3D First Person Shooter PC Game BY

ADEBANJO, IBUKUNOLUWA MICHAEL BU/18C/IT/3242

DEPARTMENT OF SOFTWARE ENGINEERING FACULTY OF COMPUTING AND APPLIED SCIENCES

BAZE UNIVERSITY, ABUJA

December, 2020

Monsters Within

Thesis Submitted in Partial Fulfilment of the Requirement for the Degree of

B.Sc.

In

Computer Science, Software Engineering

By

ADEBANJO, IBUKUNOLUWA MICHAEL

To

The Department of Computer Science Baze University, Abuja

DECEMBER, 2020

**DECLARATION**

This is to certify that this Thesis entitled Monsters Within, which is submitted by Michael Ibukunoluwa Adebanjo in partial fulfilment of the requirement for the award of degree for B.Sc. in Information Technology to the Department of Computer Science, Baze University Abuja, Nigeria, comprises of only my original work and due acknowledgement has been made in the text to all other materials used.

Date: Date-12-2020 Name of Student: Ibukunoluwa Michael Adebanjo

**APPROVED BY** …………………………………..

### Head

Department of Computer Science

**CERTIFICATION**

This is to certify that this Thesis entitled Monsters Within, which is submitted by Ibukunoluwa Michael Adebanjo in partial fulfilment of the requirement for the award of degree for B.Sc. in Information Technology to the Department of Computer Science, Baze University Abuja, Nigeria is a record of the candidate’s own work carried out by the candidate under my/our supervision. The matter embodied in this thesis is original and has not been submitted for the award of any other degree.

Date: Supervisor:

**APPROVAL**

This is to certify that the research work, 3D First Person Shooter PC Game and the subsequent preparation by Ibukunoluwa Michael Adebanjo with BU/18C/IT/3242 has been approved by the Department of Computer Science, Faculty of Computing and Applied Science, Baze University, Abuja, Nigeria.

By

Charles Isah Saidu Date

1st Supervisor

Nasiru Aboki Date

2nd Supervisor

Dr. C. V Uppin Date

Head of Department

Prof M. B. Hammawa Date

Dean, Faculty of Computing and Applied Science

Prof. A. B. Garko Date

External Examiner

**DEDICATION**

I would like to dedicate my work to my wonderful family who have wholeheartedly supported me throughout the development of this project; they gave me the motivation I needed to face all the problems I encountered head on and refine it to a greater extent. I would also like to dedicate this work to the YouTube Gaming content creators and the wonderful community members active on the Unity forums for sharing their various techniques and practices in regards to gamer development.

**ABSTRACT**

Monsters Within is a fun 3D FPS horror survival game on the Windows platform that demands its players to have fast reflexes, quick thinking and deductive skills to survive in the game environment they have been thrust into; distorted humanoid creatures have overrun the city and continue to rapidly multiply. The player’s main objective is to circumvent this by destroying all nests in the vicinity.

The quantitative research methodology was used to analyse the feasibility for the success for such a project on the Steam platform (the online store that the project will be published on). Competitive benchmarking and Open Web research were conducted to observe the success of competitors on the platform; their success and failures, as well as pricing structures, game mechanics and customer reviews were analysed to discover the core problems that needed to be solved to create a contending project.

Quantitative research is a systematic investigation that makes use of quantifiable data and verifiable facts to measure variables (reviews, sales numbers, etc.) and form generalizations from a large population size. The goal of this type of research is to outline specific research patterns.

The reviews, pricing structures, genres, date of release and overall game quality of the following titles were observed. Among Us, Phasmophobia, Titanfall, Call of Duty, Little Nightmares.

Games that had excessive micro-transactions were frowned upon by the gaming community regardless of whether it had a price or was free-to-play. Games that were of relatively low quality that had a multiplayer functionality still found success. The results showed that gamers deeply appreciated games with detailed level design and ambience, games that portrayed a puzzle like story were critically acclaimed among the audience. Games with detailed, atmospheric environments greatly appeal to gamers of the horror genre.

✔ People who play FPS games enjoy challenging gameplay over other aspects of game mechanics

✔ Uniqueness is the main factor that helps games stand out from other titles on the platform

**TABLE OF CONTENTS**

[**ABSTRACT V**](#_bookmark0)

[CHAPTER 1: INTRODUCTION 1](#_bookmark1)

* 1. [OVERVIEW 1](#_bookmark2)
  2. [BACKGROUND AND MOTIVATION 1](#_bookmark3)
  3. [STATEMENT OF THE PROBLEM 2](#_bookmark4)

[THE GAME HAS TO OFFER A FUN INTERACTIVE GAMEPLAY EXPERIENCE THAT IS CHALLENGING IN](#_bookmark5) [BOTH TERMS OF REACTION TIMES AND QUICK DECISION MAKING. THE STORY NARRATIVE HAS TO](#_bookmark5) [VIVIDLY ILLUSTRATED FROM THE PLAYER’S SURROUNDINGS AND INTEGRATIONS OF IT; WITH THAT IN](#_bookmark5) [MIND, THE ENVIRONMENT MUST ALSO ENCOURAGE EXPLORATION THROUGH THE IMPLEMENTATION](#_bookmark5) [OF INTUITIVE LEVEL DESIGNS (E.G., PLACING CERTAIN GAME OBJECTS IN A SPECIFIC MANNER TO](#_bookmark5)

[SUBTLY HINT TO THE PLAYER AT WHERE REWARDS AND NEW PIECES OF INFORMATION MAY LIE IN](#_bookmark5) [WAIT.). GAMES OF COMPETITORS OFTEN DO THIS BY PLACING A CAMERA FOCUS IN AREAS THEY WANT](#_bookmark5) [THE PLAYER TO EXPLORE, MAKING CERTAIN SOUNDS AUDIBLE FROM A CERTAIN DIRECTION OR](#_bookmark5) [LEAVING VISUAL CUES SUCH AS A BLOODIED TRAIL TO FOLLOW. THE GAMEPLAY MUST BE FUN TO](#_bookmark5) [REPEAT TO GRANT OVERALL SATISFACTION TO PLAYERS IN TERMS OF GAME MECHANICS 2](#_bookmark5)

* 1. [AIM AND OBJECTIVES 2](#_bookmark6)
  2. [SIGNIFICANCE OF THE PROJECT 2](#_bookmark7)
  3. [PROJECT RISKS ASSESSMENT 3](#_bookmark8)
  4. [SCOPE/PROJECT ORGANIZATION 4](#_bookmark9)

[CHAPTER 2: LITERATURE REVIEW 5](#_bookmark10)

* 1. [INTRODUCTION 5](#_bookmark11)
  2. [HISTORICAL OVERVIEW 5](#_bookmark12)
  3. [RELATED WORK 5](#_bookmark13)
  4. [SUMMARY 6](#_bookmark14)

[CHAPTER 3: REQUIREMENTS, ANALYSIS, AND DESIGN 7](#_bookmark15)

* 1. [OVERVIEW 7](#_bookmark16)
  2. [PROPOSED MODEL 7](#_bookmark17)
  3. [ITERATIVE METHODOLOGY 7](#_bookmark18)
     1. [METHOD (OBSERVATION) 8](#_bookmark19)
  4. [TOOLS AND TECHNIQUES 8](#_bookmark20)
  5. [ETHICAL CONSIDERATION 9](#_bookmark21)
  6. [REQUIREMENT ANALYSIS 9](#_bookmark22)
  7. [REQUIREMENTS SPECIFICATIONS 10](#_bookmark23)
     1. [Functional Requirement Specifications 10](#_bookmark24)
     2. [Non-Functional Requirement Specifications 10](#_bookmark25)
  8. [SYSTEM DESIGN 11](#_bookmark26)
     1. [Application Architecture 11](#_bookmark27)
     2. [Use Case 12](#_bookmark28)

[3.8.4 Activity Diagrams 13](#_bookmark30)

[3.8.6 User Interface Design 14](#_bookmark32)

* 1. [Summary 20](#_bookmark33)

[CHAPTER 4: IMPLEMENTATION AND TESTING 21](#_bookmark34)

* 1. [OVERVIEW 21](#_bookmark35)
  2. [MAIN FEATURES 21](#_bookmark36)
  3. [IMPLEMENTATION PROBLEMS 21](#_bookmark37)
  4. [OVERCOMING IMPLEMENTATION PROBLEMS 21](#_bookmark38)
  5. [TESTING 24](#_bookmark39)
     1. [Tests Plans (for System Testing) 25](#_bookmark40)
        1. [Reference 25](#_bookmark41)
        2. [Introduction 25](#_bookmark42)
        3. [Features Tested 25](#_bookmark43)
        4. [Features Not Tested 26](#_bookmark44)
        5. [Approach 26](#_bookmark45)

[The built-in console debuggers in unity and the Visual Studio IDE were used to observe the operations](#_bookmark46) [that occurred in the game in each and every game test scenario to see what wasn’t done properly or](#_bookmark46) [wasn’t accounted for. Going through the documentation for the Unity game engine and its packages](#_bookmark46) [proved to be crucial in solving most of the errors that occurred during the game's implantation. 26](#_bookmark46)

* + - 1. [Test Deliverables 26](#_bookmark47)

[APPROVALS 26](#_bookmark48)

* + 1. [Test Suite (for System Testing) 27](#_bookmark49)
    2. [Test Traceability Matrix (for Unit Testing, Integration Testing, and System](#_bookmark50) [Testing) 27](#_bookmark50)
    3. [Test Report Summary (for System Testing) 28](#_bookmark51)
    4. [Error Reports and Corrections 28](#_bookmark52)
  1. [USER GUIDE 28](#_bookmark53)
  2. [SUMMARY 29](#_bookmark54)

[CHAPTER 5: DISCUSSION, CONCLUSION, AND RECOMMENDATIONS 30](#_bookmark55)

* 1. [OVERVIEW 30](#_bookmark56)
  2. [OBJECTIVE ASSESSMENT 30](#_bookmark57)
  3. [LIMITATIONS AND CHALLENGES 30](#_bookmark58)
  4. [FUTURE ENHANCEMENTS 31](#_bookmark59)
  5. [RECOMMENDATIONS 32](#_bookmark60)
  6. [SUMMARY 32](#_bookmark61)

[REFERENCES 33](#_bookmark62)

[APPENDICES 35](#_bookmark63)

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| TABLE 1.1 | PROJECT RISK ASSESSMENT TABLE | 3 |
| TABLE 2.1 | FUNCTIONAL REQUIREMENT SPECIFICATIONS | 9 |
| TABLE 2.2 | NON-FUNCTIONAL REQUIREMENT SPECIFICATIONS | 9-10 |
| TABLE 4.5 | TEST TRACEABILITY MATRIX | 25-26 |
| TABLE 4.51 | TEST REPORT SUMMARY | 26-27 |
| TABLE A.1 | TEST CASE TC-001(MENU UI INTERACTION) | 49 |
| TABLE A.2 | TEST CASE TC-002(FPS MECHANICS) | 50 |
| TABLE A.3 | TEST CASE TC-003(SAVING AND LOADING GAME DATA) | 51 |
| TABLE A.4 | TEST CASE TC-004(ENEMY AI BEHAVIOR) | 52-53 |
| TABLE A.5 | TEST CASE TC-005(ENEMY AI BEHAVIOR) | 53-54 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| FIGURE 3.1 | APPLICATION ARCHITECTURE | [10](#_bookmark29) |
| FIGURE 3.2 | USE CASE DIAGRAM | 11 |
| FIGURE 3.3 | ACTIVITY DIAGRAM | 12 |
| FIGURE 3.4 | ENTITY RELATIONSHIP DIAGRAM | 13 |
| FIGURE 3.5 | MAIN MENU | 14 |
| FIGURE 3.6 | LOAD SCREEN | 14 |
| FIGURE 3.7 | GAMEPLAY | 15 |
| FIGURE 3.8 | INVENTORY | 15 |
| FIGURE 3.9 | ENEMY | 16 |
| FIGURE 3.10 | PAUSE MENU | 16 |
| FIGURE 3.16 | LOAD/SAVE GAME FILE | 17 |
| FIGURE 3.16 | GENERAL SETTINGS | 17 |
| FIGURE 3.13 | GRAPHICS SETTINGS | 18 |
| FIGURE 3.14 | CONTROL SETTINGS | 18 |
| FIGURE 3.15 | GAME ENVIRONMENT | 19 |
| FIGURE 3.16 | BOSS NEST(OBJECTIVE) | 19 |
| FIGURE 4.1 | SCENE LOADER MANAGER | 21 |
| FIGURE 4.2 | SCENE BUILD SETTINGS | 21 |
| FIGURE 4.3 | POINT LIGHTS | 22 |
| FIGURE 4.4 | MESH COLLIDERS | [22](#_bookmark31) |
| FIGURE 21.0 | LITTLE NIGHTMARES OVERALL STEAM REVIEW CHART | 36 |
| FIGURE 21.1 | LITTLE NIGHTMARES POSITIVE REVIEW 1 | 37 |
| FIGURE 21.2 | LITTLE NIGHTMARES POSITIVE REVIEW 2 | 38 |
| FIGURE 21.3 | LITTLE NIGHTMARES POSITIVE REVIEW 3 | 39 |
| FIGURE 21.4 | LITTLE NIGHTMARES NEGATIVE REVIEW | 40 |
| FIGURE 22.0 | SEKIRO OVERALL STEAM REVIEW CHART | 41 |
| FIGURE 22.1 | SEKIRO POSITIVE REVIEW 1 | 42 |
| FIGURE 22.2 | SEKIRO POSITIVE REVIEW 2 | 43 |
| FIGURE 22.3 | SEKIRO NEGATIVE REVIEW | 44 |

**LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| 3D | 3 Dimensional |
| AI | Artificial Intelligence |
| AAA  publishers | An informal classification used for games produced by major |
| ERD | Entity Relationship Diagram |
| FPS | First Person Shooter |
| GUI | Graphical User Interface |
| IK | Inverse Kinematics |
| IDE | Integrated Development Environment |
| LOD | Level of Detail |
| NES | Nintendo Entertainment System |
| PC | Personal Computer |
| UI | User Interface |
| UML | Unified Modelling Language |
| HUD | Heads Up Display |

* 1. **Overview**

# CHAPTER 1: INTRODUCTION

The project is a 3D FPS game constructed with the 2019.4 version of Unity’s game engine along with the C# programming language. The project entitled Monsters Within, set to be released on the Microsoft Windows platform, is a game that has the player controlling a soldier that has been deployed to an outbreak site in the dark, bustling, overrun city of Kuzimu; the area has been overrun with hostile distorted humanoid creatures spawning heaps of biomass that the player has the task of destroying whilst defending themselves form such creatures.

In the gaming space, there is a lack of fast paced horror games that engage the players in a detailed enthralling environment while offering gameplay mechanics that require a certain degree of skill from the player in terms of reflexes. The aim of the game project is to offer a healthy mix of these genres into something the players can have fun with but also explore the scenery of the game and the dark story hidden within it.

## Background and Motivation

The video gaming industry has been prevalent in society as far back as the early 1970s with the release of the popular arcade game Pong on the Atari console. Years passed and soon, the NES console in 1985 popularized home consoles. 3D games came into play with the release of Sega’s Saturn system in 1995 and competitors such as Nintendo and PlayStation continued to follow suit. (History.com Editors, 2018)

Now, in the modern age of gaming, the industry is larger than before with a player base of 2.7 billion people worldwide and an estimated worth of $159.3 billion by the end of 2020. This makes the industry even bigger than the global movie production and distribution industry which is worth $136 billion. Revenue generated from these games tied into the country’s revenue with China, United States of America and Japan standing at the top three podium positions. Nigeria isn’t within the top 25 countries that take advantage of such a lucrative industry. (G, 2020)

Game development can also prove to be a prospect for a new way of earning a living. And for the consumers it is important not only for the mental health of Nigerians to

engage with fun interactive activities to cope with the stressful nature of the country but it also improves one’s cognitive abilities and memory. (Dr.Reed, 2017)

## Statement of the Problem

The game has to offer a fun interactive gameplay experience that is challenging in both terms of reaction times and quick decision making. The story narrative has to vividly illustrated from the player’s surroundings and integrations of it; with that in mind, the environment must also encourage exploration through the implementation of intuitive level designs (e.g., placing certain game objects in a specific manner to subtly hint to the player at where rewards and new pieces of information may lie in wait.). Games of competitors often do this by placing a camera focus in areas they want the player to explore, making certain sounds audible from a certain direction or leaving visual cues such as a bloodied trail to follow. The gameplay must be fun to repeat to grant overall satisfaction to players in terms of game mechanics.

## Aim and Objectives

Weapons pick up locations will be intentionally placed on street pavements to encourage players to explore buildings and alleyways; this also subtly hunts that buildings are somewhat safe havens in game. Enemies will spawn from the nest so players will instinctively follow the path that they come from to learn where the objective to destroy is located. This design was chosen to make it feel like the player found the correct path based on their own merit which in turn grants them satisfaction. Collectibles will be placed in specific tight spots that players can pick up/observe to feed into the overall lore of the story.

## Significance of the Project

The implementation of this project will grant video game enthusiasts around Nigeria and the world at large a fresh source of interactive entertainment. This research encourages gamers to form new perspectives of how they think when approaching problems in the digital space as all good games should. A precise report on all phases of the

implementation of the project will be made available and the capacity for expanding upon the games structure will be set in place. This research project is currently set as a unique gaming experience in terms of its gameplay, and hence brings forth a fresh avenue for research and development in this genre. *Monsters Within*, the horror FPS game, has a unique experience to offer that will actively engage the way players think.

Due to the project being an Indie title (a video game crafted by individuals/small development teams without the financial and technical support of a large game publisher), it will be marketed as free-to-play and have considerate pricing structure for in-game cosmetics. Upon the successful release of the game, it will be marked as “Fair Use” under Licensing so that players will have the choice to freely stream the game onto suitable platforms such as Twitch/YouTube as means to earn revenue for themselves if they so choose to.

## Project Risks Assessment

**RISKS**

**Table 1.1: Project Risk Assessment Table**

|  |  |
| --- | --- |
| Inability to carry out research due to loss of time/budget constraints | Incorporate free-to-use assets, make use of pre-built packages to speed up the development process of the game |
| Loss of work due to equipment failure /loss | Version control set in place via Unity Hub’s collaboration functionality |
| Low quality final product | Post processing packs will be installed and LODs will be implemented to give the game the look of a  modern AAA title |
| Game only running properly  on high powered PCs | Optimize mesh data with LODs to make the game  accessible to a wider audience |

## Scope/Project Organization

The following sections of this project documentation describes the literature review of the technology and techniques used to develop the project in Chapter 2. The requirements, analysis and design implemented to make this project possible are outlined in Chapter 3. In Chapter 4, the implementation and testing of the game was addressed. In the remaining chapters, the summary, conclusions, limitations, links to the source code/project file and suggested improvements for the system are included.

# CHAPTER 2: LITERATURE REVIEW

## Introduction

This chapter is based on the literature review of the techniques and technology used in this thesis. Section 2.2. is based on the Historical Overview of developing games with the Unity game engine. C# is the programming language implemented in unison with Unity and as such, Section 2.3 gives descriptions of some of the literature available with regards to the application of C# in solving the research question. Lastly, section 2.4 is a summary of the entire chapter.

## Historical Overview

The work presented in this project is based on developing a Windows game in a 3D perspective.

C# is a general purpose, object-oriented programming language developed by a Microsoft team within the .NET initiative which was led by Anders Hejlsberg. (Bolton 2019)

It was integrated into Unity as its primary scripting language. Unity itself is a game engine (a software development environment for people to create games). It was developed by Unity Technologies and published in 2005 by Apple Inc. initially as a MAC OS X exclusive [reference]. The software is currently based on a subscription model where creators earning under $100,000 annually may use the full functionality of the engine for free. Developers who exceed this limit pay either a Pro, Plus or Enterprise yearly subscription depending on the amount of revenue. (Unity Technologies, n.d.)

## Related Work

Video games over the years have served as a fun source of entertainment for people of all ages; the earliest video game which was very well received is the Pong game dating back to 1968; which had two white rectangles on opposite ends of the screen that players could move vertically. The objective was to hit the square shaped ball past the other rectangle that was controlled by the other player. (History.com Editors, 2019)

From then on, games such as Mario, Sonic, Tetris, etc. were developed and published on ever changing consoles that served as the platform to host such game titles.

These games were originally developed via hard coding of low-level languages and OpenGL architecture. In modern times, they are now developed through the use of game engines - (software development tools specifically designed but not limited to the creation of video games on specific targeted platforms. i.e., Unity, Unreal Engine, Godot). (History.com Editors, 2019)

## Summary

The literature reviewed in this chapter illustrates the relevance of video games over the times and in today's present times. it also shows that while there is still a need for a programming skill set, a comprehensive understanding of gaming mechanics and game logic and overall proper planning in regards to time and resource management as well as a strong sense of direction for what ought to be set for the game’s foundation and future proofing for how it can be further expanded upon.

Therefore, it is important to explore different resources that are both readily available and offer a better horror themed game that offers a unique and entertaining experience for players in the gaming ecosphere.

Chapter 3 entails the requirement, analysis and methodology chosen to tackle the challenge of developing this video game project.

# CHAPTER 3: REQUIREMENTS, ANALYSIS, AND DESIGN

## Overview

This chapter specifies the requirement, analysis and design used in the development of this game.

Data was intentionally collected through customer reviews from the steam store; from these gatherings, an accurate depiction of the functional and non-functional requirements was created. The tools and techniques used to help develop the project will be discussed in this chapter and diagrams illustrating the application architecture will be listed.

## Proposed Model

For the development of this game, the iterative model was selected as the most appropriate methodology to utilize.

This methodology is a procedure for creating multiple varying versions (iterations) of a product in a repetitive cycle development, designing and testing.

Several versions of the product are tested at each stage of development; each version offers something new to learn about the functionality, design and usability of the product with every subsequent iteration being an improvement over the last. (Shamil, 2020)

## Iterative Methodology

This project implements the iterative model. This methodology is well suited for subsequent enhanced reversions of software which is generally how games are developed in the industry. It also allows for good version control and is accommodating to changes in project requirements.

This methodology is best suited for solo game development as it is less risky; with this process, I was never fully committed to just one design/material. As such, by testing every iteration of the design before the manufacturing process begins, design details that aren't user friendly, easy to develop or cost efficient can be eliminated early on.

With the continuous testing that took place, I was also able to refine the overall quality of the game.

Pertaining to the nature of this project, there are two main drawbacks of using this methodology. The first is that changing requirements can often lead to an overhead in time and budget. The other disadvantage is that getting an accurate estimate of the project completion becomes difficult due to the continuous changes in the software. These drawbacks were mitigated by integrating free-to-use assets (i.e., audio, textures and 3D models) to save up on the time and cost it would normally take to create them from scratch. In regards to the project completion date, due to the nature of games, I decided to aim for a game ready for a beta release in the event that the game wasn’t fully completed which will in turn allow for more in-depth feedback from customers and a faster detection of bugs. (Shamil, 2020)

## Method (Observation)

This section entails the path taken at each stage of the iterative methodology. In the requirement stage, the requirements were formed through the conducted observations of games in similar categories to what the final product was envisioned to be. All their data (system requirements, date of release, overall rating, gameplay and customer feedback) were taken into account to set standards for the project.

The stages of implementation, testing and review were performed in iterations. During implementation, scripts were written and corresponding components were added and tweaked. For this project, gray box testing as the game could be run within the engine to be able to play it from the user’s perspective. As I am the one who wrote the scripts and integrated each component, I have knowledge of the inner workings of the system. And during runtime or exiting “Play Mode” the console window shows all warnings and errors if any when certain actions are performed or during specific scenarios so that developers are able to more easily catch and fix bugs/logical errors if any. Iterations were reviewed based on whether the function worked as intended and if there were any existing problems in said iteration.

## Tools and Techniques

The tools and techniques used to develop this project were Visual Studio 2019 with C# and Unity 2019.4.11f. Visual Studio is an integrated development environment (IDE) created by Microsoft Windows. (therealjohn, 2019)

It consists of a code editor, debugger, GUI design tool and database schema. Common uses for this tool are creating programs, web applications, web services, console apps, scripts, etc. It supports C#, an object-oriented programming language that is based on Java and C++. (Bolton, 2019)

Unity is a cross platform game engine developed by Unity technologies and released by Apple Inc. (Matney, 2017) It offers most of its functionality under a free personal or educational license for individuals/teams earning under $100,000 per annum; I chose to use the personal licensing model.

Unity comes with the optional extension of unity packages that enhance workflow. I added the input system, post processing, test framework, timeline, Unity UI and Visual Studio Code editor packages. The post processing pack allows for higher quality graphics with a relatively low stress placed on the GPU. The other packages are required for basic aspects of a game (e.g., user interface, testing, scripting, etc).

## Ethical Consideration

For the sake of ethical consideration, the video game will not bring about harm to any player or their personal computer. No personal information will be collected from the players who install the game. The charts and reviews of the steam store are public and free to use for research purposes. Lastly all assets (3d models, textures, audio, sound effects, graphics) were obtained from public domain sources and or purchased from the unity asset store. The appropriate license for the Unity software was legally obtained and pirated software was not used during the development of the application.

## Requirement Analysis

During the analysis stage the compiled data was used to draw conclusions on what the target audience liked and disliked about the related games and was able to apply it to this project.

For the design stage, popular games in the horror genre such as The Last of Us and Resident Evil, were used as a point of reference and inspiration for the game level design but with a metropolitan twist on it; the menu UI was made to be simple and effective. Game models were chosen in terms of how closely they tied into the overall vision and direction of a post-apocalyptic horror setting.

## Requirements Specifications

This section specifies the functional and non-functional requirements which are the requirements that were made essential for the system to be identified as successful and the requirements that improve the caliber of the project respectively.

### Functional Requirement Specifications

The functional specifications specify the intentions of this video game application from the perspective of the user. It specifically identifies the functions that the game should perform in order to be classified as a success.

### Table 2.1 Functional Requirement Specifications

|  |  |  |
| --- | --- | --- |
| **Req.**  **No.** | **Description** | **Type** |
| FR-101 | The game should have a working FPS controller | Functional |
| FR-102 | The game shall include a user interface. | Functional |
| FR-103 | The game shall have a working inventory system. | Functional |
| FR-104 | The game should have working enemy AI to combat | Functional |
| FR-105 | The game should have a working save/load  functionality | Functional |

### Non-Functional Requirement Specifications

The non-functional specifications consist of qualities that make the game efficient, enjoyable, safe and reliable to use; such specifications are what players base their satisfaction with the game on.

**Table 2.2 Non-Functional Requirement Specifications**

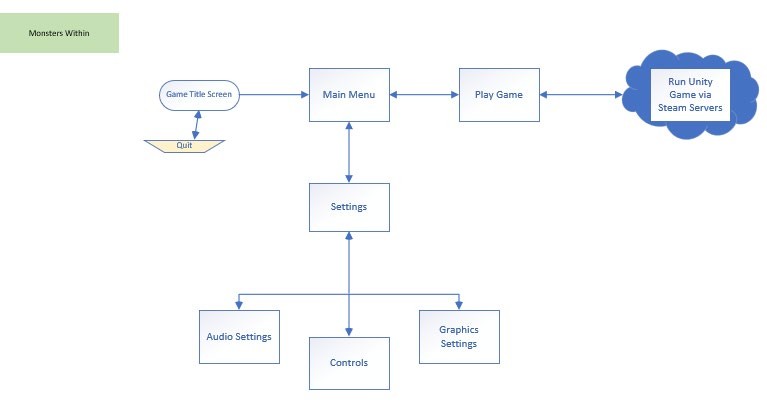
|  |  |  |
| --- | --- | --- |
| **Req.**  **No.** | **Description** | **Type** |
| NFR-  201 | Interactable objects should be examinable in a full  3D view for more in-depth exploration | Non-  Functional |
| NFR-  202 | The game should be able to run on a less powerful  system | Performance |

|  |  |  |
| --- | --- | --- |
| NFR-  203 | The UI for the menu should be simple and intuitive | Usability |
| NFR- 204 | The game environment should have graphics that can vary from AAA quality to indie titles based on  the user’s preferred settings | Usability |
| NFR-  205 | The game should accept input from controls | Configuration |
| NFR-  206 | The game’s loading screen should give the player  useful tips about the game | Reliability |

## System Design

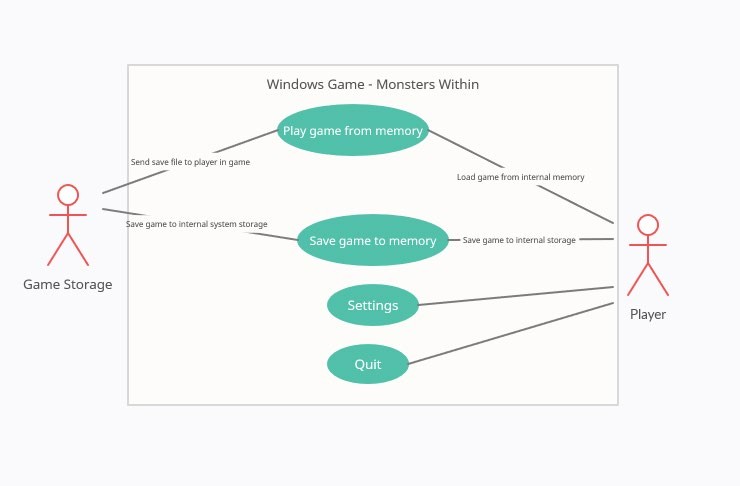
The following sections of this chapter illustrates the game’s design description and application architecture through the use of UML diagrams.

## Application Architecture



**Fig. 3.1 Application Architecture diagram (player perspective)**

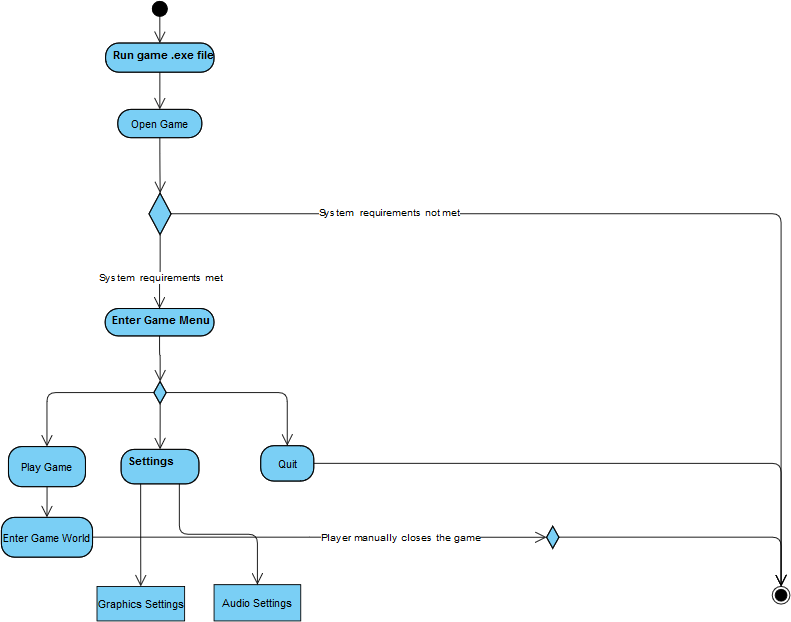
## Use Case



**Fig. 3.2 Use Case diagram**

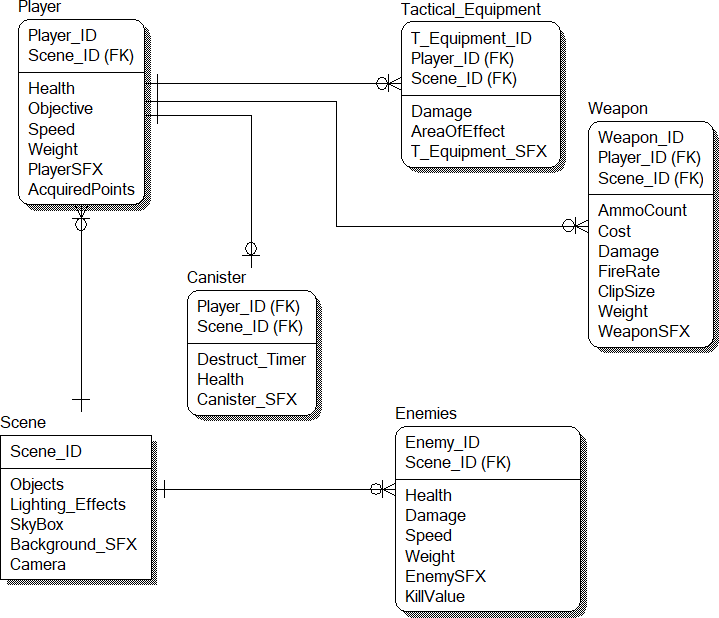
## Activity Diagrams

An activity diagram is a model that shows the process of a task or action from a use case.



**Fig. 3.3 Activity Diagram**

## Entity-Relationship Diagram (ERD)



**Fig. 3.4 Entity Relationship Diagram**

## User Interface Design

The following images below illustrates the game’s UI in all aspects of interaction ranging from the menus to gameplay.



### Fig. 3.5 Main Menu



**Fig. 3.6 Load Screen**



### Fig. 3.7 Gameplay



**Fig. 3.8 Inventory**



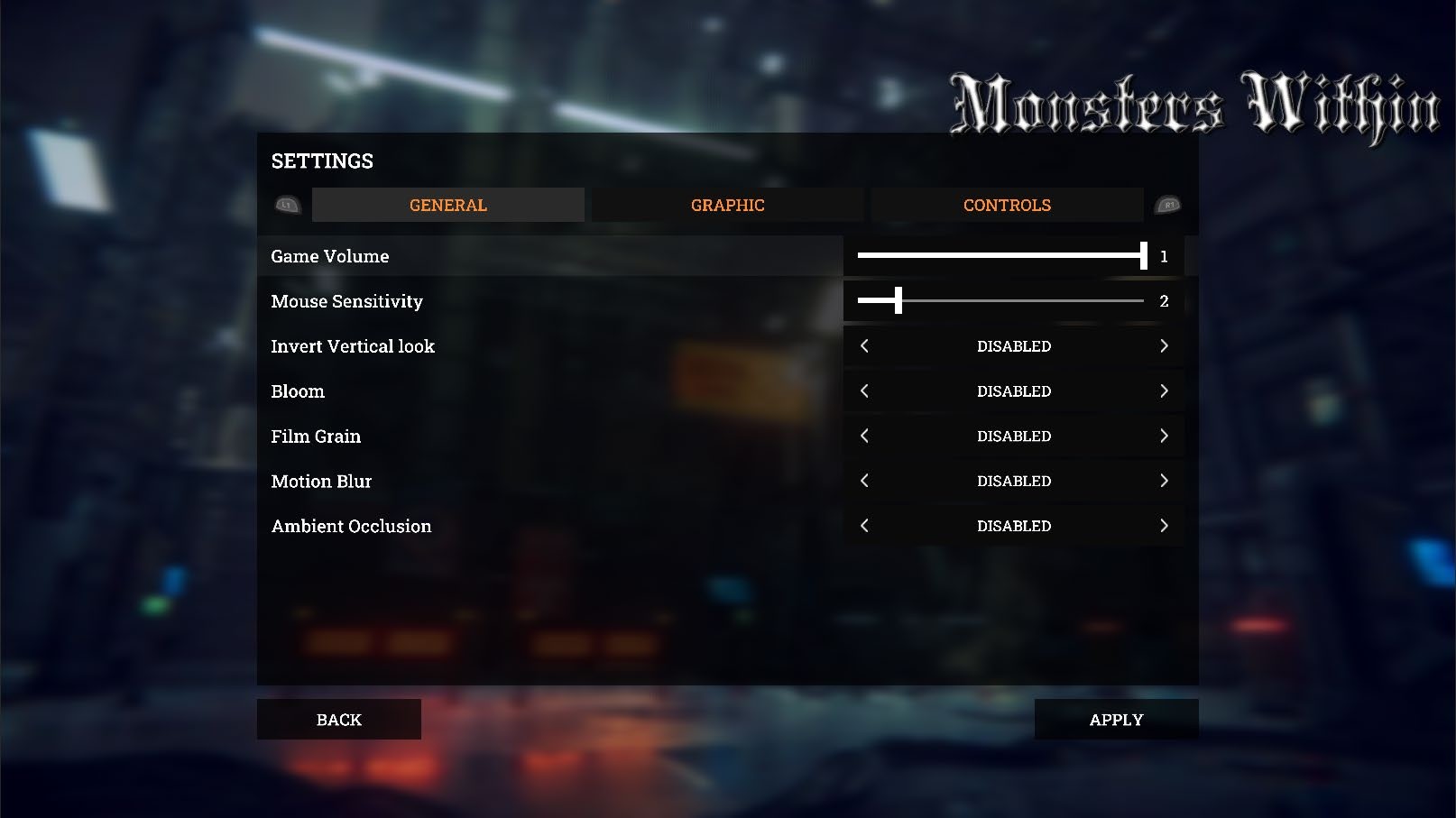
### Fig. 3.9 Enemy



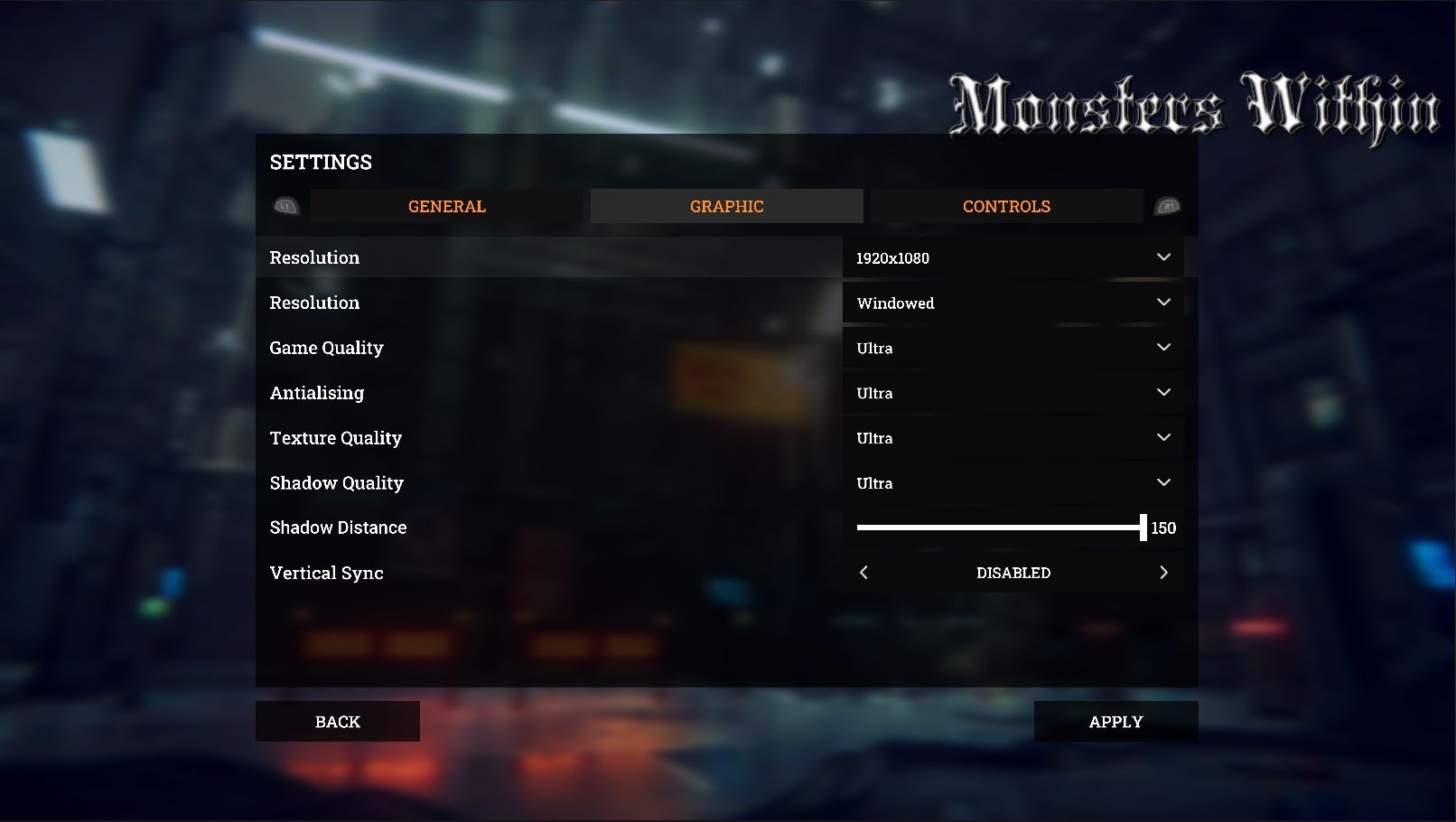
**Fig. 3.10 Pause Menu**



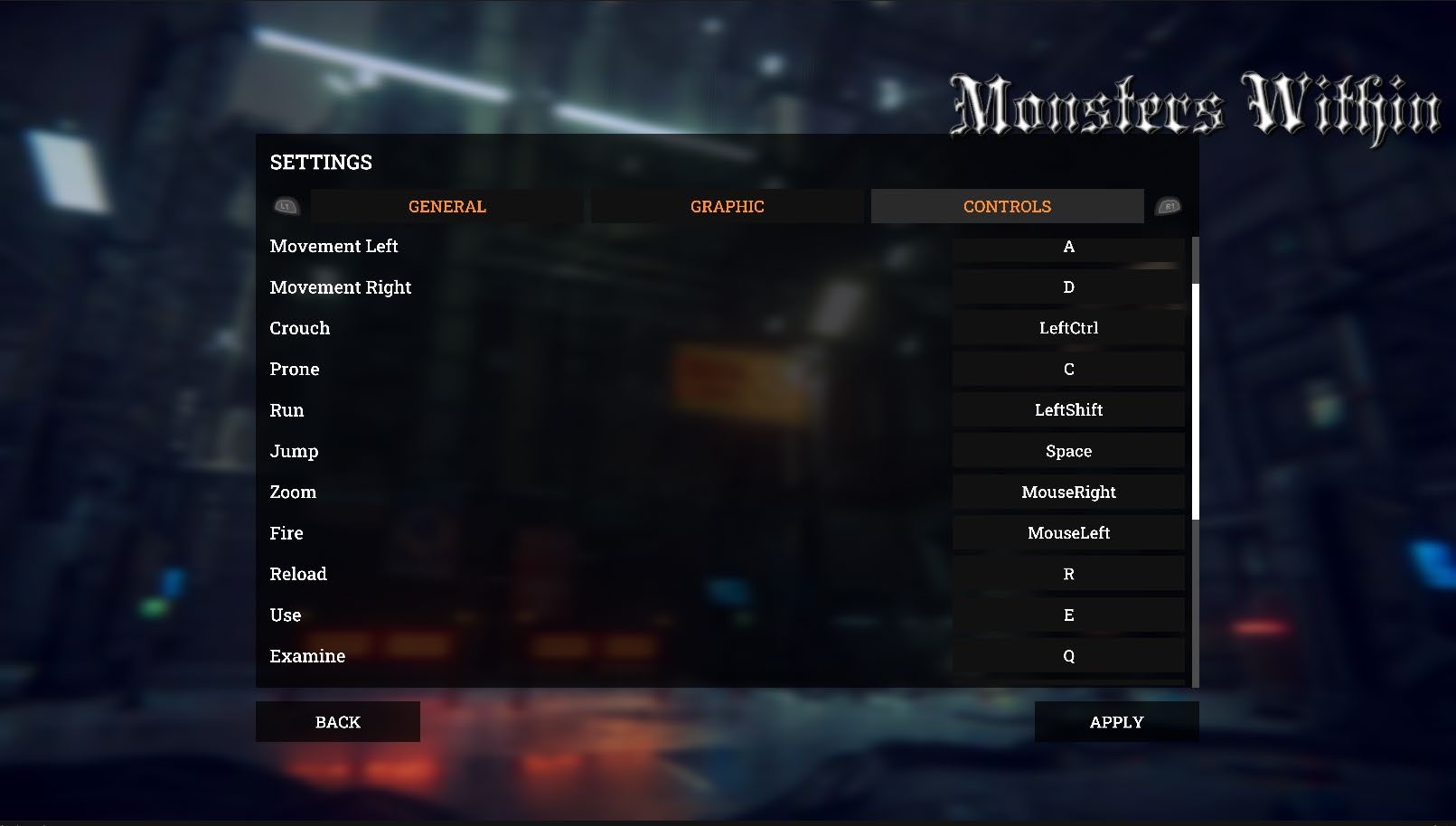
### Fig. 3.11 Load/Save Game file



**Fig. 3.12 General Settings**



### Fig. 3.13 Graphics Settings



**Fig. 3.14 Controls Settings**



### Fig. 3.15 Game environment



**Fig. 3.16 Boss nest (objective)**

## Summary

This concludes the chapter discussing the data gathering the requirements, analysis and design of the game. All data had been meticulously examined and the data gathering techniques were fully applied to building the structure of the game.

# CHAPTER 4: IMPLEMENTATION AND TESTING

## Overview

This chapter goes through the main features of the game and explains the process of the implementation that was carried out. The method of testing in the game engine will be discussed; an overview of how the game was tested within the Unity game engine will be given as well as how and why certain bugs came up and how they were resolved and or mitigated. A report of the summary of errors will be made. Lastly, the user guide/controls for the will be made available at the end of this chapter.

## Main Features

The game consists of a well detailed game environment, an inventory system, enemy AI coupled for root motion that operates based on the nav mesh/ waypoints, observable objects, FPS controller and menu. The following subsections describe how they were implemented.

## Implementation Problems

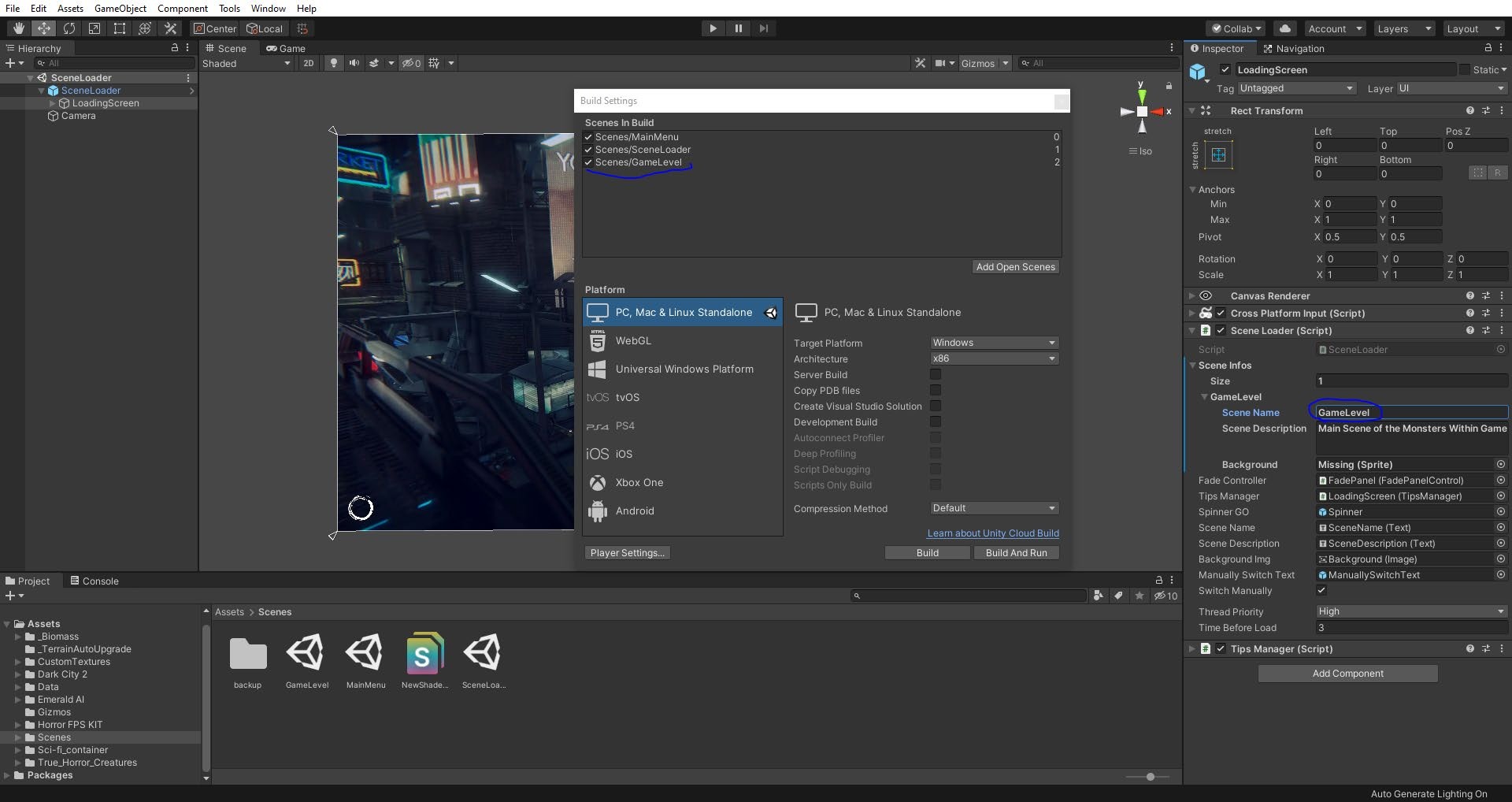
During the game’s development process several bugs were encountered. However, it was to be expected as in the process of creating the game it was inevitable to run into problems. While designing the game environment, the nests were pitch black and permeable (meaning that players could walk straight through the object). The enemy AI had problems detecting the presence of the player regardless of the distance between them. Enemy AI didn’t receive damage from gunshots. Enemy AI was phasing through all objects and did not navigate the environment realistically (they phased through all obstacles in their path). The menu buttons didn’t take the player they were intended to.

## Overcoming Implementation Problems

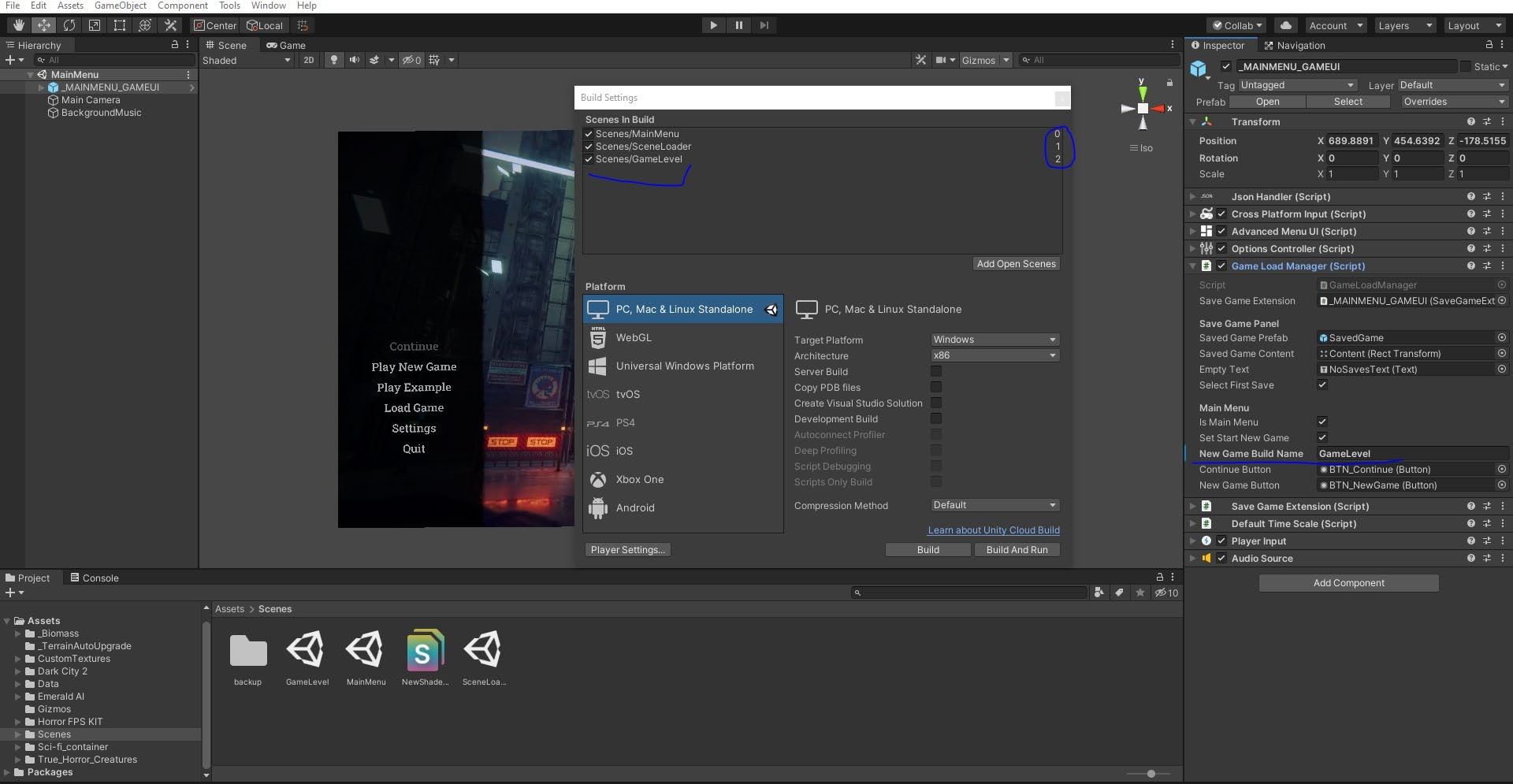
During the development of this game, overcoming problems that arose during implementation was done through the observations of components attached to the game objects for logical errors and debugging scripts in situations where the problem

stemmed from coding. The majority of bugs were fixed by searching online for people who have experienced the same or similar issues and checking for the solution on the Unity forums. YouTube Tutorials, unity documentation and open-source projects were used as a reference point for the scripts implemented in the game; this resulted in a slower pace of development but fewer bugs in scripts as a result.

The only errors stemming from the scripts were the menu navigations. The cause of the inaccuracy in menu navigation with the buttons was the improper naming conventions used in “Load Scene” Manager.



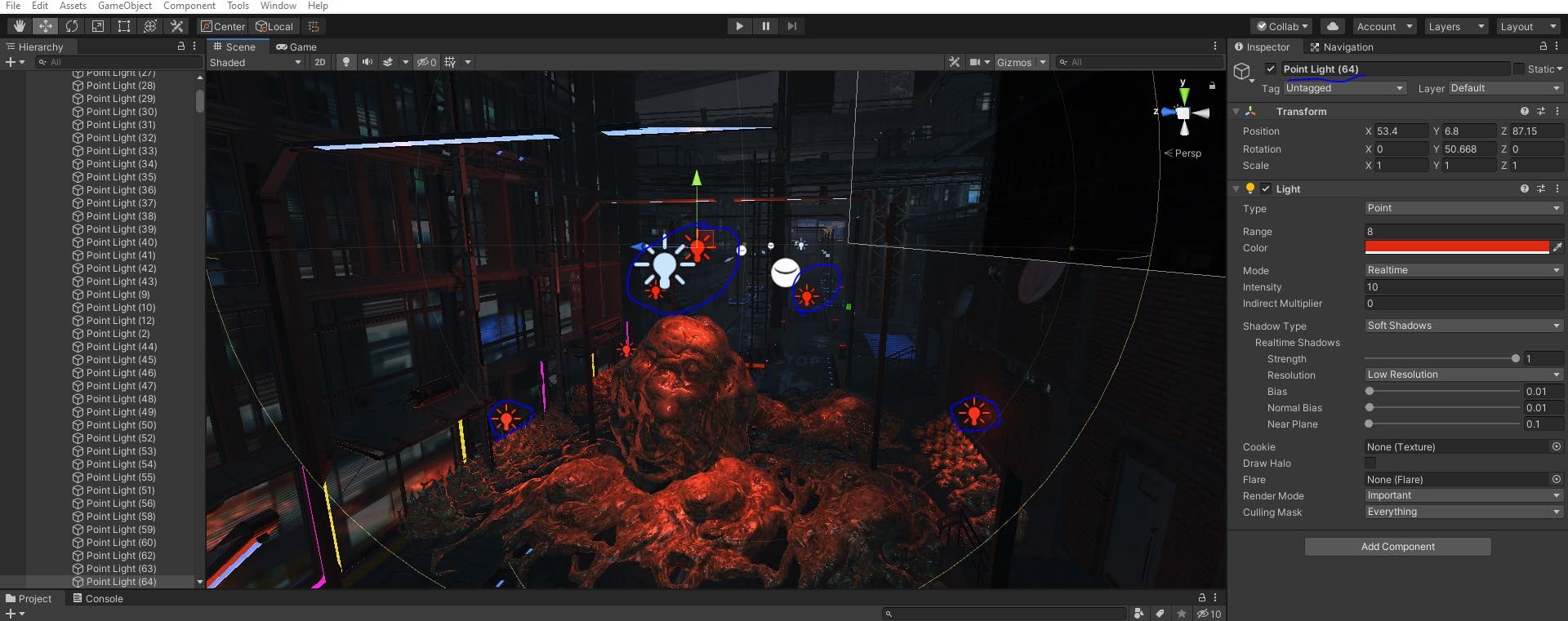
### Fig.4.1 Scene Loader Manager



**Fig.4.2 Scene Build Settings**

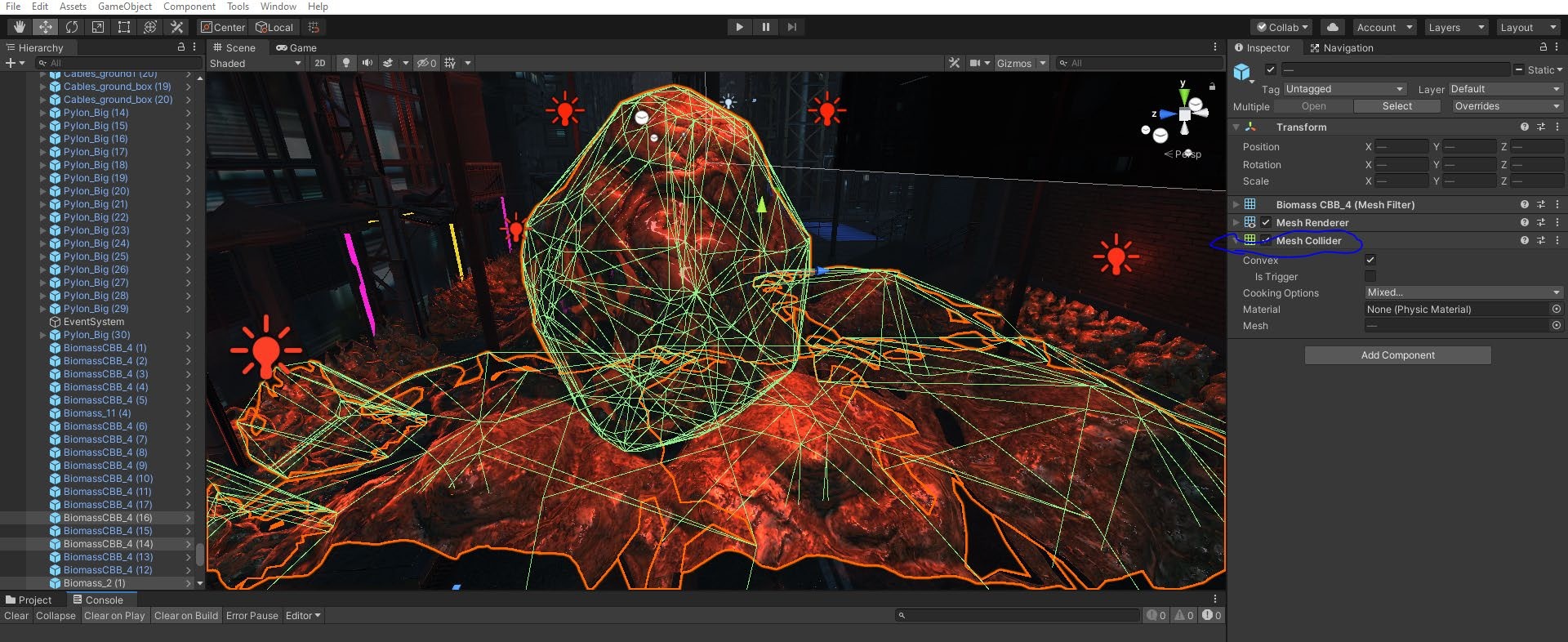
Other implementation problems came from my lack of understanding and experience with the game engine’s components.

The nests appeared pitch black because the textures weren’t applied and there were no point lights used along with the biomass asset (the 3D model used to create the gooey alien-like nests).



### Fig.4.3 Point Lights

The nests allowed characters to pass through because they weren’t marked as static objects on the nav mesh and no appropriate mesh colliders were implemented.



### Fig.4.4 Mesh Colliders

To resolve the issue of the enemy AI not detecting the player the layers of the player and enemy were readjusted. The player was set to the “Flesh” and “Player body” layers while the enemy AI was set to detect objects on the default water layer; this was corrected and also fixed the problem of them not taking damage.

## Testing

In a software development life cycle, testing is essential for. It helps to provide verification and validation on whether all of our functional requirements are met. This is done by generating test data and feeding it to the application to monitor the behaviour. This helps to identify the vulnerabilities and limitations of our application. If a feature fails to meet expectations it will be isolated from other components and examined.

The following sub-chapters go through features to be tested and document the performance of each component. A test report will also be provided towards the end. The Enemy AI not detecting the player, not receiving damage form the player originated from the same problem which was that the two characters were placed on different layers in the game level and the enemy was not set to detect anything on the player’s level.

## Tests Plans (for System Testing)

The content below outlines the test plan for the *Monsters Within* project.

### Test Identifier:

* + - * + Test Level: Master Test Plan
        + Author’s Name: Ibukunoluwa Michael Adebanjo
        + Author’s Contact Email: [ibukunoluwa3242@bazeuniversity.edu..ng](mailto:ibukunoluwa3242@bazeuniversity.edu.ng)

### Reference

* + - * + Monsters Within Game
        + Work Plane
        + Detailed project documentation
        + Test summary

### Introduction

The *Monsters Within* game’s master test plan is provided here which will verify and validate the functional requirement. Different testing methods such as white and black box testing will be performed.

### Features Tested

The following is a list of the areas to be focused on during testing of the application.

* Saving game progress
* Loading an existing game file
* Menu UI navigation
* General settings
* Graphics settings
* Using control settings
* Player movement (walking, sprinting, crouching and strafing)
* Player inventory system
* Object observation
* FPS weapon controls (aiming, shooting, reloading)
* Player – Enemy Interaction
* Killing the enemy AI

### Features Not Tested

* + - * + The nest objective
        + Nest destruction
        + Level progression

### Approach

The built-in console debuggers in unity and the Visual Studio IDE were used to observe the operations that occurred in the game in each and every game test scenario to see what wasn’t done properly or wasn’t accounted for. Going through the documentation for the Unity game engine and its packages proved to be crucial in solving most of the errors that occurred during the game's implantation.

### Test Deliverables

The deliverables of this test plan are as follows:

* + - * + Test cases
        + Test report
        + Traceability matrix
        + Test results
        + Error report

### APPROVALS

Ibukunoluwa Michael Adebanjo

## Test Suite (for System Testing)

### Test Suites Performed

Testing in the Unity game engine is performed as a black box form of system testing. System testing evaluates the completeness and compliance of the system in accordance with the specified requirements; the system’s functionalities are tested from an end-to- end perspective. In black box testing, the software is tested from the user’s perspective. In relation to Unity, developers have the functionality of running the game in “Play” mode where the game can be operated in the “Game” window to see it from the player’s (users) perspective; errors and or warnings are generated on runtime in the “Console” window if there are any. All test cases are given in Appendix D.

## Test Traceability Matrix (for Unit Testing, Integration Testing, and System Testing)

**Table 4.5 Test Traceability Matrix**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req.** | **Description** | **Priority** | **Test**  **Case** | **Test Date** | **Test Result** |
| FR-101 | The game should have a working FPS controller | High | 1 | 18/10/2020 | Pass |
| FR-102 | The game shall include a user interface. | High | 2 | 08/11/2020 | Pass |
| FR-103 | The game shall have a working inventory system. | Medium | 3 | 09/11/2020 | Pass |
| FR-104 | The game should have working enemy AI to combat | High | 4 | 26/10/2020 | Pass |
| FR-105 | The game should have a working save/load functionality | Medium | 5 | 11/11/2020 | Pass |

## Test Report Summary (for System Testing)

Below is the summary of the tests that were carried out

**Table 4.51 Test Report Summary**

|  |  |
| --- | --- |
| Summary of executed tests | Results |
| Number of functions tested: | 5 |
| The number of functions not tested: | 0 |
| The number of tests passed: | 5 |
| The number of tests failed: | 0 |
| Percentage of tests passed: | 100% |
| Percentage of tests failed: | 0% |

## Error Reports and Corrections

* Improper booting of the loading screen (Runtime error). This was caused by an improper setup of the main camera in the Scene Loader scene.
* Player falling through the ground (Logical error). This was caused by the inaccurate placement of the player character’s feet in relation to the terrain.
* Player moving through static objects (Logical error). This was a result of a lack of knowledge of how game objects worked; as it was with all other issues other than the two afore mentioned errors.

## User Guide

The user guide gives players a control scheme for how to operate the player character in the game environment as well as interact with it. The menu’s UI was designed to be simple and extremely user friendly with all the buttons being accurately labelled and easy to understand what their function is. Player’s start the game off entering the main menu where they have the choice of quitting the game, editing the settings, or going to play a new game or continue from a save file. All forms of navigation are executed with button clicks as the user input. The outline of the controls and the instructions pertaining to the navigation of the menu’s UI will be located in Appendix E.

## Summary

This chapter clearly established the main features of the game and defines the techniques used to integrate these features. Issues that came up during implementation and testing were thoroughly observed and methods used to resolve these issues were mentioned. The test plans, procedures and findings of the tests were stated along with a user guide. Bugs and errors spotted during testing were documented and the appropriate corrections were made were possible. The following chapter entails the conclusion, overall discussion and recommendations for the video game project.

# CHAPTER 5: DISCUSSION, CONCLUSION, AND RECOMMENDATIONS

## Overview

This is the last chapter of the thesis and as such, it will contain the overall assessment of this project, its challenges and limitations encountered during its development life cycle and information on how the project can be subsequently improved in the future. Lastly, based on the findings of the project, suggestions regarding the project’s direction will be examined.

## Objective Assessment

Despite the project not aligning fully with the initial plan, all functional requirements of the video game project were met. The video game has a solid foundation that will easily allow for subsequent improvements and additional features to be implemented into the game’s architecture. This game is currently ready to be placed in the early access stages of video game release on the steam store where players can experience the game and share their input what they think of it in its current state, improvements they believe would be beneficial to the game as well as any other ideas they have.

## Limitations and Challenges

The major limitations hindering the full implementation of this project was the budget, manpower, time and experience.

Games generally tend to be developed by teams consisting of members specializing in a specific field in relation to game development (i.e., game level designers, digital artists, programmers, etc.). This teamwork is an essential aspect of game development which can make or break the final product; the nature of this project led to me having to develop this on my own. During the development process I had to fulfill the roles of each team member.

The second limitation was time; resuming in the last week of September and getting my new topic approved in October, I had 3.5 months to develop the game. As a beginner in game development who was also working alone, this proved to be problematic throughout the stages of development. For context, Among Us, a popular 2D mystery

solving indie game which is simplistic in nature, had 3 professionals working on the project. It took 2 months to develop, 3 months to fix all bugs and get out of the beta stages and 6 months to craft new maps. This data was obtained from their developer logs. (InnerslothDevs, 2020)

The Unity game engine was designed to make things easier for small time studios/individuals to make their own games without the need for a large studio base. While it does make game development more accessible and easier, it is still better worked on with a team. I initially had the misconception that I only needed my coding skills and everything else would be a drag and drop feature; I was wrong.

There is a heavy requirement for knowledge and expertise in graphics, UI, game physics, navigation (path finding), game level design and animations. As a result, a good amount of time was spent learning about these fields as opposed to developing the game in full.

Designing the game environment which was placed as a top priority based on the conducted observations of the success of other games, took a month which left less time for fleshing out full features for other aspects of the game.

The final challenge to overcome was the budget that game studios typically have for a project. This obstacle was comparatively easier to overcome as there were several sources for free-to-use audio, graphics and prefabs. Models still had to be scripted, animated and put together though.

## Future Enhancements

This project has plenty of room to grow and subsequent updates with extensions of the already built-in features will be implemented in the near future. The game’s first priority for future development will be to integrate more weapons and story driven items for the player to interact with.

While the UI for the main menu, pause menu and settings has been completed, there are often minor bugs in the loading screen that delays the startup of the game. These bugs will be fixed and the game as a whole will be more optimized so that the requirement for running the game will be lower and hence, accessible to more players. There are surprisingly few bugs in the project in regards to gameplay, so there will be a focus on adding other enemy types in the event that the game generates enough revenue for the purchase of high-quality enemy assets.

Lastly, there will be new maps designed for the game to tie into the overall story and keep things fresh for the players to avoid the risk of having them grow tired of the game play loop.

## Recommendations

Upon drawing conclusions from the findings of this project, it has been established that a deep understanding of Unity’s documentation and the use of its pre-built packaging system is an essential core aspect of developing games in the most efficient manner there has been a light shed on the fact that there is an ever-growing demand for games in modern society as well as a lack of paced FPS games in the horror genre.

## Summary

This chapter brings an end to the project documentation. It described in great detail, the objectives of the project, its scope, risk assessments, the requirements, analysis and design; the methodologies used during its development life cycle and how it was both implemented and tested for real life usage. Lastly improvements that could be made in the time forthcoming was discussed as well as recommendations for the gaming community in Nigeria.

# REFERENCES

Bolton, D. (2019). *All about the C# Programming Language*. [online] ThoughtCo. Available at: https://[www.thoughtco.com/all-about-the-c-programming-language-](http://www.thoughtco.com/all-about-the-c-programming-language-) 958330. [Accessed 6 Jan. 2021]

History.com Editors (2018). *Video Game History*. [online] HISTORY. Available at: https://[www.history.com/topics/inventions/history-of-video-games](http://www.history.com/topics/inventions/history-of-video-games) [Accessed 31 Oct. 2020].

Reed, Dr.K. (2017). *12 Surprising Health Benefits of Playing Video Games*. [online] Positive Health Wellness. Available at: https:[//w](http://www.positivehealthwellness.com/fitness/12-surprising-health-benefits-playing-)ww[.positivehealthwellness.com/fitness/12-surprising-health-benefits-playing-](http://www.positivehealthwellness.com/fitness/12-surprising-health-benefits-playing-) video-games/ [Accessed 11 Nov. 2020].

G, D. (2020). *43+ Video Games Industry Statistics, Facts, and Trends for 2020*. [online] TechJury. Available at: https://techjury.net/blog/video-games-industry- statistics/ [Accessed 13 Dec. 2020].

Farell, N., 2021. Best Call of Duty games ranked. Available at: https://blog.gfuel.com/best-call-of-duty-games-ranked [Accessed 4 Jan 2021].

Devs, I. (2020). *Among Us 2*. [online] itch.io. Available at: https://innersloth.itch.io/among-us/devlog/171026/among-us-2 [Accessed 12 Dec. 2020].

Matney, L. (2017). *With new realities to build, Unity positioned to become tech giant*. [online] TechCrunch. Available at: https://techcrunch.com/2017/05/25/with-new- realities-to-build-unity-positioned-to-become-tech-giant/ [Accessed 20 Nov. 2020].

McIntyre, M. (2020). *The 15 Best Call of Duty Games of All Time Ranked*. [online] Gaming Gorilla. Available at: https://gaminggorilla.com/best-call-of-duty-games/ [Accessed 4 Jan. 2021].

topictrends.org. (n.d.). *Reasons Why Entertainment Is Important*. [online] Available at: <http://topictrends.org/reasons-why-entertainment-is-important/>[Accessed 4 Jan. 2021].

Resnick, B. (2020). *How are Americans coping with the pandemic? Not well, says CDC data.* [online] Vox. Available at: https://[www.vox.com/science-and-](http://www.vox.com/science-and-) health/2020/5/29/21274495/pandemic-cdc-mental-health [Accessed 4 Jan. 2021].

Birdman, B. (2016). *Top 10 Call of Duty Games of All Time - YouTube*. [online] [www.youtube.com.](http://www.youtube.com/) Available at: https://[www.youtube.com/watch?v=71uv\_MBhgNM&list=PLRVTZK-](http://www.youtube.com/watch?v=71uv_MBhgNM&list=PLRVTZK-) ep0cDLIoqh8teVKVJoeIh69OBT&index=10 [Accessed 3 Jan. 2021].

Sanchez, M. and Claiborn, S. (2020). *The Best Call of Duty Games, Ranked - IGN*. [online] [www.ign.com.](http://www.ign.com/) Available at: https://[www.ign.com/articles/2019/06/01/the-best-](http://www.ign.com/articles/2019/06/01/the-best-) call-of-duty-games-ranked [Accessed 1 Jan. 2021].

Technologies, U. (n.d.). *Powerful 2D, 3D, VR, & AR software for cross-platform development of games and mobile apps.* [online] store.unity.com. Available at: https://store.unity.com/#plans-business. [Accessed 09 Jan. 2021]

Shamil, Prof.F.R. (2020). *Iterative Model in software engineering advantages disadvantages*. [online] T4Tutorials.com. Available at: https://t4tutorials.com/what-is- iterative-model-software-engineering/ [Accessed 2 Jan. 2021].

therealjohn (2019). *Visual Studio Tools for Unity*. [online] docs.microsoft.com. Available at: https://docs.microsoft.com/en-us/visualstudio/gamedev/unity/get- started/visual-studio-tools-for-unity?view=vs-2019 [Accessed 3 Jan. 2021].

**APPENDICES**

**Appendix A - Project Document**

The project’s documentation for the analysis, design and implementation of the

*Monsters Within* video game projects:

IN-DEPTH PROJECT DOCUMENTATION

**Full Candidate Name:** Ibukunoluwa Michael Adebanjo

**Student ID:** BU/18C/IT/3242

### Monsters Within

A 3D FPS windows game.

***Course of Study:*** B.Sc. Software Engineering

#### Objectives and Background Study

Entertainment can be described as any kind of activity that is capable of drawing the interest of an audience and maintaining their attention throughout the process. It is a form of amusement or diversion from day to day activities that bring about pleasure for the participating audience. One of the more prevalent forms of entertainment in today’s modern society is video games. With the current global state of affairs leaving many people unsatisfied and often depressed, no more than ever there is a need for entertainment that is both delightful and interactive.

*Monsters Within* is a 3D FPS game that will be made available on the Windows platform. It will be designed to fit the horror genre of games and offer players gruesome looking enemies, boots-on-the-ground gameplay and an enthralling environment full of mystery and items that tie into the overall story of the game which will be told in a puzzle like manner.

#### Statement of the Problem:

Now more so than ever, people in Nigeria (and globally as a matter of fact) have become more anxious and depressed as studies have shown.

People have a natural urge to be entertained in some way whether it be a physical or mental challenge. In regards to this matter, psychologists cite “Lucifer’s Principle” which states that people often need the illusion of control in their lives. When playing video games, there is conflict and a challenge to overcome. When the challenge in question has been beaten people receive a dopamine hit in their brains that gives them a sense of accomplishment. This phenomenon is similar to when people are able to lift a certain amount of weight in the gym or successfully prepare a decent meal.

Without video games or any sort of entertainment for that matter, life becomes boring and somewhat unsatisfying.

## Appendix B – Observation Reports

**Steam Reviews for Little Nightmares** [Used as a basis for what to model story-telling and game atmosphere after]:

*Overall Status*: “Very positive”

The chart below shows the total number of positive and negative reviews in the subsequent years after the game’s initial release date till present day.



### Fig.21.0

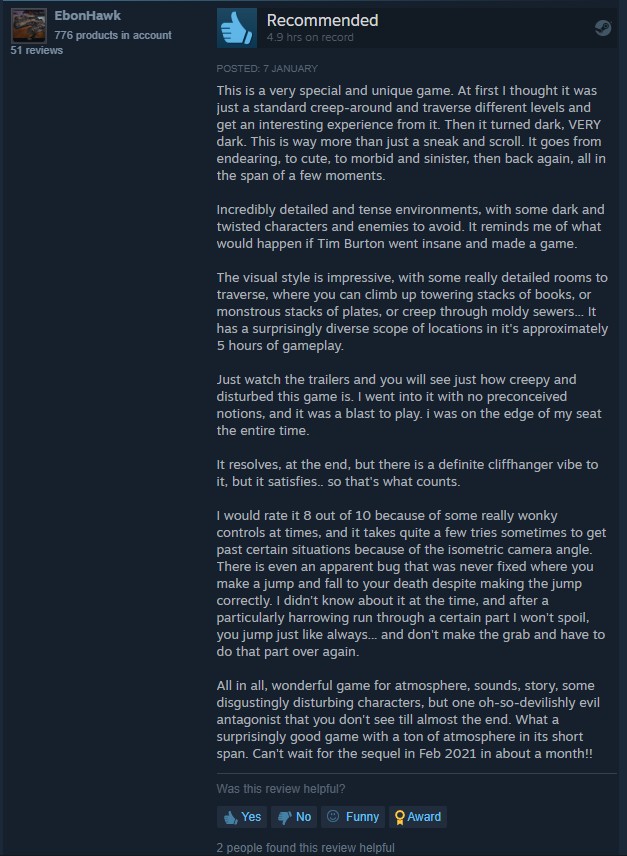
The following images are handpicked screencaps that were used as the main basis of judgement on player reviews based on the fact that they, by the community themselves, were voted as the most helpful and were also the most liked in both positive and negative reviews throughout the lifespan of the game’s release.

The consensus for positive reviews praised the game for its gritty yet beautifully crafted atmosphere that clearly showed attention to detail. The game environment helped paint a picture of the events that conspired in the game’s world and leaves the story fragmented so that the player is tasked with trying to put together pieces of the puzzle that is the narrative the game developers were trying to convey; reviews such as this to back up this claim that it was a pleasurable experience.



### Fig.21.1

As such, the *Monsters Within* game project included a visually appealing and well detailed environment to enhance the player’s gaming experience in immersion and puzzle-like storytelling.



### Fig.21.2



**Fig.21.3**

The negative reviews of Little Nightmares generally revolve around the controls that often feel wonky due to the game’s isometric camera angle and hitboxes from boss grab moves. *Monsters Within* will be designed in the first-person perspective and will thus be able to avoid such problems.

### Fig.21.4

**Steam Reviews for Sekiro Shadows Die Twice** [Used as a basis for what to model story-telling and game atmosphere after]:

Overall Status: “Very positive”

The chart below shows the total number of positive and negative reviews in the subsequent years after the game’s initial release date till present day.

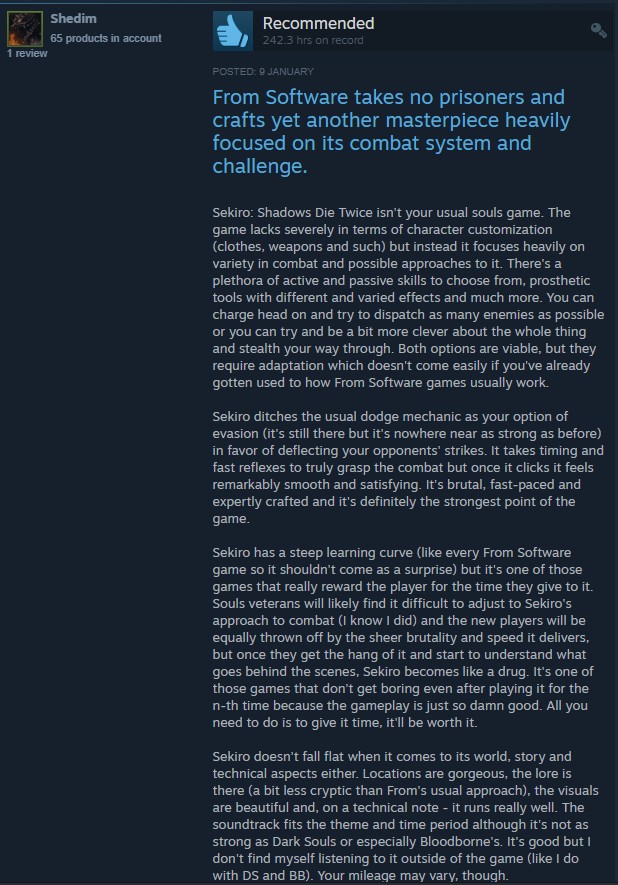


### Fig.22.0

The general outlook for positive reviews praised the game for its challenging combat, beautiful atmosphere and yet again, puzzle-like story telling through the environment, enemies and in-game items. As such, the enemies and in game items will be crafted in a manner that ties them into the overall narrative of the game.



### Fig.22.1



**Fig.22.2**

The negative reviews had a consensus of the game’s difficulty being suitable for players who were seeking a more casual and highlighted that the choice of gameplay would only cater to a specific demographic of customers that absolutely loved being challenged in terms of skill.

In response to this, the enemies will remain intelligent and do relatively significant damage to players, they won’t be made so difficult as to have players be forced to memorize attack patterns and build muscle memory.



### Fig.22.3

**Steam Reviews for COD Franchise games** [Used as a basis for what to model the game mechanics after]:

Overall Status: “Mixed”

There are a total of 16 main Call of Duty games and as such, I decided to conduct an observation on which of the titles the community by large, agreed was the best and

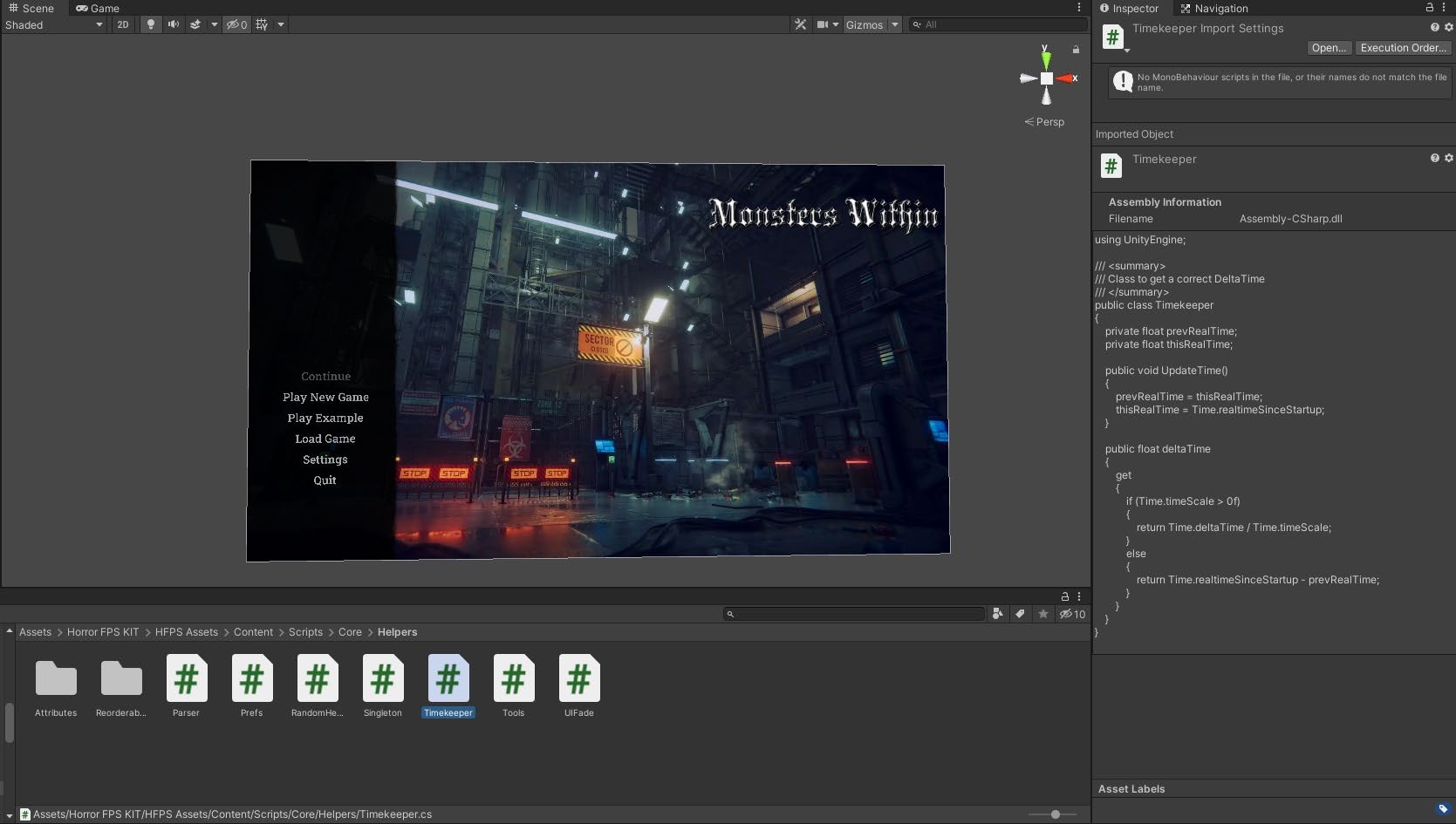
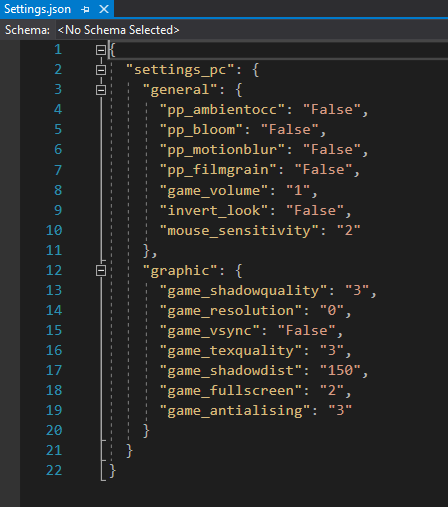
worst in the series as well as why they gave each their positions in this ranking. With reference to popular mainstream video game blogs and videos from respectable members of the community sharing their opinions, it can be concluded that Call of Duty 4: Modern Warfare, Call of Duty: Modern Warfare 2 and Call of Duty Black Ops 2 were the most enjoyed titles.

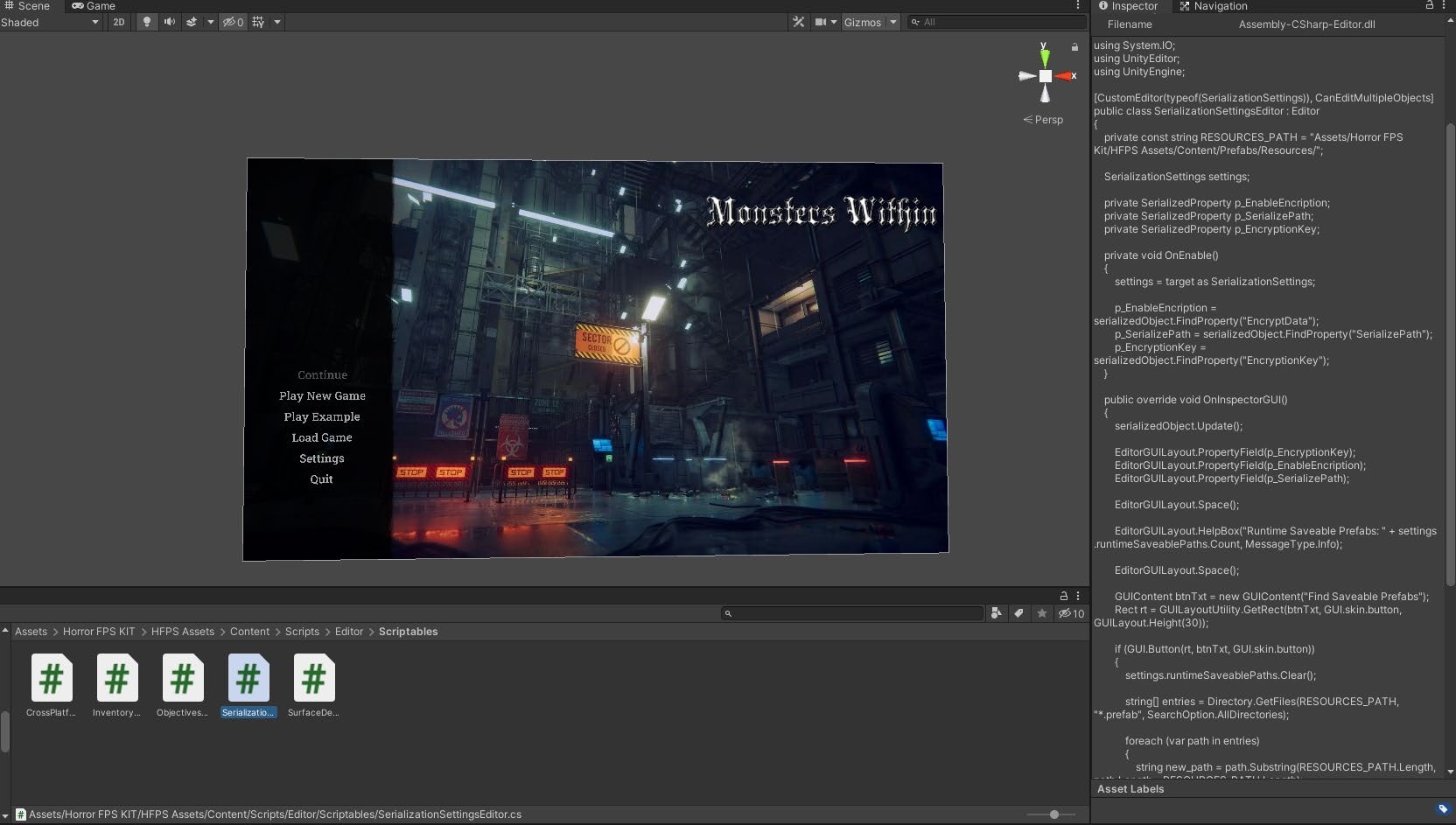
This was largely due to the maps being fun to traverse, and the controls being most precise and engaging in terms of combat. A common theme among the most disliked being Call of Duty Infinite Warfare and Call of Duty Advanced Warfare. The main complaints were that these titles were, in terms of gameplay, too gimmicky (characters having jetpacks and wall running) and unbalanced (advantages were given to players who spent more money on the game via supply drops which ultimately led to many players leaving the game altogether).

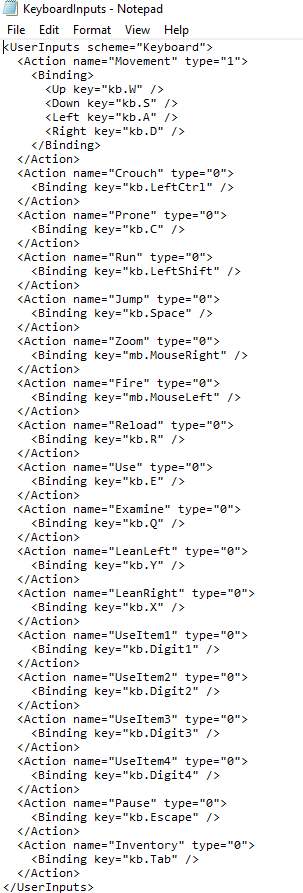
After taking into account all of these findings, the need for maps designed to be suited for combat will be acknowledged and implemented in the video game project of *Monsters Within*; the project is not intended to be a competitive game, hence, supply drops won’t pose to be a problem as it won’t be added to the game.

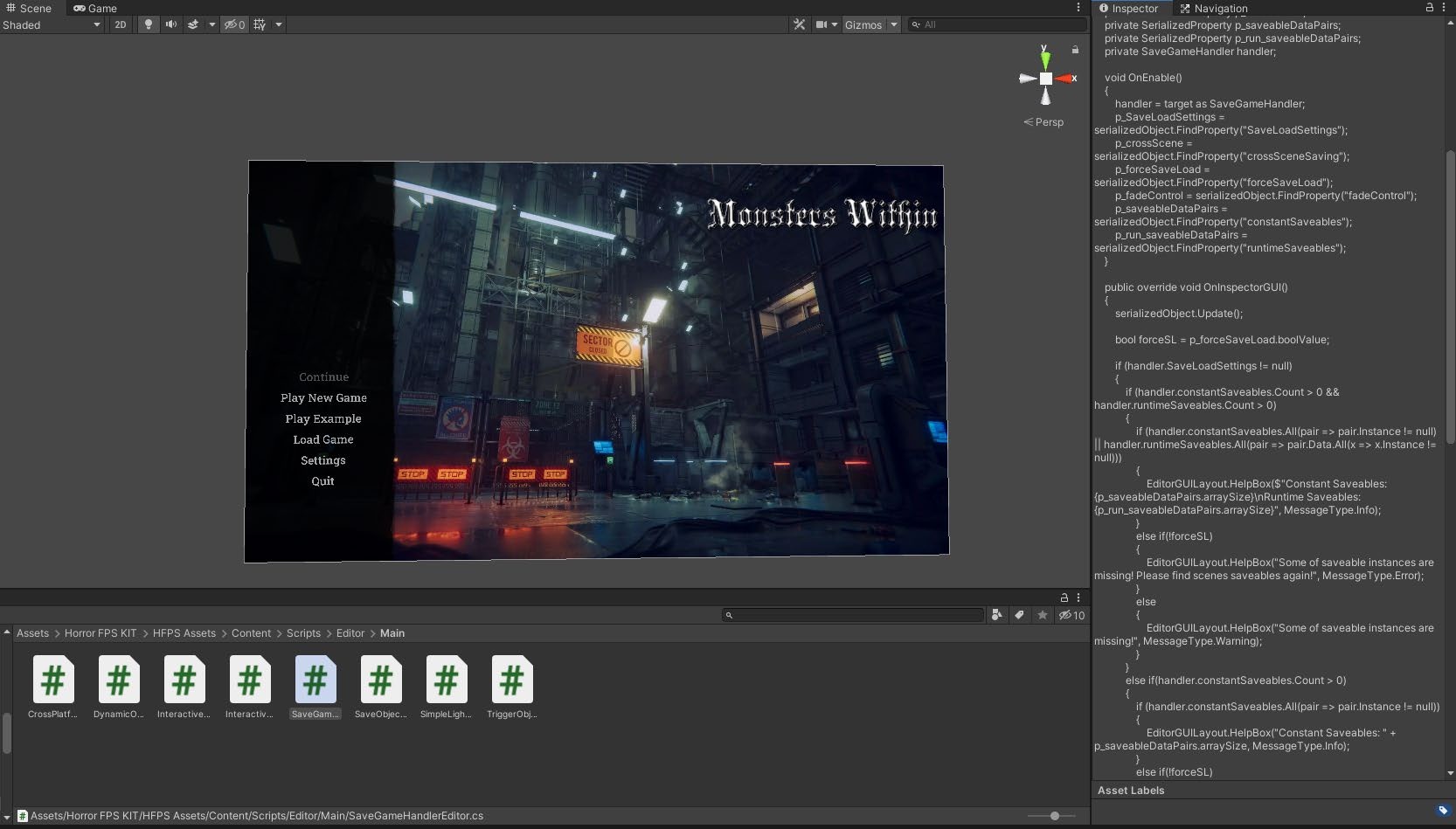
The core mechanics of the gameplay will be kept to a minimalistic standard the fans of the FPS genre are well accustomed to (basic movement, sprinting, crouching, hip firing, aiming and strafing).

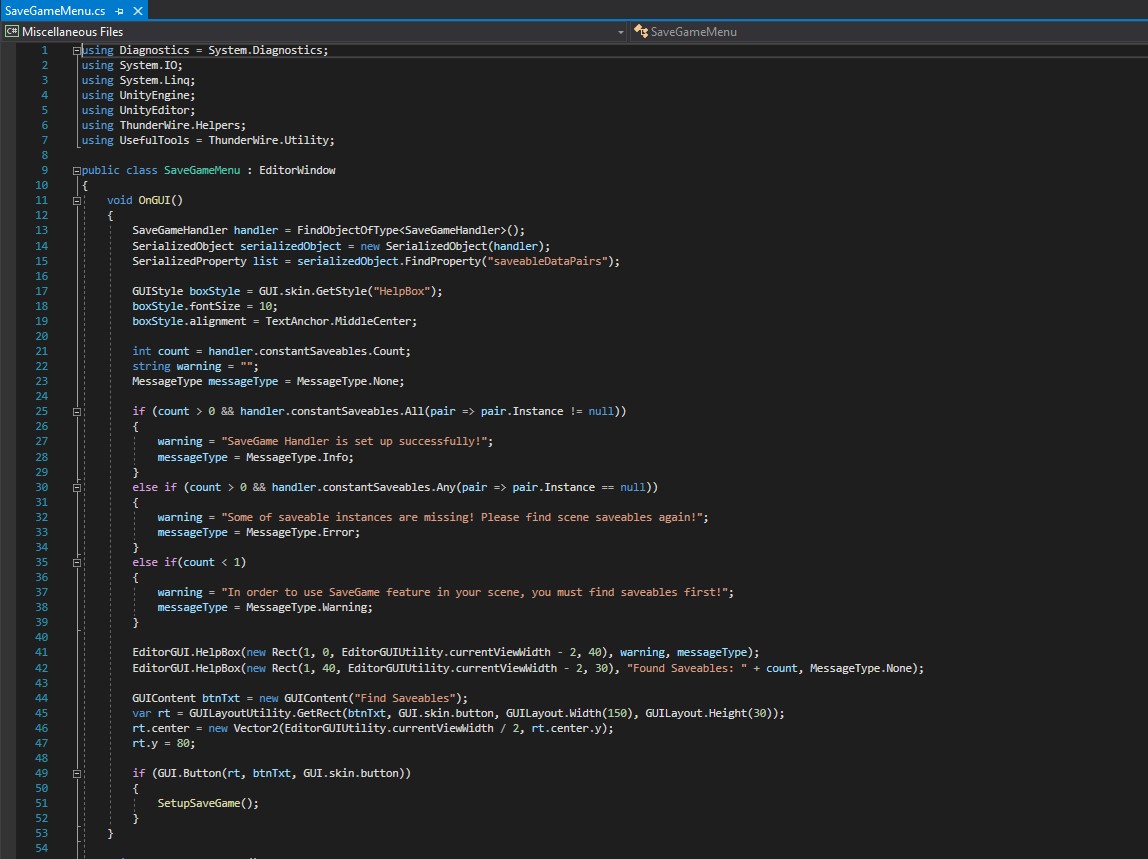
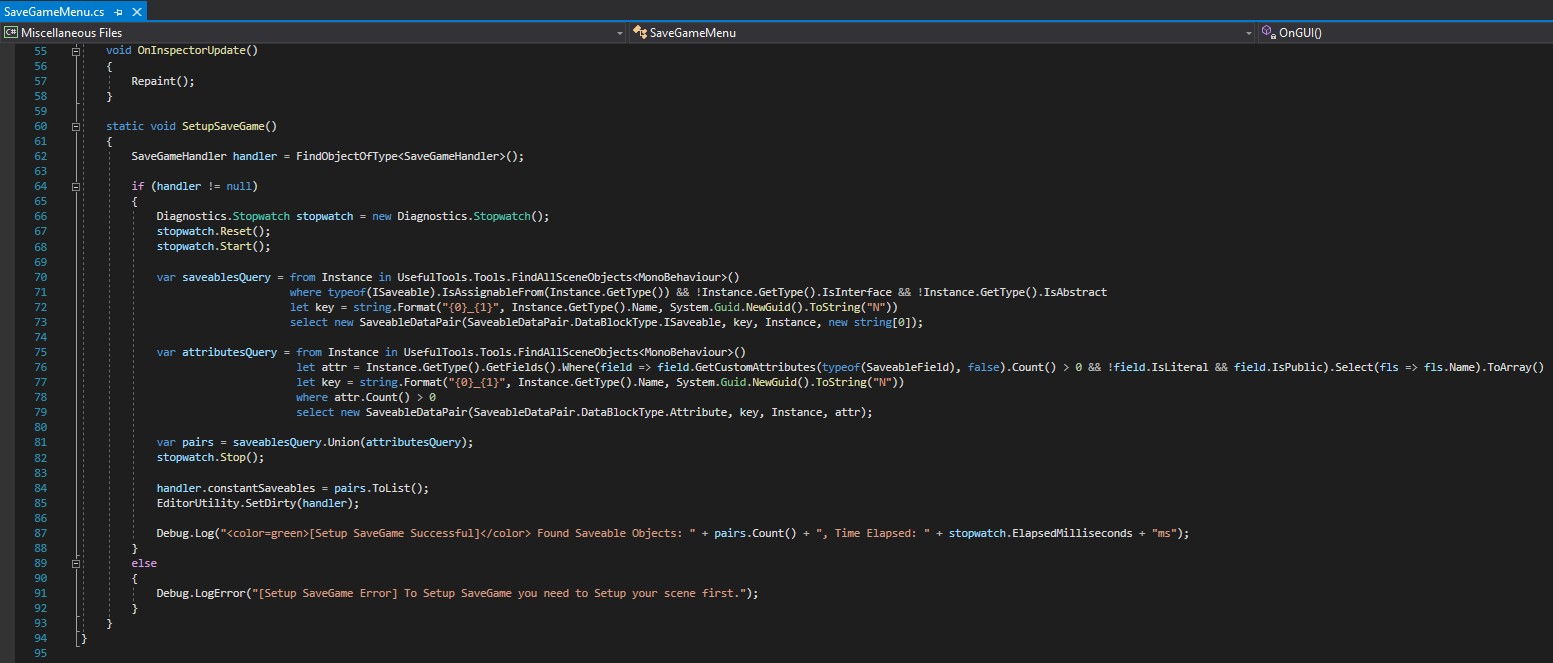
## Appendix C – Source Codes











**Appendix D – Test Cases**

*Note: All tests were performed within the engine as opposed to a functional game build (this is why the test procedures begin with “Run from Scene view”); running it within the engine allows for easier debugging and highlighting where the errors originated from.*

### Table 5.1: Test case TC-001(Menu UI Interaction)

|  |  |
| --- | --- |
| Test Suit ID | R-100 |
| Test Case ID | TC-001 |
| Test case summary | Ensure that the menu UI works as intended |
| Related Requirement | R-XXX |
| Prerequisite | * The game folder must be present in the local storage of the   computer   * The game must be loaded on a computer with a dedicated   graphics card |
| Test Procedure | 1. Run game from Scene View 2. Click the buttons on the menu to see if they take the user to the intended destinations (i.e., settings, the game level, save   files) |
| Test Data | Player Input (Keyboard entries, mouse buttons and movement) |
| Expected Result | The user should not experience crashes when operating with the  main menu. The player should be taken to their desired destination |
| Actual Result | The menu was properly linked to other aspects of the game and no  crashes were experienced |
| Status | Test case succeeded |
| Remarks | The test was a success |
| Created by | Ibukunoluwa Michael Adebanjo |
| Date Created | 4th January, 2021 |
| Executed by | Ibukunoluwa Michael Adebanjo |
| Date of Execution | 4th January 2021 |
| Test Environment | Software: Unity (Game Engine)  Hardware: Eluktronics MAX-17 (CPU: Intel core i7-9750H, GPU: RTX 2060) |

**Table 5.2: Test case TC-002(FPS Mechanics)**

|  |  |
| --- | --- |
| Test Suit ID | R-101 |
| Test Case ID | TC-002 |
| Test case summary | Ascertain the functionalities of the First-person character |
| Related Requirement | R-XXX |
| Prerequisite | * The game folder must be present in the local storage of the   computer   * The game must be loaded on a computer with a dedicated   graphics card |
| Test Procedure | 1. Run game from Scene View 2. Start a new game or load a saved game from the menu 3. Check if the character can walk, run, examine and pickup objects, look around, aim and shoot firearms, reload firearms, crouch and strafe left and right. The afore mentioned actions must also be successfully executed only when the intended corresponding input controls have been used (well-adjusted   key binding)   1. Check if the character can access the item inventory |
| Test Data | Player Input (Keyboard entries, mouse buttons and movement) |
| Expected Result | The player should be able to move and control the character as  described and be able to accurately navigate the game environment |
| Actual Result | The player behaved as expected; weapons and objects were handled as anticipated. The character’s camera was initially wonky giving the appearance that the whole world was swaying left and right. This was fixed in the next build of the game to offer a higher quality  player experience. |
| Status | Test case succeeded (minor bugs) |
| Remarks | The test was a success |
| Created by | Ibukunoluwa Michael Adebanjo |
| Date Created | 24th December, 2020 |
| Executed by | Ibukunoluwa Michael Adebanjo |

|  |  |
| --- | --- |
| Date of Execution | 24th December, 2020 |
| Test Environment | Software: Unity (Game Engine)  Hardware: Eluktronics MAX-17 (CPU: Intel core i7-9750H, GPU: RTX 2060) |

### Table 5.3: Test case TC-003(Saving and Loading Game data)

|  |  |
| --- | --- |
| Test Suit ID | R-102 |
| Test Case ID | TC-003 |
| Test case summary | Check the Save/Load game functionality |
| Related Requirement | R-XXX |
| Prerequisite | * The game folder must be present in the local storage of the   computer   * The game must be loaded on a computer with a dedicated   graphics card |
| Test Procedure | 1. Run game from Scene View 2. Play a new game by clicking the “Play new game” button 3. Progress through the game to a certain extent 4. Go to the pause menu by hitting the ESC button 5. Select the Save game option 6. Go back to the pause menu 7. Click the “load game” button 8. The option to load the saved game should be visible with the   date and time stamp attached |
| Test Data | Game save file data |
| Expected Result | After clicking the save button from the pause menu, the save data should be stored in the game folder path and be loadable for  continuation. The time stamp must also be present and accurate. |
| Actual Result | The save functionality worked on the first attempt without issues and  had an accurate time stamp |
| Status | Test case succeeded |
| Remarks | The test was a success |

|  |  |
| --- | --- |
| Created by | Ibukunoluwa Michael Adebanjo |
| Date Created | 5th January, 2021 |
| Executed by | Ibukunoluwa Michael Adebanjo |
| Date of Execution | 5th January 2021 |
| Test Environment | Software: Unity (Game Engine)  Hardware: Eluktronics MAX-17 (CPU: Intel core i7-9750H, GPU: RTX 2060) |

**Table 5.4: Test case TC-004(Enemy AI Behavior)**

|  |  |
| --- | --- |
| Test Suit ID | R-103 |
| Test Case ID | TC-004 |
| Test case summary | Check the Save/Load game functionality |
| Related Requirement | R-XXX |
| Prerequisite | * The game folder must be present in the local storage of the   computer   * The game must be loaded on a computer with a dedicated   graphics card |
| Test Procedure | 1. Run game from Scene View 2. Play a new game by clicking the “Play new game” button 3. Progress through the game to a certain extent until an enemy   is encountered   1. Engage the enemy in combat and observe behaviour |
| Test Data | Enemy prefab |
| Expected Result | The enemy’s animations should play in their correlating scenarios without issues  The enemy, if not spotted by the player’s character, should aimlessly wander about a set of waypoints. When the character is within a close enough distance, the enemy should be alerted of the character’s presence and often roar as a warning or outright tush towards the player and attack. Upon the enemy taking a certain amount of  damage, it should die and being rendered inactive. |

|  |  |
| --- | --- |
|  | The enemy must also be able to properly navigate the environment  through the use of the baked nav mesh. |
| Actual Result | The enemy used its animations when and as it was expected.  The enemy was able to navigate the environment with ease when not in a state of alert. When the player engaged the enemy creature, it roared as a warning and pursued the player to attack it. While it took damage and eventually died, the enemy failed to do damage to the player.  The cause was the lack of foresight regarding the enemy damage stats in correlation to the player’s total health and rate of health regeneration. This was later adjusted to see if a difference would be  made in the next test case. |
| Status | Test case failed |
| Remarks | The test was a failure |
| Created by | Ibukunoluwa Michael Adebanjo |
| Date Created | 5th January, 2021 |
| Executed by | Ibukunoluwa Michael Adebanjo |
| Date of Execution | 5th January 2021 |
| Test Environment | Software: Unity (Game Engine)  Hardware: Eluktronics MAX-17 (CPU: Intel core i7-9750H, GPU: RTX 2060) |

**Table 5.5: Test case TC-005(Enemy AI Behavior)**

|  |  |
| --- | --- |
| Test Suit ID | R-103 |
| Test Case ID | TC-005 |
| Test case summary | Check the Save/Load game functionality |
| Related Requirement | R-XXX |
| Prerequisite | * The game folder must be present in the local storage of the   computer   * The game must be loaded on a computer with a dedicated   graphics card |

|  |  |
| --- | --- |
| Test Procedure | 1. Run game from Scene View 2. Play a new game by clicking the “Play new game” button 3. Progress through the game to a certain extent until an enemy   is encountered   1. Engage the enemy in combat and observe behaviour |
| Test Data | Enemy prefab |
| Expected Result | The enemy’s animations should play in their correlating scenarios without issues  The enemy, if not spotted by the player’s character, should aimlessly wander about a set of waypoints. When the character is within a close enough distance, the enemy should be alerted of the character’s presence and often roar as a warning or outright tush towards the player and attack. Upon the enemy taking a certain amount of damage, it should die and being rendered inactive.  The enemy must also be able to properly navigate the environment  through the use of the baked nav mesh. |
| Actual Result | The enemy used its animations when and as it was expected.  The enemy was able to navigate the environment with ease when not in a state of alert. When the player engaged the enemy creature, it roared as a warning and pursued the player to attack it.  The enemy is now damaging the player to a point that makes it a  threat. |
| Status | Test case succeeded |
| Remarks | The test was a success |
| Created by | Ibukunoluwa Michael Adebanjo |
| Date Created | 6th January, 2021 |
| Executed by | Ibukunoluwa Michael Adebanjo |
| Date of Execution | 6th January 2021 |
| Test Environment | Software: Unity (Game Engine)  Hardware: Eluktronics MAX-17 (OS: Windows 10 Pro 64-bit, CPU: Intel core i7-9750H, GPU: RTX 2060) |

## Appendix E – User Guide/Manual

### Player Guide:

* Play the game by navigating to the game folder and running the “.exe” file.
* You will be met with a main menu giving you the options to Play a new game, load a game save file form local storage, go to Settings or Quit the game.
* To play a new game select “Play New Game” and you will be taken to a loading screen where tips about the game will be briefly given before moving on to the game level.
* In-game, the player has the option to interact with the game environment as intended, or go to the Pause Menu by hitting the ESC button.
* In the pause menu, you can alter the game settings by clicking the “Settings” button, save the game by clicking the “Save” button, go to the main menu by clicking the “Return” button.
* In settings, you can change the game controls as well as the audio and graphics of the game.

### Controls:

* W, A, S, D to move the player Up, Left, Down and Right respectively.
* X and Y makes the player strafe left and right respectively
* Hold the Shift key to sprint
* Press Q to examine interactable objects
* Press E to add interactable objects to the inventory
* Press the TAB key to open the player inventory; items collected can be viewed, used and dropped by clicking on the item and selecting the desired action from the drop-down list that will appear once an item is selected
* Press the R key to reload a firearm
* Hold the right click mouse button to aim a firearm
* Press/hold the left click mouse button to attack with a melee weapon or shoot the firearm
* Scrolling up or down with the mouse wheel will swiftly let you switch to the last equipped item