This Report is submitted in partial fulfilment of the requirements for the BSc Honours Computing (Games Programming) Degree at Edge Hill University.

Callum Mclaughlin

Supervisor: peter matthew

3rd Person Dungeon Game

RESEARCH AND DEVELOPMENT PROJECT | SUBMITTED 19/05/2022

2022

# Abstract

The main objective of this project was to create a third person dungeon crawling game with a focus on the AI of the enemies. A variety of techniques and methods were adopted to create the final submission, level design skills, working with animations, triggering events using triggers and of course lots of coding.

The main method for the AI, which was the focus of this project, was making use of the Navigation Mesh, and having the NPCs (Non-Player Characters) patrol the area of the Nav Mesh until they were within the range of the player and would then chase and attack them.

The game is designed to give the player a fun experience with good AI design but also challenge them in the form of the AI enemy skeleton and the dragon boss. After completing most of the main objective of the game the results were good.

The AI for the skeleton and the boss works well in terms of tracking the player and following them when the player is within their detection range. The results for attacks from the AI are varied as there were a few problems with it that will be discussed later in the report.

# Acknowledgements

I would like to extend my deepest appreciation to Peter Matthew who helped me massively throughout the duration of the project. Peter was on hand whenever I needed him to help fix errors in my code and give advice on the direction of development for each system in the game.

1. Contents

[i. Abstract 1](#_Toc103798727)

[ii. Acknowledgements 1](#_Toc103798728)

[Chapter 1 - Introduction 4](#_Toc103798729)

[Chapter 2 - Background and Literature Review 6](#_Toc103798730)

[Chapter 3 - Problem Analysis & System Requirements 9](#_Toc103798731)

[Functional Requirements 11](#_Toc103798732)

[Non-Functional Requirements 15](#_Toc103798733)

[Chapter 4 – Design 16](#_Toc103798734)

[Chapter 5 – Implementation 18](#_Toc103798735)

[Level Design 18](#_Toc103798736)

[Player Movement & Third Person View 18](#_Toc103798737)

[Player Attack 18](#_Toc103798738)

[Character Animations 19](#_Toc103798739)

[Player Health Bar 19](#_Toc103798740)

[Enemy/Boss Movement and Navigation Mesh 20](#_Toc103798741)

[Enemy & Boss Attack 21](#_Toc103798742)

[Hit Detection 22](#_Toc103798743)

[Boss Room Trigger Event 23](#_Toc103798744)

[Ambient Music /& Sound Effects 23](#_Toc103798745)

[Lighting 24](#_Toc103798746)

[Chapter 6 - Testing and Known Problems 25](#_Toc103798747)

[Testing Plan 25](#_Toc103798748)

[Evaluation of Developmental Success 30](#_Toc103798749)

[Known Problems 31](#_Toc103798750)

[Chapter 7 – Conclusion 32](#_Toc103798751)

[Chapter 8 - Critical Evaluation 33](#_Toc103798752)

[Further Work 34](#_Toc103798753)

[Bibliography 35](#_Toc103798754)

[Appendices 37](#_Toc103798755)

[Appendix One – Player Model 37](#_Toc103798756)

[Appendix Two – Skeleton Model 37](#_Toc103798757)

[Appendix Three – Dragon Model 38](#_Toc103798758)

[Appendix Four – Level Design Sketch 38](#_Toc103798759)

[Appendix Five – Script Design 39](#_Toc103798760)

[Appendix Six – Nav Mesh and AI Pathing Design 39](#_Toc103798761)

[Appendix Seven – State Machine Diagram 40](#_Toc103798762)

[Appendix Eight – Project Plan 40](#_Toc103798763)

[Appendix Nine – Survey Data One 41](#_Toc103798764)

[Appendix Ten – Survey Data Two 41](#_Toc103798765)

[Appendix Eleven – Survey Data Three 42](#_Toc103798766)

[Appendix Twelve – Survey Data Four 42](#_Toc103798767)

[Appendix Thirteen – Survey Data Five 43](#_Toc103798768)

# Chapter 1 - Introduction

The idea for the project came from a game called Final Fantasy XIV which is an MMO-RPG created by Square Enix (Square Enix, 2022). Within Final Fantasy there are a multitude of things to do and one of the most popular and most active is fighting through dungeons with other players. You can also fight through the dungeons using something called the “Trust System” which was a feature introduced in the Shadowbringers Expansion. This system allows you to take on dungeons from the Main Scenario Questline with a party of allied NPCS (Non-Player Characters) (ConsoleGamesWiki, 2022).

The main aim of this project was to create well designed and well performing AI on a smaller scale than in Final Fantasy. The idea of having the player fight through a dungeon and fighting enemies that have been coded to fight them and have an allied NPC helping you fight them too seemed like it would be a challenge to implement but also very interesting and fun.

There was a vast amount of research completed before the project’s development had started and this researched yielded a lot of helpful material which helped get the project development off to a solid start. Most of the research was in the field of AI development for Unity and using state machines to control how the enemies will behave.

The first previous work that was found on this topic was a tutorial on YouTube in which the creator of the video, GDTitans, teaches the viewer the methods in which you can create State Machine Behaviours in unity using the animator controller and conditions on the transitions between states. The main method of how state machine behaviours is managed is by adding scripts to the states within the animation controller. This tutorial was very helpful for understanding the animation controller further and how it can be utilized to control the AI.

However, this was not the method that was used in the project as another method was discovered that seemed more manageable and efficient when working on such a large project. This method involved coding scripts which would then be attached to the respective character and would control their movement, attacks, animations and for the skeleton enemy it would also control its patrolling path. This method will be discussed in further detail later in the report.

Other methods using in this project include box colliders for detecting when the characters have been hit by another and to deal damage to the characters as well as play the hit sound clips when the trigger is activated. Nav Mesh agents were used for the skeleton enemy in order to setup a patrolling path which it walks around when it is not chasing the player.

Sliders were used on top of images to create the health bars for the player and the boss which change colour the lower the players health goes to 0, starting from green a fading to orange yellow then red. The player health bar is always active on the screen and the boss health bar shows at the bottom of the screen once the player enters the boss room.

Audio sources were used to play the different sound effects, background music and the boss theme in game. For the main player controls the unity input handler was used along with a “Player Locomotion” script to move the player around.

All the above methods will be discussed in further detail later in the report.

# Chapter 2 - Background and Literature Review

The idea for the project came from inspiration from many different games across multiple genres with the focus of the project being on the AI in the game. The first game that was researched for the AI components of the project was Battlestate Games’ Escape From Tarkov, a hardcore and realistic online First-Person action RPG game with MMO features. Set in the fictional region of “Norvinsk” which is located on the frontier between Russia and Europe, Tarkov is a city that was plunged intro anarchy thanks to an event known as the Contract Wars. The city is roamed by former PMCs and Black Ops unis and more importantly the Scavenger Gangs, which make up most of the AI in the game (Battlestate Games, 2022)[[1]](#endnote-1).

Escape from Tarkov, in basic terms, is a “Looter-Shooter” in which the player enters a raid with other players and AI scavengers and searches the map for gear and items whilst killing other hostile players and the AI scavengers as they make their way to the extraction point. If the player dies in the raid, they lose all their gear, the only way to get the gear and items they have back to their Stash is to escape, Hence the name, “Escape From Tarkov” (Battlestate Games, 2022).

The scavenger AI in Escape From Tarkov seems to be designed around state machines. After playing the game for a while this becomes apparent as most of the AI are static and holding down key positions on the map that may have good loot or quest items. There also seems to be some implementation of Nav Mesh Agents as there are AI PMCs in some of the maps that are more difficult than the scavenger AI to kill but they have more gear and better rewards if you can manage to eliminate them (Battlestate Games, 2022).

A major hurdle in the games industry in the past 10 years has been “Shortest Pathfinding” which is the method of trying to find the most optimised and shortest path for an AI to take in order to complete the action that has been triggered, for example, moving towards the player when they are within range. Therefore, Navigation Mesh (NavMesh) is quite interesting because it has a large area of implementation and is effective in the shortest pathfinding problem because its optimization was made with effective tracing using segmentation line. In a paper discussing NavMesh pathing finding for ghosts in the Pacman game, it was concluded that NavMesh and an A\* algorithm, which is a searching algorithm that is used to find the shortest path between an initial point and a final point (Simplilearn, 2021), had become the best solution in solving the shortest pathfinding problem for the needs of the games industry today (Zikky, 2016).

Upon conducting further research into the area of AI it was discovered that the NPC (Non-Player Character) AI may be divided into several components although across multiple games those components may not be the same. For example, in fighting games like Tekken 7 and Mortal Kombat, combat AI is often the largest AI subsystem, and it may be further divided into enemy AI that fight the player and ally AI that control the NPCs to help the players in fights (Tom ́aˇs Plch, 2014). Non-Combat AI governs the rest of the NPC behaviour in most games, and it could be further divided into interactions such as bartering and dialogues such as the case in Bethesda’s Elder Scrolls V: Skyrim which boasts an in depth and detailed AI architecture (Bethesda, 2022).

In December 2020, CD Projekt RED released their next, highly anticipated title, Cyberpunk 2077 and the game got off to a rocky start to say the least. For PC players there were some minor bugs and technical glitches, but it all fell into the bounds of what was to be expected from such an ambitious game upon release and everyone expected those little bugs to be fixed soon enough in the days following the release of the game, but they weren’t. Instead, CD Projekt Red told fans that they wanted to do a thorough job of cleaning up the games unexpected flaws and that it would take some time to do so (Kljajic, 2020).

Console players had the worst experience of the game as there were reports of car pileups on the streets in the game and AI freezing on the spot in the middle of cutscenes and the player then losing progress when they had to quit the game and reload. Cyberpunk was riddled with lots of bugs, and it was apparent to a lot of the players that the game wasn’t ready for release, especially the console versions.

CD Projekt Red eventually released the long-awaited Update 1.5 in February 2022 which for PC users meant a rebuild of the perk trees, the ability to change the main characters appearance, new apartments to buy but most importantly AI Improvements (O'Connor, 2022). The update was well received and breathed a new lease of life into the game as the average player count increased by 51.35% after the update was released in February 2022 (SteamCharts, 2022).

After reviewing all this research, it was decided that a Navigation Mesh approach would be the most suitable for the game that was being developed. The Navigation Mesh could be used to tell the AI where it could navigate to and give it a patrol path and once the player comes within a certain distance the AI would then move the NPC towards the player and attack when within the necessary range. This implementation will be discussed in further detail later in the report.

# Chapter 3 - Problem Analysis & System Requirements

Most of the techniques and methods for implementing the systems that were required for the project were already explored in multiple modules during Year 2 and Year 3 of the course. This meant most of the basic systems for the game were implemented quickly and in a timely manner in accordance with the project plan.

The first big obstacle however would be the implementation of the AI in the game and as this was the focus of the game, most of the time spend on the project and all the implemented systems went to developing the AI algorithms and Navigation Meshes.

As previously stated, the AI implementation took some time to implement as there needed to be extensive research first in order to create a fully functional AI system which was responsive and seemed realistic and reasonable to the player.

As for system requirements for the project, the Unity engine was used for most of the development of the project as well as Visual Studio 2019. The reason for the older version of Unity instead of the most recent version was to ensure that there were no version errors whenever the project was update or any problems and as the task allowed me to use whichever software and version of software, the decision was made to keep the same version of Unity throughout the project to keep it consistent and reduce the amount of updating and possible errors that could occur.

Visual Studio 2019 was used as the programming environment as it was familiar from work on previous modules during Year 2 and Year 3. This familiarity helped a lot when trying to tackle errors and being able debug efficiently because of the familiarity with the environment.

As the focus of the project was on the AI and the scripting of the game, the Unity Assets Store was used to acquire assets for the game such as the level design prefabs like walls, floors and detailing objects. Character models were also acquired from the unity store and any audio files used have been referenced at the end of the report.

Using assets from the store instead of creating them from scratch saved a lot of time and it would not have been feasible to create every game object from scratch and develop the AI system in the given time frame.

Another Requirement for the development of the game was a high spec computer that could handle the computational power needed to develop a game in unity. Having multiple resource intensive applications running at the same time is common when developing a game. As mentioned in the project proposal, a system of a suitable specification was already acquired and was used for the duration of the project, the specification of the system is listed below are exceed the recommended specifications for all the software used during the development of the project:

|  |  |
| --- | --- |
| **Motherboard** | ASUS® ROG STRIX Z390-F GAMING |
| **CPU** | Intel® Core™ i7 Eight Core Processor i7-9700K (3.6GHz) 12MB Cache |
| **GPU** | 8GB ASUS ROG STRIX GEFORCE RTX 2080 SUPER - HDMI, DP |
| **RAM** | 16GB Corsair VENGEANCE RGB PRO DDR4 3200MHz (2 x 8GB) |
| **Storage** | 1TB INTEL® 660p M.2 NVMe PCIe SSD |
| **PSU** | CORSAIR 650W VS SERIES™ VS-650 |
| **Operating System** | Microsoft Windows 10 |
|  |  |

## Functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement ID** | **Requirement** | **Must/Want** | **Comments** |
| FR001 | User must be able to move the player | Must | Player must be able to move using WASD keys |
| FR002 | User must be able to make the character attack | Must | Player must be able to attack using Left Mouse Button |
| FR003 | Health Bar must be visible and working | Must | Health bar must be always visible on screen and when health depletes it should gradually change colour from green to red |
| FR004 | Animations for each model must work correctly | Must | Animations for each character models movement, attack etc. must work |
| FR005 | Hit detection must work for each character | Must | When a character attacks and hits another the OnCollision methods should trigger |
| FR006 | Player must die and menu screen show when health bar is depleted completely | Must | - |
| FR007 | Enemy AI Skeleton must patrol the nav mesh boundary | Must | The skeleton should only patrol the second level but not towards the stairs of the boss room |
| FR008 | Enemy AI Skeleton must chase player when the player gets within range | Must | - |
| FR009 | Skeleton must attack enemy player when in range and attack is not on cooldown | Must | - |
| FR010 | Hit detection for skeleton hitting player must work | Must | Take health from player when skeleton hits player and if health falls below 0, player dies |
| FR011 | Hit detection for player must work | Must | When the player is hit, they must take damage and it should reflect that on their health bar |
| FR012 | When player walks into boss room, the boss activation trigger should activate | Must | When the player enters the boss room, the boss roar sound should play, and the boss healthy bar and name should show at the bottom of the screen |
| FR013 | When the player gets within range of the boss the boss should chase | Must | - |
| FR014 | When the boss is within attack range of player the boss should attack | Must | - |
| FR015 | When the boss attacks the player, the player should lose health | Must | - |
| FR016 | The player should be able to attack the boss and damage the boss | Must | When the player attacks the boss, the boss should take damage and should be shown at the bottom of the screen on the boss’ health bar |
| FR017 | When the boss healthy is fully depleted the boss should die | Must | - |
| FR018 | When the player enters the boss room, the boss music should start playing | Must | Boss music starts playing when the player enters the boss room and after the boss roars |
| FR019 | Upon game start the ambient dungeon music should play | Must | - |
| FR020 | When the player hits an enemy, it should play the hit sound effect | Must | - |
| FR021 | The boss roar sound should play upon entering the boss room | Must | - |
| FR022 | There should be an Ally AI to help the player fight through the dungeon | Want |  |
| FR023 | The Ally AI should only heal the player sometimes when their health gets low | Want | Will roll random number when player health gets below 50% and if the number is odd, heal player, if not it goes on cooldown for 10secs and tries again |
| FR024 | The ally AI should follow the player | Want | - |

## Non-Functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement ID** | **Requirement** | **Must/Want** | **Comments** |
| N-FR001 | The game should launch without any errors | Must | - |
| N-FR002 | The game should run at a smooth framerate above 30fps | Must | - |
| N-FR003 | The game must not crash at any point in the level | Must | - |
| N-FR004 | There should be no input lag with the player controls for the user | Must | - |
| N-FR005 | The game should not stutter or freeze at all | Must | - |
| N-FR006 | The game must be able to be played in Windows and MacOS | Must | - |

# Chapter 4 – Design

As mentioned in the project proposal, an agile methodology approach was the initial plan to be used during the project but after some time in development after the interim report it didn’t seem to be the right approach as iterations weren’t being made, rather there was a struggle to get some iterations of the systems working. From then it was decided to switch to a waterfall approach which worked out a lot better. Work was done on one system until it was working a polished and only then would the next system be started.

Switching development methods during the development process usually isn’t a good idea and can disrupt the flow of work but as the project was being developed by a one-man team, only one person needed to make the switch and it worked out very well in the long run.

In normal circumstances this would not have worked as well when working in a team as expecting others to change the development cycle in the middle of a project usually means having to re-do all the design documentation and methods to fit the new methodology.

In order to analysis the requirements of the game effectively they needed to be matched with the aim of the project. The aim of the project was to create a 3rd person dungeon crawler game with a focus on the AI implementation of the enemy NPCs. One of the main objectives with developing the AI was getting the NavMesh working properly. A rough sketch can be seen in Appendix Six which shows the planned pathing areas for each of the enemy AI.

To create a game like this a few goals needed to be met such as implementing player movement controls, attacking and health scripts, the same for the enemy NPCs that would be controlled by the AI scripts. A rough outline of the script design that was written out at the start of the project can be seen in Appendix Five. The game also needed a level design which was made from scratch using models from the unity assets store pack, ultimate Low Poly Dungeon (Broken Vector, 2022). An early sketch of the intended level design can be seen in Appendix Four.

The models used in the game for the player (Codewart Game Assets, 2022), Skeleton enemy (IronicGame, 2017)and the dragon (Dungeon Mason, 2021) were also acquired from the Unity Assets store. Each model imported also included animations with it that were used for their various actions throughout the game. These models can be seen in the appendices (Appendix 1-3).

Other assets that were used in the project from the unity store include Torture Devices (iPoly3D, 2021), UltimateDungeonPack (Broken Vector, 2022)and the “Radagon Of The Golden Order” Theme from the critically acclaimed Elden Ring (Saitoh, 2022).

As mentioned previously in Chapter Two, after extensive research, it was decided that a Navigation Mesh approach would be the most suitable for the game that was being developed however that was not the only option presented when the subject was researched.

Another method of implementing the AI would have been using State Machines which are a concept in which a container stores the status of an object at any given time. Then if it’s given an input, it can provide an output based on the current state, transitioning to a new state in the process (Shora, 2020). State machines can be represented by State diagrams and an example of one can be seen in Appendix Seven.

The reason state machines were not used was because it felt easier to implement and use nav mesh agents as it was more visually engaging, and it seemed easier to use as you could see the area in which the character would be patrolling.

# Chapter 5 – Implementation

## Level Design

The entire level was created from scratch using assets from the unity assets store. Each wall, floor and ceiling were placed individually to create a completely unique map that fit the design sketch that was created prior to the development of the project. The design sketch can be seen in Appendix Four.

As mentioned in the design section there were a few asset packs that were used to create the final level design. Most of these asset’s packs were “Low Poly” as this would give the game better performance instead of using high resolution textures and models.

## Player Movement & Third Person View

There player movement controls were implemented using Unity’s input system which is an extra package that can be installed from the unity package manager. Using the input manager makes it a lot easier to implement player controls as the input system generates a lot of the code automatically.

To get start you can click on the player model being use and add a “Player Input” component and under that component click the “Create Actions” button which will prompt you to enter a name for a new file which is the player controls and when you open this file after it is created it displays a few pre-populated actions maps for player controls and even some for UI (Unity, 2021).

You can then simply assign Keys to each of the player control commands, Up, Down, left and Right to WASD. Next was the player locomotion script, which was created to move the player, set the players speed, handle the rotation of the player and play the footstep sounds whenever the player is moving. For implementing the player controls there was a very useful tutorial lesson on YouTube which teaches how to implement the Input System into existing scripts and new scripts in different ways (BMo, 2021).

## Player Attack

The player attack script is a simple script that has an attackCooldown method that uses an IEnumerator which waits for a set amount of time before resetting the canAttack Boolean back to true meaning g the player attack cooldown is gone, this is done so the player can’t spam attacks and it also has an axeAttack method which sets attacking to true and starts the cooldown for the next attack, plays the attack animation and the attack sound as well. The axeAttack method is called when the player clicks the Left Mouse button/Mouse 0.

## Character Animations

The player movement animations were implemented using blend trees and blend tree parameters. When the player presses one of the movement keys it starts the animation playing and when the player attack whilst running it pauses the running animation to play the attack animation then goes back to the running animation if the player is still moving.

Its very simple to implement blend trees as it was just a matter of bringing in the animations to the base layer and making simple transitions for some of the animations and blend trees for the movement. A few trigger parameters were created that could also be used for the attack animation and the death animation so when the players health reaches zero or when they attack/press the left mouse button, the trigger can be activated for the respective animation which plays the animation in game.

## Player Health Bar

In order to implement a health bar for the player there needed to be some research done first as to the different options of implementing this functionality. Eventually a method was found in which the health bas is implemented using a slider as the main visual component which would change colour as the players health decreased, it would go from green to red gradually (Graves, 2020).

The idea for the changing colour health bar also came from a paper on developing MOBA games using Unity engine and it talked about how each damageable object has a health bar that indicates how much damage that object or character has taken which could be indicated by red and how much health the character has left, indicated by green for example (D. Polanče, 2017).

The health bar on screen is an image with a fill and as the players health decreases it lowers the health bar and the colour of the health bar depending on how low their health is. Green being full health then it transitions to yellow, orange and finally red when the health bar is almost completely depleted.

The script controlling the Health Bar on screen is the HealthBarScript which also holds the values for the players max health and current health as well as having the TakeDamage method to decrease the players health when an enemy attacks them.

The HealthBarScript uses an animator to manage changing the size of health the player has left when they take damage and the changing colour of the health bar as the health decreases. Current Health is set to the max health variable on start and the same for the slider values.

The update method in this script checks if the players health is less or equal to 0 and if it is, the Death animation trigger is called, and the player death animation will play as well as the Game over function which is a method that makes a reference to the GameOverScreen.Setup(0 script which freezes the game and brings up the game over screen.

The TakeDamage() function in the HealthBarScript checks if the players health is less than or equal to 0 again and if it is, kills the player and does the same as mentioned in the update function but if the player does have health left it sets the current health equal to current health minus the enemy attack damage and sets the slider value and gradient colour to match this accordingly.

## Enemy/Boss Movement and Navigation Mesh

As mentioned earlier in the report, the enemy skeleton and the boss enemy, the nightmare dragon, both use the same AI and movement scripts. The two AI use navigation meshes and patrol the boundaries of these nav Meshes until they come into range of the player, then they will chase the player and start attacking them.

Some research needed to be completed in order to fully understand the methods of implementation for Navigation meshes and some quality tutorials that gave an in-depth analysis on Navigation Meshes helped the development of the games own AI and Nav Meshes massively (GDTitans, 2022).

The start method assigns variables that get the layers position as well as the enemy/boss position. In the update function then the moveToPlayer() method is called and if the distance to the player gets to a certain range, in other words if the enemy gets too close or within range of stack, they stop moving and will start attacking.

The moveToPlay method has a distance variable that is used to measure the distance between the player and the enemy/boss. This is done using a Vector3.distance method and then passing in the position of the player and enemy.

Next comes the if statement that states if the distance between them is less than or equal to howClose, which is the set distance the enemy/boss needs to be at before attacking, the animation trigger for the boss walking is activated in the AIScript boss and the boss keeps moving towards the player.

For the AIScriptEnemy version there is no animator trigger as the skeleton animations were set up using conditions on the transitions. The reason for doing two different ways was to show that there is more than one way to do these things and to also explore the option of trying it a different way.

The final condition on the script is the else which is there so that if the enemy/boss is close enough, they stop walking.

Although finite state machines were not used for the AI in this game, the idea of implementing it was explored early on as there was a very useful book by Dr. Davide Aversa that focused specifically on AI programming in unity which was very informative and very interesting. The book had lots of example code snippets for different types of AI with different behaviours and it helped a lot with coming to understanding the different methods of AI development in Unity (Aversa, 2022).

## Enemy & Boss Attack

The BossAttack and the EnemyAttack scripts are both the exact same, the only difference being the distance at which each character can attack, for the boss it can attack from a value of 7 or less and the skeleton from a value of 3 or less.

The script starts off with references to the animator, the enemyweapon and creating a two Booleans named bossCanAttack/enemyCanAttack and bossIsAttacking/enemyIsAttacking which will be used to give the boss a cooldown on his attacks like in the player script. There is also a float declared which will be used as the cooldown time for the attacks for both scripts, enemyAttackCooldown/bossAttackCooldown.

Each script also makes refence to their respective AI script, AiScriptEnemy and AIScriptBoss which are used to get the distance from the player. The Methods of the scrip start with the update function which is setup so that if the distance the boss/enemy is from the player is less than or equal to the minimum attack distance that is stated (3 for enemy and 7 for boss), the enemy/boss can attack, and it calls the enemyAttack method.

The enemy attack method starts the cooldown for the enemy attack right away by setting the canAttack bool to false and the isAttacking bool to true. Then the animation trigger for the attack animation is activated and the animation will play. After this there are coroutines set up in the form of IEnumerators which are there to make use of the WaitForSeconds function, which lets us have a cooldown for the attack and reset the canAttack back to true when the waitForSecond function has finished.

## Hit Detection

The HitDetection script handles all the damage events between the player, the skeleton enemy and the boss. There are references to player, enemy and boss attack scripts as well as the players healthbar script. There are also references made to the aiScripts of the enemy skeleton and the boss in order to access their movement later in the script.

In the OnTriggerEnter method we have a few if statements which deal which each type of attack, the first being the player attacking the enemy. If the tag of the object that enters the collision is the “Enemy” tag which is the skeleton and the player isAttacking is true, the animator is used to play the skeleton hit animation and the death animation as well as playing the hit sound effect.

As mentioned before the aiScript reference that was made earlier is then used to access the moveSpeed of the skeleton and set it to 0. The next two if statements deal with the player taking damage from the enemy skeleton and the boss. If the tag is player and the enemyIsAttack/bossIsAttacking, a float value is passed into the TakeDamage method of the healthScript which we made a reference to earlier.

The final if statement deal with the player dealing damage to the boss. Originally the plan was to have the player continually damage the boss and have the health bar deplete just like the players but there were several errors and setbacks that led to it not being able to be properly implemented so the boss dies in one hit. This was not the planned outcome but because of the deadline and delays in other aspects of the games development it wasn’t able to be implemented fully but at the very least the functionality is there for the boss to deal damage to the player and for the player to kill the boss.

## Boss Room Trigger Event

The bossRoomTrigger script is attached to a box collider that is placed right on the entrance to the boss room. When the player steps into the box collider the OnTriggerEnter method is called which uses the animator of the boss to play the animation trigger so the boss roar, this also plays the boss roar audio clip and starts the coroutine to delay the boss music from playing until the boss roar is finished. To ensure that the music doesn’t play again and loop over the top of itself if the player steps back into the box collider there was an alreadyPlayed Boolean used to prevent this so the entire functionality is surrounded by an if statement that only executes the code inside if the music hasn’t been played already.

In this script there is also an attempt at implementing a cutscene camera but there were a few bugs that couldn’t be worked out in time such as the camera switching and audio cutting out then the player animator would stop working when the camera switched back to the players from the cutscene camera (Loop, 2020). The code has been left in to show the attempt and that logically it should work, it just wasn’t finished in time and polished.

## Ambient Music /& Sound Effects

The ambient music was very simple to setup. An empty game object was created and named Music Control and an audio source was attached to it as well as a Music Control Script.

The Dungeon Music audio file was placed in the audio source and the script attached to it ensures that it plays when the game starts, that it never duplicates and when a new instance of the game is opened it restarts, e.g., when the restart button is pressed.

All the sound effects in the game have been discussed in previous sections.

## Lighting

The lighting for the level is a simple world light called development light that has default settings and then around the level there are torch models that have been placed and they also have point lights attached to them.

# Chapter 6 - Testing and Known Problems

## Testing Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test ID** | **Test** | **Expected Outcome** | **Input** | **Result** |
| T001 | User can be able to move the player in all directions | User can move player character using assigned controls for movement | WASD keys | As Expected |
| T002 | User can attack using the assigned control | Player can attack when clicking LMB | Left Mouse Button | As Expected |
| T003 | Health Bar is always visible and works. | Health Bar is visible and works. E.g., when player is damaged the health bar decreases and changes colour. | N/A | As Expected |
| T004 | Animations for each model must work correctly | Animations for each mode play correctly whenever the conditions are met | Various Animation Triggers | As Expected |
| T005 | Hit detection must work for each character | When the player hits the enemies, they should die and when the player is hit they should take damage to their health bar | N/A | As Expected |
| T006 | Test that when the player health bar reaches 0 the player should die | When the player health reaches 0 the death animation plays and the game over screen is shown | N/A | As Expected |
| T007 | Test enemy skeleton and boss AI | Test that enemy AI skeleton patrols from game start and AI boss tracks player when the player comes into range | N/A | As Expected |
| T008 | Enemy AI Skeleton must chase player when the player gets within range | Enemy AI Skeleton must chase player when the player gets within range | N/A | As Expected |
| T009 | Test that the skeleton attacks player | Skeleton must attack enemy player when in range and attack is not on cooldown | N/A | As Expected |
| T010 | Test that the skeleton dies when hit | Hit detection for skeleton hitting player must work when player attacks | N/A | As Expected |
| T011 | Test that the player can get hit by skeleton | Hit detection for player must work | N/A | As Expected |
| T012 | Test that the boss sequence plays when the player enters the boss room | When player walks into boss room, the boss activation trigger should activate | N/A | As Expected |
| T013 | Test that the boss AI works, and the boss chases the player | When the player gets within range of the boss the boss should chase | N/A | As Expected |
| T014 | Test that the boss attack works | When the boss is within attack range of player the boss should attack | N/A | As Expected |
| T015 | Test that the boss attacks damage player | When the boss attacks the player, the player should lose health | N/A | As Expected |
| T016 | Test that the player can hit the boss | The player should be able to attack the boss and damage the boss. Bos should die on one hit | N/A | As Expected |
| T017 | Test that the ambience sound works | Upon game start the ambient dungeon music should play | N/A | As Expected |
| T018 | Test that the hit sound effect plays when the player attacks | When the player hits an enemy, it should play the hit sound effect | N/A | As Expected |
| T019 | Test that the boss roar sound and animation work | The boss roar sound should play upon entering the boss room | N/A | As Expected |
| T020 | Test that the boss health bar and name show upon entering boss room | Boss health bar and name shows up at the bottom of screen when entering room | N/A | As Expected |
| T021 | Test that the restart button works on the game over screen | When the user clicks restart button on the game over screen the game should restart | N/A | As Expected |

## Evaluation of Developmental Success

Overall, the project was a success as the main objective of the project was to produce good AI that worked well through different behaviours. There were a few problems that were encountered during the development which persist in the final submission but regarding the main focus of the project, the development on AI enemies, they are a success as they work well in patrolling, pathfinding/finding the player and attacking the player.

After the development of the game had finished a questionnaire was created for 5 users to fill out after playing the game for a few minutes. The results of this survey support the argument that the project was a success as one of the questions “Would you like to see a fully developed and polished version of this demo game” received a yes answer from every game tester. The data from the google forms page can be seen in figure 9 of the appendices.

Most of the game testers included in the survey are avid gaming fans as 80% of them play games 10hrs+ per week with one playing an average of 5-10 hrs per week. Many of the participants referred top big games series’ such as Dark souls, Elden Ring and Devil May Cry when asked if the game had any elements in it that they have seen in other games. The data response for that question can be seen in Appendix 10.

When questioned on the behaviour of the AI in game and whether of not they seem realistic and like other big games most of the participants agreed that the AI seem natural and well made but there were a few problems such as the boss not having varying attack animations and the boss AI moving through the player sometimes making it hard to hit. The answers for this question can be seen in Appendix 11.

The participants were also asked what they liked most about the dungeon game and most of them agreed that the boss sequence was their favourite part of them game with some of them describing it as immersive and epic. One participant commented on how they liked the map design and the overall layout of the dungeon. The answers for this question can be seen in Appendix Twelve.

Finally, when the participants were asked if they had any suggestions for improvements that could be made to the game, specifically the AI there were a few different suggestions which start with the dragon hit box detection sometimes not working and the character being able to walk through the dragon. The player weapon not being attached to the hand was mentioned a few times also. One user suggested more animations for the boss and for the boss to have more attacks. One user also pointed out that sometimes the skeleton AI gets back up after dying and resumes chasing the player. The responses for this question can be seen in Appendix Thirteen.

## Known Problems

There are a few minor problems that are still persistent in the game. The only major problem is that the boss health and health bar does not work as intended. Originally the boss health bar was meant to be reduced gradually as the player attacked the boss so that the boss fight would last a while and be intense but because the boss damage script was not working as intended the boss had to be created as a one hit kill just like the skeleton.

Other minor problems with the game are that the axe weapon for the player is not attached to the hand so when the players hand move for the attack animation the axe does not move with it. Unfortunately, this couldn’t be fixed even when all options that were researched were tried such as position constraints, hard coding the position of the axe to the hand and even attaching the axe to the bone of the player’s right hand.

The skeleton AI in the game works very well and performs exactly as intended in all circumstances however sometimes if the players weapon collides with the skeletons body while it is dead it will restart the skeleton in some way and it will get back up and resume chasing and attacking the player.

The boss AI works as intended and when the player gets close enough the boss detects them and starts chasing and attack them however the boss doesn’t stop short of the player as is intended and it should stop juts a few metres before the player when it wants to attack or when it’s chasing the player but instead it walks into the player and makes the gameplay a bit annoying at times. A fix wasn’t found for this but hopefully with future development the error will be found and fixed.

# Chapter 7 – Conclusion

In conclusion, the main objective of this project was to create a third person dungeon game with a focus on the development of the AI enemies. A variety of techniques and methods were adopted to create the final submission and the result is quite good considering the time frame and the entire development being created by one person.

The main objective of the project has been completed along with other ele52ments of the project that have all been completed in time for the final submission. Although most of the objectives of the game are complete they are not all as polished as they could be and that will be a task for future development of the project to add to a portfolio of work in the future.

Overall, I am very proud of what UI have achieved in the given time frame and given that most successful games are developed by hundreds of developers and usually taking them a few years, I believe what I have achieved is commendable and I undertook a project that was probably bigger and had more elements to it than most.

# Chapter 8 - Critical Evaluation

The main requirements of the project have been completed and most of the aspects that were outlined in the initial proposal have been implemented. As for the requirements and aspects that were proposed initially that haven’t been implemented the reason for them not being completed and implemented in the game is that some of the earlier components took longer than was originally planned to develop. The AI for the boss and the skeleton especially took a few weeks to fully implement to the standard they are at now.

The main requirement that was initially outlined in the project proposal that is now missing from the game is the Ally AI that was originally going to be implemented to help the player fight through the dungeon but because of the development set backs for the other AI and for other parts of the project there wasn’t enough time to implement them into the project, instead the focus was put on the enemy AI and perfecting them as much as possible in the time frame given.

As mentioned earlier in the report originally the development cycle chosen was originally agile but as development was slow and there were a lot of barriers early on in development especially with the AI, I felt it was better to switch to a waterfall approach and ensure that each aspect of the project was completed before moving on to the next and I feel like this worked out for the better and was a very good decision which benefitted the project in the end.

In terms of implementation, I feel as though everything I did regarding programming was done well, C# was used to write the scripts in Visual Studio, and they were then used accordingly in unity. Some references were made to other scripts in some of the scripts to allow interactivity and access to variables and methods that needed to be used in that script.

When looking at the original project plan I can safely say that after the middle of march when work had been underway for the AI scripts, the project plan wasn’t followed as strictly as it had been before because I had started getting errors that were hard to fix and development issues when I needed to do extensive research on AI and how to implement them in the game that took some time which I didn’t account for originally.

Towards the end of march, I start using the project plan as more of a guideline as to what I need done and by when so everything could be done in a timely manner, but I didn’t stick to it as much as I could have or should have. There were times where tasks took a lot longer than was planned and that threw me off and I would have to rearrange the time allocations for each task like when the AI took longer than expected and the boss sequence.

## Further Work

In the future I plan to continue the development of the game and fix some of the problems that weren’t fixed in time such as the boss health bar functionality, polishing the AI more and implementing the Ally AI to help the player through the dungeon. I also plan to extend the level of the game and create more floor to the dungeons and more bosses. I plan to add the game to my portfolio website that I have created showcasing all the working I’ve done in Software Development and Games Programming over the last 5 years.

# Bibliography

Aversa, D. D., 2022. *Unity Artificial Intelligence Programming.* Fifth ed. Birmingham: Packt Publishing.

Battlestate Games, 2022. *Official Escape from Tarkov Website - About.* [Online]   
Available at: https://www.escapefromtarkov.com/  
[Accessed 12 05 2022].

Bethesda, 2022. *The Elder Scrolls.* [Online]   
Available at: https://elderscrolls.bethesda.net/en/  
[Accessed 11 05 2022].

BMo, 2021. *How to use Unity's New INPUT System EASILY.* [Online]   
Available at: https://www.youtube.com/watch?v=HmXU4dZbaMw  
[Accessed 15 05 2022].

Broken Vector, 2022. *Ultimate Low Poly Dungeon.* [Online]   
Available at: https://assetstore.unity.com/packages/3d/environments/dungeons/ultimate-low-poly-dungeon-143535  
[Accessed 10 05 2022].

Broken Vector, 2022. *Ultimate Low Poly Dungeon.* [Online]   
Available at: https://assetstore.unity.com/packages/3d/environments/dungeons/ultimate-low-poly-dungeon-143535  
[Accessed 01 04 2022].

Codewart Game Assets, 2022. *LOWPOLY - Universal Character (Modular).* [Online]   
Available at: https://assetstore.unity.com/packages/3d/characters/lowpoly-universal-character-modular-199781  
[Accessed 11 03 2022].

ConsoleGamesWiki, 2022. *Trust System.* [Online]   
Available at: https://ffxiv.consolegameswiki.com/wiki/Trust\_System  
[Accessed 09 05 2022].

D. Polanče, I. M., 2017. *Developing MOBA games using the Unity game engine.* Opatija, IEEE.

Dungeon Mason, 2021. *Dragon for Boss Monster : HP.* [Online]   
Available at: https://assetstore.unity.com/packages/3d/characters/creatures/dragon-for-boss-monster-hp-79398  
[Accessed 25 04 2022].

GDTitans, 2022. *3D ENEMY AI in UNITY - (E02): CHASE AND ATTACK.* [Online]   
Available at: https://www.youtube.com/watch?v=b-WZEBLNCik  
[Accessed 02 05 2022].

Graves, S., 2020. *Create DARK SOULS in Unity ► EP. 10 Player Health & Damage.* [Online]   
Available at: https://www.youtube.com/watch?v=iJSfBwC121c&t=529s  
[Accessed 11 05 2022].

iPoly3D, 2021. *LowPoly Medieval Torture Devices.* [Online]   
Available at: https://assetstore.unity.com/packages/3d/props/lowpoly-medieval-torture-devices-190333  
[Accessed 04 04 2022].

IronicGame, 2017. *RPG - Skeleton.* [Online]   
Available at: https://assetstore.unity.com/packages/3d/characters/humanoids/fantasy/rpg-skeleton-35463  
[Accessed 11 04 2022].

Kljajic, A., 2020. *Cyberpunk 2077's poor AI is related to bugs and will be fixed, says CDPR.* [Online]   
Available at: https://www.altchar.com/game-news/cyberpunk-2077s-poor-ai-is-related-to-bugs-and-will-be-fixed-says-cdpr-avGWJ5j9Eqvg  
[Accessed 11 05 2022].

Loop, 9., 2020. *How To Animate Your Camera In Unity.* [Online]   
Available at: https://www.youtube.com/watch?v=yLoAKlKlM4E  
[Accessed 15 04 2022].

O'Connor, A., 2022. *Cyberpunk 2077 update 1.5 brings better AI, reworked perks, and appearance customisation.* [Online]   
Available at: https://www.rockpapershotgun.com/cyberpunk-2077-update-15-is-out-now-with-better-ai-reworked-perks-appearance-customisation-and-more  
[Accessed 11 05 2022].

Saitoh, T., 2022. *Radagon of the Golden Order.* [Sound Recording] (FromSoftware).

Shora, N., 2020. *State Pattern using Unity.* [Online]   
Available at: https://www.raywenderlich.com/6034380-state-pattern-using-unity  
[Accessed 21 04 2022].

Simplilearn, 2021. *A\* Algorithm Concepts and Implementation.* [Online]   
Available at: https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/a-star-algorithm#:~:text=What%20is%20an%20A\*%20Algorithm,shortest%20path%20to%20be%20taken.  
[Accessed 10 05 2022].

Square Enix, 2022. *Final Fantasy XIV - Online.* [Online]   
Available at: https://www.finalfantasyxiv.com/  
[Accessed 09 05 2022].

SteamCharts, 2022. *Cyberpunk 2077 - Active Players.* [Online]   
Available at: https://steamcharts.com/app/1091500  
[Accessed 10 05 2022].

Tom ́aˇs Plch, M. M. P. O. ́., 2014. An AI System for Large Open Virtual World. 1(2), pp. 44-51.

Unity, 2021. *Input Manager.* [Online]   
Available at: https://docs.unity3d.com/Manual/class-InputManager.html  
[Accessed 13 05 2022].

Zikky, M., 2016. Review of A\* (A Star) Navigation Mesh Pathfinding as the Alternative of Artificial Intelligent for Ghosts Agent on the Pacman Game. *EMITTER International Journal of Engineering Technology,* 4(1), p. 9.

# Appendices

## Appendix One – Player Model

A person in a garment

Description automatically generated with medium confidence

## Appendix Two – Skeleton Model



## Appendix Three – Dragon Model

A picture containing dinosaur, reptile

Description automatically generated

## Appendix Four – Level Design Sketch

Diagram

Description automatically generated

## Appendix Five – Script Design

Text, letter

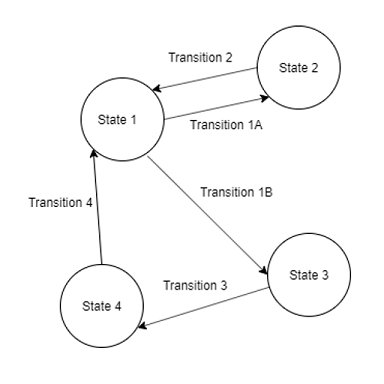
Description automatically generated

## Appendix Six – Nav Mesh and AI Pathing Design

A picture containing diagram

Description automatically generated

## Appendix Seven – State Machine Diagram



## Appendix Eight – Project Plan

Chart

Description automatically generated

## Appendix Nine – Survey Data One

Chart, bubble chart

Description automatically generated

## Appendix Ten – Survey Data Two

Graphical user interface, text, application, email

Description automatically generated

## Appendix Eleven – Survey Data Three

Graphical user interface, text, application, email

Description automatically generated

## Appendix Twelve – Survey Data Four

Graphical user interface, text, application, email

Description automatically generated

## Appendix Thirteen – Survey Data Five

Text, timeline

Description automatically generated with medium confidence

1. [↑](#endnote-ref-1)