Hunter*. Below describes the aspects of enjoyment from the two games, followed by a list of other games that members enjoyed, highlighting aspects and mechanics that the group was interested in.

2.1 *Hades*

Hades is a rogue-lite game that is held dear by all members of the group for its character design, move-to-move gameplay as well as the overarching gameplay loop.

The gameplay loop in *Hades* includes attempting to escape a seemingly random layout of a dungeon. The dungeon is generated on a room to room basis with the challenges, enemies and rewards randomly chosen upon entry.

Hades will serve as the inspiration for the dungeon generation design and part of the gameplay loop. A randomly generated dungeon will be built for the player to attempt to make as much progress as possible encountering enemies of increasing difficulty the longer and deeper they play.

The idea of having a 'central hub' to return to after each dungeon attempt is in the cards to store other external mechanics and for the player to manage resources and their inventory.

2.2 *Monster Hunter*

Monster Hunter is a franchise about hunting for resources. Resources are integral parts of acquiring stronger or more suitable gear for the task ahead. This leads to a fair amount of grind as resource drops are randomized completely or influenced by

accomplishments during the hunt.

Monster Hunter is a game of mastery and gear progression since there are no ways of improving the characters base stats.

The inspiration from *Monster Hunter* will influence the player's mindset towards entering the dungeon while in the dungeon and to keep the player going into it. The player will need the resources dropped from the enemies inside the dungeon to improve their weapons and gear. Drops will be determined solely by the chance of the item dropping from the killed enemy.

2.3 Other Media

- Cyberpunk, Skyrim
  - First Person
  - RPG
  - Crafting
  - Inventory Design
- GORN, Beatsaber
  - Possible VR Gameplay
  - Hand Motions
- Solo Leveling
  - Dungeon Crawling
  - Loot acquired from defeated enemies
  - Unknown dungeon layout

Chapter 3

Pre-Alpha Prototype

3.1 Systemic Overview

3.1.1 Procedural Dungeon

Ash is working on a procedural generated dungeon that changes its layout each time the player enters the dungeon in an attempt to keep gameplay fresh and stimulating for the player.

The Dungeon will reform itself entirely randomly in a fashion similar to the stories of Daedalus' Labyrinth from Greek Mythology. The Labyrinth was an inconceivably massive maze that shifted continuously.

3.1.2 Enemy AI

Enemy AI is to be modeled to attempt to counter the player, keeping gameplay dynamic and challenging to the player.

The dungeon will collect data from the player's exploits, collecting data from enemies and the player to create a network of sorts that the dungeon remembers and acts on. This is also implemented to make the player think about using new approaches and weapon-types to exploit the dungeon's new weaknesses and counter its strengths.

3.1.3 Crafting

Crafting is a way to incentivise players entering the dungeon multiple times in the hopes of acquiring the materials they need for upgrades or a new weapon. A need for resources makes a need to play.

Crafting in *Monster Hunter* is the sole purpose for hunting specific monsters over and over again. FL will aim to do the same, incentivizing players to enter the dungeon

in the hopes of collecting the rarer materials to receive the more sought after gear.

Chapter 4

Alpha Prototype

4.1 Pre-Alpha Results

4.1.1 Playtesting

Due to a lack of a comfortable time frame in which to sit with a playtester and interrogate how they interact with the prototype and ask them questions about the system; playtesting was minimal for this prototype. However the small amount of testing that was achieved showed that base mouse sensitivity was far too high, the existence of an attack combo was not obvious to the player and that the attack animations felt sluggish and uninteresting due to the small amount of player feedback.

Since there was a small amount of testing that actually came back, the only thing that could be improved upon at this time was the mouse sensitivity being too high. The tester said they became easily disoriented since a small movement of their mouse translated to the player turning around multiple times, thus the sensitivity was drastically reduced.

4.1.2 Functionality

The plan for this iteration quickly became making a framework that will theoretically support the use of multiple types of offhand abilities. This means creating a pathway through which the equipped offhand dictates the ability that is currently available for use.

This being said, the ability must be attached to the object and be able to communicate to the animator and the player's input. Thus, the offhand holds a reference to the animation name and handles all component handling on the off hand itself such as collider handling and effects.

4.1.3 Combat System

The Player combat system contains a simple 3 hit combo and the ability to block incoming damage. The current system makes the player take damage if enemies get too close to the player. This is a simple solution to demonstrate and test player health and communication systems.

In later iterations the method for the player to take damage will be called externally and thus place the act of taking damage in control of the enemies rather than the player just taking damage for the sake of it.

The enemy combat system contains a basic state machine that will allow for further states to be implemented with ease. Currently, the enemy AI has two different states that include a chase state, and an attack state. Further plans have been made to implement further enemy states,such as that as a dodge state.

4.1.4 Inventory and Crafting

A basic crafting system is still in development and can unfortunately not be previewed by the "Pre-Alpha". While the system is near completion, it is not in a working state that allows for its use and functionality to be properly displayed as of yet.

The inventory system currently implemented allows for the player to view their inventory, pickup items, as well as having a UI that shows the system updating the inventory.

The system is also currently set up to allow for easy expansion of resources and actually are in a state where they are working well with the crafting system currently in development. Most of the time on this system was focused on creating a modular system that allowed for easy expansion and easy integration with other systems.

4.1.5 Enemy AI

Changes to the original idea of the dynamic AI have been opted for. Originally, the idea was to have an AI that acted and learnt on the spot, manipulating movements accordingly, however this system was overly complex and advanced.

As such, the use of an enemy state machine was used to control the basic states of the enemy, such as chasing the player, and attacking the player. With this change, the use of the neural network has been transferred to that of a dynamic difficulty system.

4.1.6 Dynamic Difficulty AI

The idea for this dynamic difficulty system is for the game to adapt and change enemies after every complete/failed run and change attributes of the enemies to either strengthen or weaken them. Ideas currently involve things such as stronger attacks, quicker attack speeds, more health, etc.

Currently this system is still in very early stages of development and experimentation, hence unable to appear in this prototype. Since this idea is also very theoretical, it may come to fruition that this idea may have to be scrapped entirely, however that is not the current goal.

4.2 System Changes and Developments

Unfortunately, due to lack of planning, and an inadequate amount of work done required to make this project feasible, a large amount of the game has been reworked. This is prevalent in all aspects, including areas such as game mechanics, game genre and the overall intended experience of our game.

As such, the following will show the new design process used, visualizing systems, core-mechanics as well as outlining the new intent and direction for the game.

4.3 Game Re-Evaluation

After evaluating the created systems currently implemented, a list was drawn up of aspects that we had currently finished building the foundations for, half finished systems, as well as systems that had yet to been started – described below – as per the date of the 'Pre-Alpha'.

4.3.1 Procedurally Generated Dungeon

The first appearance of the procedurally generated dungeon had yet to take place and one of the core systems on the 'Dungeon Crawling' aspect had yet to be visualized. While updates from Ash mentioned progress to the system, there was no clear indication as to how complete the system really was, a result of lack of any physical evidence and activity of the repository. This later stemmed to an intervention which will be covered in a later section.

4.4 Enemy AI

During the development process of the 'Pre-Alpha' build, the complexity of creating a full neural network for each enemy proved extremely difficult and implausible. Adjustments where made such that the neural network would be responsible for the dynamic difficulty aspect of the dungeon, adjusting enemy stats and variables to dynamically change the difficulty. This however also proved to be implausible due to the huge time sync just to learn neural networks. While research and experimentation took place regarding neural networks, the very distinct lack of progress in terms of time available become very apparent, leaving a result of nowhere near adequate development.

As for adjustments regarding the enemy behaviour, the decision to go with a finite state machine was taken. This approach was opted for as it allowed the enemies to have basic states in which they could interact with the player with relative ease, while still being modular enough that more states could be added with minimal complexity and recoding needed.

4.4.1 Crafting and Inventory

The basic functionality of the inventory system was complete. The player could interact with items that were dropped from enemies, successfully being able to store them in an inventory. This inventory could also be displayed at anytime during the game, while still being able to move around and play the game (akin to Souls-like games).

The crafting system was around mid-completion. Crafting recipes where made, were able to craft and resources were able to be used. Incomplete features included that of a interactable crafting bench, as well as having the items purely being pseudo-crafted, and not taking actual effect in the game (such as having it added to an inventory and being able to use them).

4.4.2 Combat System

The combat system was easily the most developed system currently in the prototype. It consisted of a basic 3-hit combo in which a player could attack enemies, dealing damage and potentially killing them. A blocking system was also included to allow the mitigation of some of the incoming damage from enemies - while the player had their shield activated. The system also involved the use of animations to time attacks (creating triggers during certain frames of the animation), as well as using colliders to detect enemies.

4.4.3 Virtual Reality

Absolutely no progress. Definitely not enough time to even consider it given the huge number of outstanding systems as well as undeveloped systems.

4.5 Mechanic and System Culling

Due to the huge lack of progress and huge split in resources on many mechanics, a necessary culling of many sub-system, and reworking of core mechanics was needed.

Due to the actual lack of proper planning and just "going with the flow", a clear vision of goals and expected outcome of the systems was very grey, leaving many in an incomplete state.

As such, a huge reworking and more in-depth broken of mechanics was necessary, allowing for an easier evaluation of bare-minimum systems that had to be in place.

This led to the removal of the following system and mechanics:

1. *Dynamic AI Difficulty using Neural Networks*
   Undertaking neural networks in such a short span of time, especially since this required the acquisition and learning of the knowledge of how to create and use one from scratch, did not fit the allocated time correctly.

2. *Crafting System*
   A larger system that originally was going to server as the progression system for the game, allowing for the player to create new equipment that could be used, allowing for different playstyles, was scrapped in favour for a different progression system that will be discussed in detail later. This form of progression as a system was more complex than originally thought out to be, and required the need to develop many different variations in progression, such as weapon damages, weapon styles, etc. Once again, due to time constraints, it proved implausible to develop this within the allocated time.

4.6 Mechanic and System Reworking

Taking note of previous failures, specifically the lack of specification to systems and what they entail, was considered as to create a better and more fleshed out game, aiming to create a game with vision and a clear sense of progress, outcome and development. This in turn resulted in a reworking of the project management, which

can be found, detailed in the Project Management Document.

4.6.1 Combat System

Serving as the direct and core mechanic in which the player interacts with, he world, and in turn plays the game, the combat system was a large focus on the future development and cornerstone of our future builds.

Combo System

In an attempt to minimalize the excess clutter as well as need to create many different weapons and to work in tandem with other mechanics, the choice of player having access to one weapon was opted for, allowing for a deeper focus and experience in the combat as a whole. This focuses on the polishing of one system – focusing on balancing system and making it feel "good" and working – to ideally create an experience in which the player learns mastery over said weapon. This ties in with the progression system, which will be later discussed with the shop system.

The focusing on one weapon, allowed for the further expansion on the combat mechanics themselves, allowing for introduction of a combo system. The combo system focuses on the use of learning patterns in which the player can execute their attacks to counter enemy actions and survive.

The combo system can be described and visualized in the binary tree below, dictating the different state of the combo as well as their ending points.

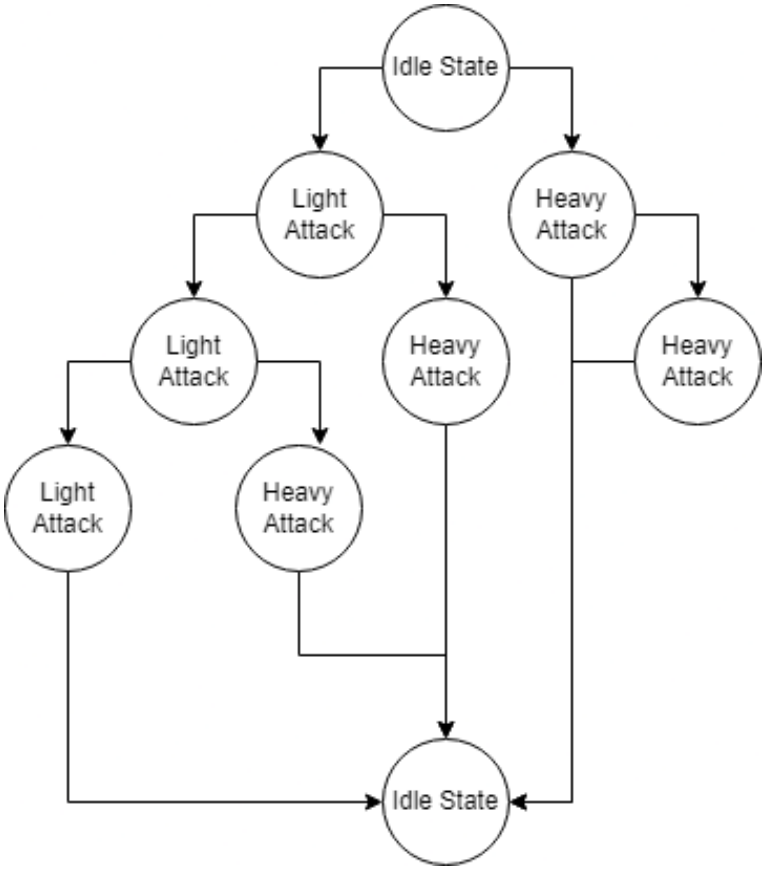

Figure 4.1: Binary tree showing combinations of attack combos

To create a combo system that had a meaningful impact on the player experience and to prevent button mashing (in an attempt to create a more skill-based feel), the player is punished by having their attacks lock after excess spamming of attacks. This aims to work in tandem with the stunning effect, encouraging the player to explore these combos and build strategies around them.

The aim of the combo system is not only having the player punished due to unthought out attacks, but also aims to reward them with the exploration of said combos. This idea is implemented using combo damage multipliers, that are affected by the order in which players choose to attack. The aim of this is to encourage the player to explore the different combos, getting a feel for the different damages, attack speeds and play styles the player can choose during combat.

Stunning Effect

The stunning effect is a mechanics that, as mentioned previously, aims to work in tandem with the combo system. The general concept is that when an entity is hit (both player and enemy), they will be temporarily stunned and unable to attack, while also interrupting the combo. The duration of the stun will be reliant on whether the

attack was light or heavy, hence adding another layer of depth to which the player is to consider attacking the enemy.

Combat System and Character-Player Interaction

Given the feedback on the 'Pre-Alpha', it came to light that the player has very limited options in terms of how they are able to interact with the game due to one single combo, dodge, block, and move being their only gameplay controls.

With this feedback, it was made a priority to give the player variety in their attack patterns as well as giving the prototype enemies some variety in the same space - albeit not to the same extent as of the 'Alpha' build. The main incentive to perform these different combos is to allow the player to output more damage against enemies by combining light and heavy attacks in a series of combination attacks.

With the added UI elements that show the layer the controls available to them at all times, the player should not feel so lost in playing the game.

Through internal playtesting, a concern was found in how the enemies transition between states and how the animations they perform are not totally honest in conveying what state the enemies are in. To combat this enemies now have an animation to act as a midpoint between walking and attacking, this creates a more fluid transition from 'AttackState' to 'ChaseState'.

Enemies are also locked in place when in the 'AttackState', this eliminates the occurrence of enemies floating toward the player while still in an attack animation. As said, enemies received an attack combo overhaul. Enemies, upon entering the 'AttackState', will generate a number that will determine the length of their next combo to a maximum of 3 attacks per combo, these attacks do a flat rate of damage on every hit.

4.6.2 Procedural Generation

The procedural generation is currently setup such in away that the player spawns in a predefined spawn area. Once the player spawns here, the rest of the map within a certain radius is generated, such that the player can wonder freely without being constrained.

The idea of this methos is to create a dungeon layout like that of the Labyrinth mentioned within the Greek Myths, having a constant changing state. This is implemented by the means of the dungeon despawning areas furthest away from the player (once they move to area) and replacing it with newly generated labyrinth passages,

such that the spawn radius and dungeon generated remains constant.

Each passage on the labyrinth then aims to have a chance of having an enemy spawn in that room when the room is generated. This chance will have to be playtested to see which feels the most natural and may even result in only certain rooms being able to spawn enemies.

A shop area is also randomly spawned at a lower chance. This ties in with the shop system - which will be detailed later – and serves as a place where the player can either upgrade their weapons or heal a fraction of their health.

There is also an even smaller chance of the labyrinth spawning an arena portal. This portal will act as a gateway to the final stage of the game. Here the player will encounter a tougher boss-like enemy, in which they need to defeat. Exploration is then further presented to the player to explore this room, in aims they realise that this is the final area of the game. Since this aims to be the finale, in which the player will; either have to restart or complete the game, the initial idea is for the player to stumble upon it early at least once in their runs, and not knowing what they are up against. Through here they will learn knowledge about the game and its endpoint, while still having some hope of replayability, as this aims to be a learning curve as to whether they are well equipped enough for the final fight or not.

Below is a flowchart that visualises the basic workings of the procedural generation and how it aims to interact with other systems.

The design process was laid out by Ash as such:

Basic physical layout
The map will be made up of separate blocks in a grid pattern. Each standard block will have either zero, one or two walls, with the blocks with two walls having them be either adjacent or opposite each other. The first special block is the shop, which a player must enter in order to access the shop. The second special block is the arena entrance, which will take the player to the final arena once they enter the block.

Basic internal structure The blocks will be stored in a 2D array. The player will be located on the block at the center of the array. The array will store all the blocks within a 5-block radius. Each cell in the array will hold a Block structure. The block structure is made up of the block type, the rotation of the block, the x and z coordinates of the block in the world space, and four Boolean values for the state of each of the four sides, i.e., whether the side is a wall or if it is open.

Initial map The player starts off in an open 3x3 area, which is the starting point every time the game is played. From there, the array is populated (and the map generated) by creating blocks to the front of the start area, then to the left and right, and finally behind the starting area. This initial generation will use the same formula

as the rest of the map generation, but is slightly altered. There will also be no shop and arena blocks in this initial map.

System overview When the player moves into a new block, a new row/column must be added to one side of the array, and one from the opposite side must be removed. Each row/column is shifted by 1 to create the open elements for the new row/column, and the last row is deleted. For each element in the open space, a new block must be added. For each new block, the possible block types for that space must be determined and one will be selected. The rotation and location for the new block will be calculated. This new block will then be added to the array and spawned on the map.

Detailed block generation Once the array has been rearranged, there will be one empty row/column for the new blocks (there will still be blocks in the array, but since they will be overwritten it is effectively empty).

The first block is determined slightly differently to the rest of the blocks. The position of the adjoining block in the previous row/column is retrieved to calculate the position of the new block. Depending on whether it is a new row or column, either the x or z coordinate will be used for all the new blocks, while the other will be calculated by adding onto the previous block's location.

For the first block, the state of the adjoining side of the adjoining block is found, i.e., whether there is a wall or not, and the state for one of the adjacent sides is randomly chosen. Using the states of these two sides, the possible blocks that can be placed there are found, and one is randomly chosen with a rotation that matches the state of each of the sides. The state of those two sides is recorded, as well as the state of the other two sides.

For the rest of the blocks the same method is used, except instead of randomly deciding a state for one wall it is determined by the previous block generated.

Once the new block has all of its elements, it is added to the array and the physical block is added to the world space.

Below are flow charts and a diagram of the process:

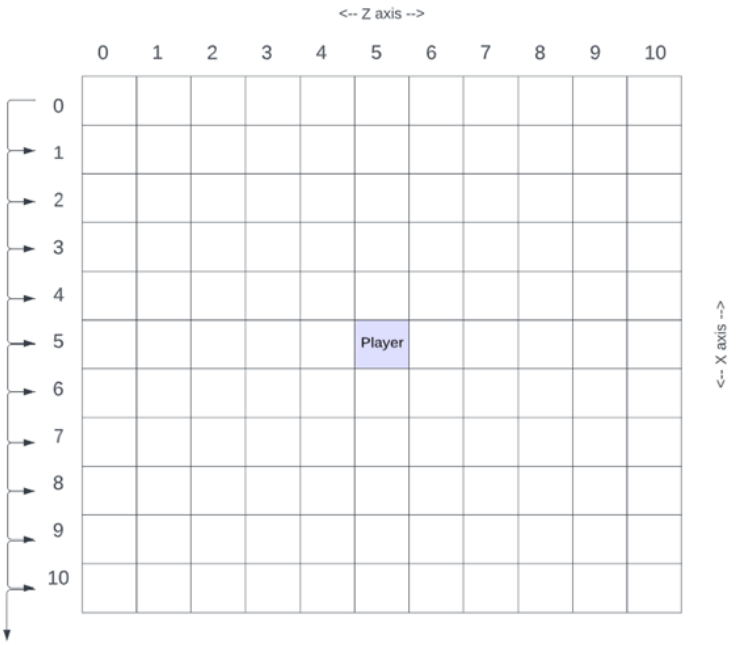

Figure 4.2: Layout of the array with arrows showing the movement of the rows when a new row is added at the top

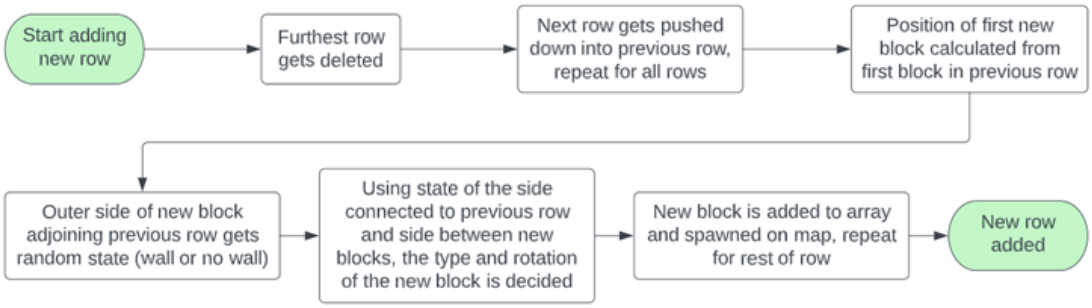

Figure 4.3: Flow diagram of the process to add a new row to the map

The shop unlock function was not working as planned. After changing a function from fixed update to update, the problem seemed to have been solved. The shop has not been fully implemented yet however.

A separate character had to be created for my testing. The proper character requires input from a controller, while I only have access to a keyboard and mouse. This was a minor problem.

Shop System

Following the scrapping of the crafting system, a more simplistic approach was taken towards the progression system. This consists of the aforementioned arena, as well as its link to the shop system.

The shop system aims to be a place in which players are able to better equip themselves for the dungeon. Once encountering the shop, the player has the option to browse the wares for weapon upgrades and even the chance for a small healing potion, but of course at a cost.

This ties into the resource and inventory system that was previously created to server as a currency for these small pit-stops in which the player can better prepare themselves.

Currently, the aim is to create three main upgrades, being weapon damage upgrades, weapon attack upgrades and heals for the player (as there will be no other way for the player to heal in a run).

This aims to encourage the player to explore the labyrinth more to find more enemies, gain loot and be able to afford the items in the shop.

This system acts as the progression system as it allows the player to better their equipment as well as top of their health to be prepared to fight the final boss in the arena, in aims to complete the game.

Below consists of table or currently predefined values of what the items in the shop costs. These values will most likely be tweaked in accordance with feedback give from play-testing sessions.

Item Name	Item Effect	Gold Cost	Crystal Cost
Whetstone of Sharpness	+15 Weapon Damage	350	5
Whetstone of Swiftness	+0.5 Weapon Attack Speed	1200	10
Gourd of Immortality	+50 Player Health	10000	30

Table 4.1: Table Showing Information Regarding Shop Items Currently Implemented

4.6.3 Visual Feedback

Building on feedback give from the 'Pre-Alpha', it became apparent that players were unaware and not made aware through the game functionality of when they hit enemy, where their weapon swung and other game feel issues.

This aims to be resolved with the use of visual indicators which provide visual feedback to the players as to what is happening.

The first indicator consists of the simple crosshair. While it was previously guessed that this would add another layer of depth to the game in terms of difficulty, it rather became clear that players were unable to understand where they were aiming and made more of a frustrating experience than that of a challenge.

Another was to add weapon trails for the different types of attacks. This was a much-needed visual indicator that should prove to benefit how the player interprets the combat system, as well as making them more aware as to what they are doing. This consists of the basic colour variations of the weapon attack such that the player can differentiate between light and heavy attacks. This also aims to further allow for players to understand and master the combo system, which is very reliant on the order of light and heavy attacks. The trails give the player some understanding of where the attack is taking place, the trails last long enough that the path followed by the active part of the weapon is clear to the player, however a lack of anchoring pbjects such as arms connected to the weapon still leave room for misinformation and disorientation.

The last major visual indicator was that of damage popups. This allows players to be able to visualize as well as be able to formulate a pattern as to which combos will be needed to kill an enemy. This was also added in the hopes that players will experiment with different combos, finding a method that suits their playstyle more while being able to visualize and compare damage outputs.

A smaller visual indicator is that of the button mapping. This serves as a basic tool to help players learn the basic functionality of the character and how to use them, due to the lack of a current tutorial. The aim for this indicator is to provide the basic input mappings such that the player can understand how to control the player, while eluding the clear combo patterns, which aims to make the player learn through experimentation of different attacks and how they are chained.

4.7 Development Going Forward

This section serves as a breakdown as to what is completed by the ‘Alpha’ Prototype, as well as highlighting core issues that need to be resolved and system that need to be worked on for the ‘Beta’ release.

Current 'Alpha' Build State

Currently, the 'Alpha' does not consist of one build, but rather exits in three separate parts. This was due to a cause in a proper merging of integration of the procedural generation with the rest of the current product.

The first part of the prototype consists of a worked upon version of the 'Pre-Alpha'. This build focuses on the player and enemy interaction from the previous build while also having included new aspects. These new aspects include the use of visual feedback systems. As of this build, all the visual feedback systems were implemented as mentioned in the scope for the build.

The second part of the prototype consists of the procedurally generated dungeon. This dungeon follows the scope as planned to contain a Greek Mythological feel of the Labyrinth. This scene shows the basic interworking's of the procedural generation and how the player will navigate through it. It also contains a placeholder enemy, for the testing of the enemy navigation within the procedural generated world.

The third part of the prototype consists of the shop system. This system is near completion, however, is limited towards integration and is hard to visualize actual changes to the game, as this would have to be tied in with the shop activation in the procedural generated shop. It consists of the ability to randomly generate a store with items. It also involved the working checks to see if the player can buy a selected item, as well as being able to apply the affect of the bought item on purchase. Currently however, the actual navigation regarding the shop is in development and as such is hard to fully demonstrate within a build.

'Alpha' Build Deviations and Challenges

The main reason for the lack of integration of the combat system and the procedural generation focuses mainly on the original problem of the enemy's navigation among the procedurally generated map. Since the enemy is using a NavMesh component to navigate and locate the player, a NavMesh was required to be baked. However, it came to realization in the attempt to merge these two systems that NavMesh could not be baked onto a prefab to allow for full map generated surfaces.

While possible to generate a mesh on the surface, it would only be confined to the block, providing seams to the NavMesh that do not interconnect with each other, making it impossible for enemies to move from block to block. As such, this provided an unforeseen problem that required more research as well as a full day to amend with a suitable solution that settled for computer performance with the desired outcome and behaviour of enemies.

This left less time to work on the completion of the shop system, which stems to

one of the main reasons as to why the navigation regarding the shop was not completed by the specified date.

Plans for the 'Beta' Build

Currently, the build progress for the game has been dramatically improved since the shift to a more formal project management structure. Information and tasks are easier to outline as well as assign, adding to a more productive team and productive development towards the game.

Plans for the 'Beta' include the full integration of the systems in a seamless way with no bugs. This involved the merging of the three separate builds without errors and should allow for a working, full-game prototype to take place.

Involved after the integration is the completion of outstanding systems. This includes navigation of menus and shops, as well as the integration of enemy spawning with the procedural generated world. This will also require the integration of the shop system to the trigger event of the shop tiles in the world.

Following that, the navigation of the player to the arena is required as well as the creation, modifying and changes to a tougher boss-like enemy needs to be put into effect. This should server as the final of the core structures for the base of the game.

Following the full completion of the base game, playtesting is to be carried out such that feedback can be given to work on the adjustments, fine-tuning and balancing of the game. This also serves as an opportunity to identify lacking systems and ways to improve them, allowing it to be modified accordingly for the 'Beta' release.

Chapter 5

IMPORTANT NOTICE

Please note that this game is built for controller and not having one connected will most likely not allow the player to move.