THE SURVIVORS

(A Multiplayer Game Using Artificial Intelligence)

MAJOR PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY

COMPUTER SCIENCE AND ENGINEERING

SUBMITTED BY

AMAN CHAUHAN (1805158)

ARYAN MAHESHWARI (1805163)

GURJEET SINGH TATLA (1805174)

SUBMITTED TO

Dr. VIVEK THAPAR

Department of Computer Science and Engineering

Guru Nanak Dev Engineering College

Ludhiana, 141006

Vivek Thapar
6/6/2022

Abstract

Video games have become an integral part of our culture in a relatively short period of time. The industry is also developing into a major pillar of many modern economies, as game development programs are introduced in many developed countries. This coincides with a time when it has never been easier to release a game in the stock market. Over the past two decades, game development teams need financial support and technical support to pass rigorous testing of stadium management to be allowed access to their development computer systems. Today, anyone with a tablet and a computer, even a laptop computer, can create a game and sell it in less time with financial support. This is not to say that every game is successful: it is still important to have a good understanding of the technical aspects involved in making games and the concepts involved in designing the games that people will want to play. Sometimes the best way to develop this knowledge is to start from scratch.

The goal of our game project is to design and develop an explicit 3-D computer game using Unreal Engine and Autodesk Maya. In our project, we decided to design and develop a 3-D virtual game where the goal of the game is to find and defeat all the enemy AI bots.

The Survivors is a 3D virtual environment game in which single or multiple players will fight/survive using weapons (eg. guns) against a random number of artificially intelligent enemy bots until either all bots are eliminated or all players are eliminated/quit. The game is designed in the Windows environment and is written in C ++ and visual script language using Unreal game engine.

ACKNOWLEDGEMENT

We are highly grateful to Dr. Sehijpal Singh, Principal, Guru Nanak Dev Engineering College (GNDEC), Ludhiana, for providing this opportunity to carry out the major project work in Game Development.

The constant guidance and encouragement received from Dr. Parminder Singh, H.O.D., CSE Department, GNDEC Ludhiana has been of great help in carrying out the project work and is acknowledged with reverential thanks.

We would like to express a deep sense of gratitude and thanks profusely to Dr. Vivek Thapar, without whose wise counsel and able guidance, it would have been impossible to complete the project in this manner.

We express gratitude to other faculty members of the computer science and engineering department of GNDEC for their intellectual support throughout the course of this work. Finally, we are indebted to all whosoever have contributed to this project.

Aman Chauhan

Aryan Maheshwari

Gurjeet Singh Tatla

List Of Figures

List Of Tables

TABLE OF CONTENTS

Chapter 1. Introduction of Project

1.1 Introduction

This report describes the process involved in making a 3D Shooter Multiplayer game, The Survivors (A Multiplayer Game with Artificial Intelligence) using the Unreal Engine. This game is built specifically for PC devices.

The Survivors is a game created as a Major Project for the Computer Science and Engineering Department. It is a 3D virtual environment game in which single or multiple players will fight/survive using weapons (eg. guns) against a random number of artificially intelligent enemy bots until either all bots are eliminated or all players are eliminated/quit. Enemy bots are guided in such a manner that they can chase players and adjust its position according to the fighting pattern of actual players from which they can be able to defeat players. Enemy bots can also cooperate with each other.

The idea of game development is not new for us. We also developed a game during our Minor Project and did our 6 Months industrial training in Game development but we never developed a multiplayer game before this and developing such a game will increase our knowledge and help to gain experience in the game development field.

As everyone knows, there are several types of computer games. Shooter is one of the best-known game genres. A shooter game is a game which makes the player feel within the game world. After the success of Doom, which is accepted as the first shooter game, many companies took a crack at this type of game. Nowadays, the best selling game is a kind of shooter.

1.2 Project Category

This major project comes under the category of game development. In game development, various types of tools and methods are used. Developing a game is not just doing programming

but also needs to work in virtual environments, which is the foremost step of game development. 3d and 2d art is further categorized into various types,which have their own elements.

1.3 Objectives

The major objectives of this game project are:

- To design and implement a 3-dimensional game written in C++ using a game development studio.
- To facilitate multiple players in a single session/room
- To implement enemy bots with the help of Artificial Intelligence.

In addition to these, the following objectives are also completed:

- To implement attacks between players and enemy bots using weapons.
- To implement enemy bots that chase and attack players until death.
- To implement multiple enemy bots that use AI to raise difficulty.
- To design a futuristic and thrilling game map.
- To design a user-friendly user interface.
- To add sound effects to players, enemies and weapons.
- To add background music to make the game more enjoyable.

1.4 Problem Formulation

In this context the main questions posed here are:

How can a multiplayer game be implemented without the use of an expensive game server?
and
How is artificial intelligence used to enrich the gaming experience?

The core focus of this project was to develop a game that is universally accepted and hence

improves our knowledge in game development, and in conjunction, software development.

Multiplayer games such as Call of Duty, PUBG, the World of Warcraft and Counter-Strike have attracted millions of players on a global scope and have achieved huge commercial success. Such games employ high quality graphics, rapid response servers and, to some extent, Artificial Intelligence, all fields that interest Computer Science engineers.

Our focus was to build a game that implemented a simple functioning multiplayer without having to purchase and host a game server. In addition, we wanted to implement artificial intelligence in a way different from the norm in order to learn and increase our own knowledge.

1.5 Existing System

Shooter video games or shooters are a subgenre of action video games where the focus is almost entirely on the defeat of the character's enemies using the weapons given to the player. Usually these weapons are firearms or some other long-range weapons, and can be used in combination with other tools such as grenades for indirect offense, armor for additional defense, or accessories such as telescopic sights to modify the behavior of the weapons.

Shooter games test the player's spatial awareness, reflexes, and speed in both isolated single player or networked multiplayer environments.

Third-person shooters are characterized by a third-person camera view that fully displays the player character in his/her surroundings. Notable examples of the genre include the Tomb Raider series, several entries in the Resident Evil and Metal Gear Solid franchises, Star Wars: Battlefront and Gears of War. Third person shooter mechanics are often incorporated into open-world adventure games.

There are many third person shooter video games such as Call of Duty, PUBG, the World of Warcraft and Counter-Strike which have achieved huge commercial success using their own unique systems, graphics and, as required, Artificial Intelligence logic.

1.6 Proposed System

The Survivors is a 3D shooter video game in which single or multiple players fight/survive using weapons (primarily guns) against a random number of artificially intelligent enemy bots in a virtual environment map until either all bots are eliminated or all players are eliminated. It will incorporate a crosshair icon for aiming the shots to compensate for the difficulty of aiming from a third-person camera.

The game shows the player from a "behind the back" perspective. The third-person perspective allows the game designer to design more personalized player characters and directs the player's attention like watching a film.

The Survivors allow players to see the area surrounding the player character clearly. This viewpoint facilitates more interaction between the character and their surrounding environment, such as maneuvering tight quarters. The third-person perspective is better for interacting with objects in the game world, such as jumping on platforms or engaging in close combat.
Third-person perspective allows for the design of a large and spacious environment with numerous possibilities..

Chapter 2. Requirement Analysis and System Specification

2.1 Feasibility Study

This study describes all the possibilities that come as questions to both the developers and other users during the development of software.

2.1.1 Game Concept

A 3D virtual environment game map in which single or multiple players will fight/survive using weapons (eg. guns) against a random number of artificially intelligent enemy bots until either all bots are eliminated or all players are eliminated/quit.

2.1.2 Financial Feasibility

The Survivors is a multi player game that does not work using a single dedicated game server, so there is no hosting cost, and if it is deployed to a game server, it will only consume internet data to send data back and forth. For the game players, it is completely a free to play game, the only potential reason that it may consume extra internet data is the ads that this game may have in the near future which is considered to be relatively low. Additionally if we add some premium weapons or player skins in future the player may need to pay to acquire them which is a choice for the players and is not a must. The points mentioned above indicate that the project is financially feasible.

2.1.3 Technical Analysis

- Designing the 3D virtual environment map, character models and weapons
- Enabling player movement and animation

- Enabling weapons to fire
- Implementing health and death (health=0) mechanics
- Enabling enemies through Artificial Intelligence
- Implementing multiple players in a single lobby.
- Implement end of game conditional (win/loss)

2.1.4 Market analysis

- Genre: Shooter
- Platform: PC
- Age Range: 8+

For this project we need hardware configuration like:

- Processor - intel i5 or greater (or similar to this)
- Ram - 8gb or greater
- Graphic Card - Nvidia GTX with Vram 4gb or greater (or similar)
- Storage - 1TB or greater

Software requirements includes:

- Epic Game Launcher
- Unreal Engine 4 or greater
- Autodesk Maya
- Blender
- Photoshop

2.1.5 Resource Cost

- Unreal Game Development Engine: Open Source
- Autodesk Maya: Student License
- Blender: Open Source
- Photoshop: Student License

2.1.6 Legal Feasibility

The game assets that were used to make this game are either completely free or built by us. And as mentioned in the previous sections this game uses freely available software and tools which are intended for the use of game developers and/or students. There cannot be any conflict regarding plagiarism of any other game, because we have already followed the rules of the licenses to make this game

2.2 Software Requirement Specifications

This section covers the requirements specification of our game “The Survivors”. It includes the specification of this documentation with general description, specific requirements, and analysis of models. It also includes changes management of this requirement specification in case of any change.

2.2.1 Introduction

This section specifies the purpose of the document, document convention and document scope.

2.2.1.1 Purpose of SRS

This Software Requirements Specification (SRS) part is intended to give a complete overview of the game “The Survivors” including the action flow, initial user interface and story therein. The SRS document details all features upon which we have currently decided with reference to the manner and importance of their implementation.

2.2.1.2 Document Conventions

This document will use the pronoun “we” to refer to the development team. As the development team is also responsible for the SRS document, no ambiguity arises from its usage. There is a clear distinction, however, between the use of the words “player/gamer” and “character.” The “player” is the human being interacting with the game in the real world, while the “character” is the in-game player avatar being manipulated.

2.2.1.3 Intended Audience and Reading Suggestions

The SRS document gives the project team a way to ensure the game’s adherence to the original vision. Although the document may be read from front to back for a complete understanding of the project, it was written in sections and hence can be read as such. For an overview of the document and the project itself, see Overall Description. For a detailed description of the game-play elements and their interaction with the character, read System Features. Readers interested in the game-play interface and navigation between different front-end menus should go through External Interface Requirements. Technical standards to which the team will hold the project are laid out in Other Nonfunctional Requirements.

2.2.1.4 Scope

This Software Requirements Specification (SRS) describes the functional and nonfunctional requirements for the project. The game "The Survivors" was conceived by the 3 team members of our group as having an anticipated development cycle greater than the length of the semester. The team wishes to carry on the project until its completion. The game will continue to grow until we feel it is satisfactory for open-source distribution.

The scope of the project is to develop a 3-dimensional video game. The system shall use the Unreal Engine to handle most of the game development process. The project will be based on creating a shooter game with the goal in mind of being fun and gaining experience in game development.

2.2.2 General Description

This section includes the perspective of our product and the system environment it requires. It specifies the QFD (Quality Function Deployment) of our game and also the User Story of it.

2.2.2.1 Product and Business Perspective of the Game

Software product development is a paradigm shift from routine application maintenance and support in the software industry. Development of a game/software product from scratch is a significant challenge for any organization. It requires considerable investments in terms of effort and cost and also confirms client involvement, knowledge about the client market (for example, Google play). This report product perspective describes the overall description.

2.2.2.2 System Environment

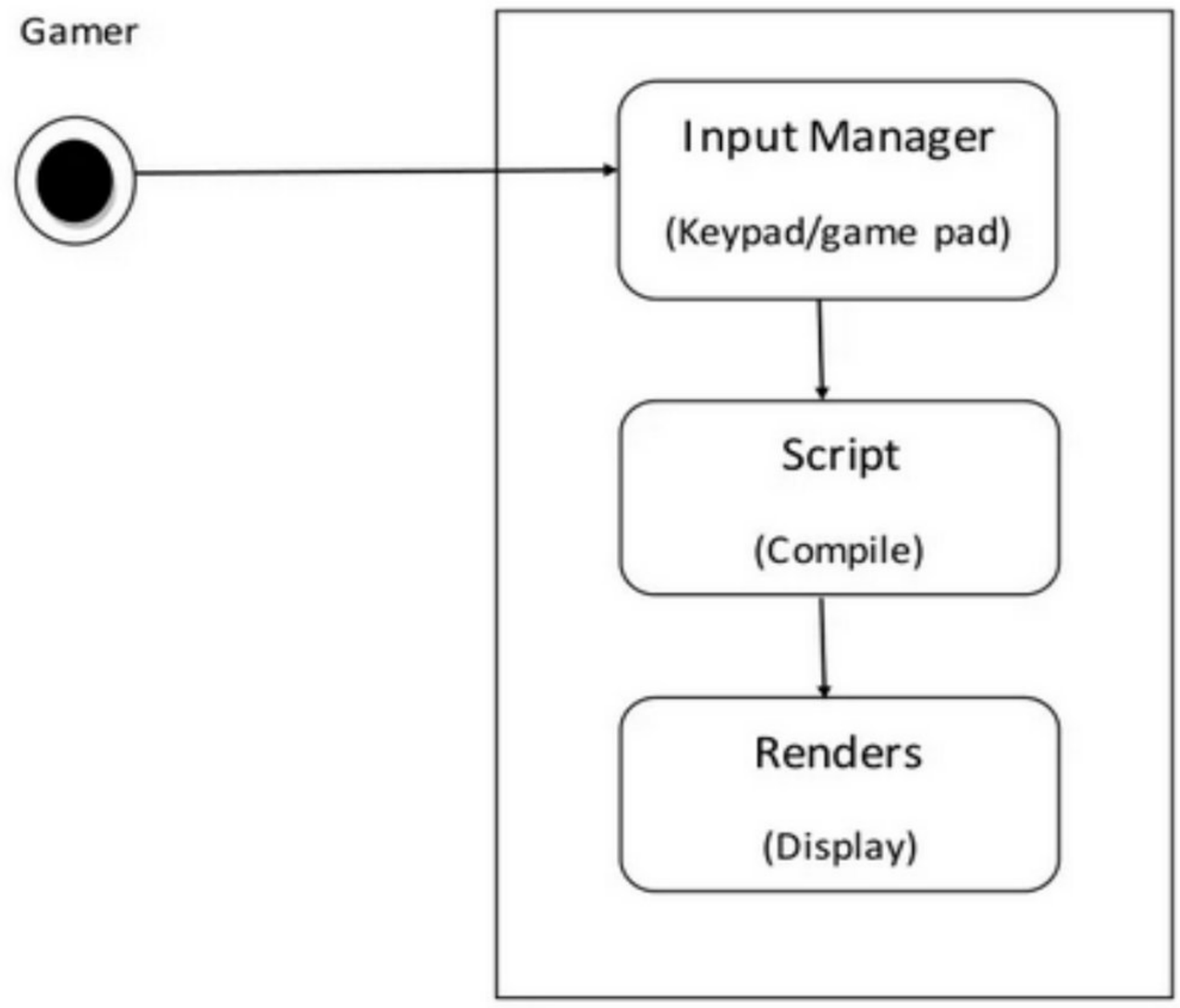

Fig 2.1: System User Interaction

Gamers can interact with the system by giving input to the game. System gives those inputs to the script, and if any change occurs (for example, if any value is changed), it modifies the render to display the changes (for example, a character can change its place).

2.2.2.3 Quality Function Deployment

Quality Function Deployment is a technique that translates the needs of the customer into technical requirements for software/game. It concentrates on maximizing customer satisfaction from the Software engineering process. With respect to our project the following requirements are identified by a QFD.

Normal Requirements

Normal requirements consist of objectives and goals stated by clients. They are as follows:

1. User friendly and efficient system.
2. Minimum maintenance cost.
3. Availability of expected requirements within the PC configuration.
4. Easy to operate.

Expected Requirements

These requirements are implicit to the system. Their absence will be a cause for dissatisfaction. They are as follows:

1. Develop the system with limited cost.
2. Maximum graphics quality.

Exciting requirements

These requirements are for features that go beyond the customer's expectations and prove to be very satisfying if present. They are as follows:

1. Maximum high regulation with minimum hardware.
2. Easy to understand.
3. Allow for character configurations.

2.2.3 Specific Requirements

This section covers the project requirements.

2.2.3.1 Functional Requirements

1. The player character(s) must move around the virtual environment as per user input.
2. The player character(s) must aim and shoot as per user input.
3. The player character(s) must lose health when receiving enemy bullet(s).
4. The player character(s) must die when health reaches zero.
5. The enemy characters must chase player character(s).
6. The enemy characters must aim and shoot at player character(s).
7. The enemy characters must die when their health reaches zero.

2.2.3.2 Performance Requirements

1. The system must be highly responsive regardless of player input.
2. The system must be reliable, trustworthy and consistently good in performance.
3. The system must be available 24/7
4. The system must be available to be used any number of times by the user(s).
5. The system must be capable of producing desired results or even better results.

2.2.3.3 Security Requirements

1. The system must not be susceptible to attacks by modders or hackers.
2. The system must not provide any back room entry into a user's device to potential thieves.

2.2.3.4 Look and Feel Requirements

1. The system shall be attractive to the target audience.

2. While playing the game the player will feel a realistic view.
3. The gameplay shall move smoothly through the various actions.
4. Multiplayer shall work without any disruption on any player screen.
5. There shall be no button press delay for.
6. The user shall be able to enjoy the hard hitting background music included in the game.

2.2.3.5 Hardware Interfaces

"The Survivors" is a PC gaming application designed specifically for the Windows platform and is functional on both Windows and MacOS. Gaming application data is stored locally on the game engine elements. "The Survivors" has been developed for Windows 10 and all subsequent releases. In the future we will release it for other platforms as well.

2.2.3.6 Software Interface

"The Survivors" has been developed using a series of game development tools. Working tools are as follows:

1. Unreal Engine
2. Autodesk Maya
3. Adobe Photoshop
4. Epic Game Launcher

2.2.3.7 User Characteristics for the System

There is only one user at a time in a software on a single device and the user interacts with the game (system) in different ways. So, Gamer is the only one who communicates with the system through playing the game. And this gamer can be any person.

2.2.4 Analysis Model of Project

This section describes the Software Requirements Specification of our project by analyzing the proper models of requirement engineering.

2.2.4.1 Scenario Based Model

This Model depicts how the user interacts with the system and the specific sequence of activities that occur as the software is used.

Use Case Scenario

The following table summarizes the use cases of the system. We have created the use cases based on the user's view of the game.

Table 2.1 : Use Cases Table

Level 0	**Level 1**	**Level 2**
“The Survivors” Game	Play	Create Server
		Join Server
		Replay
		Exit Game
	Quit	

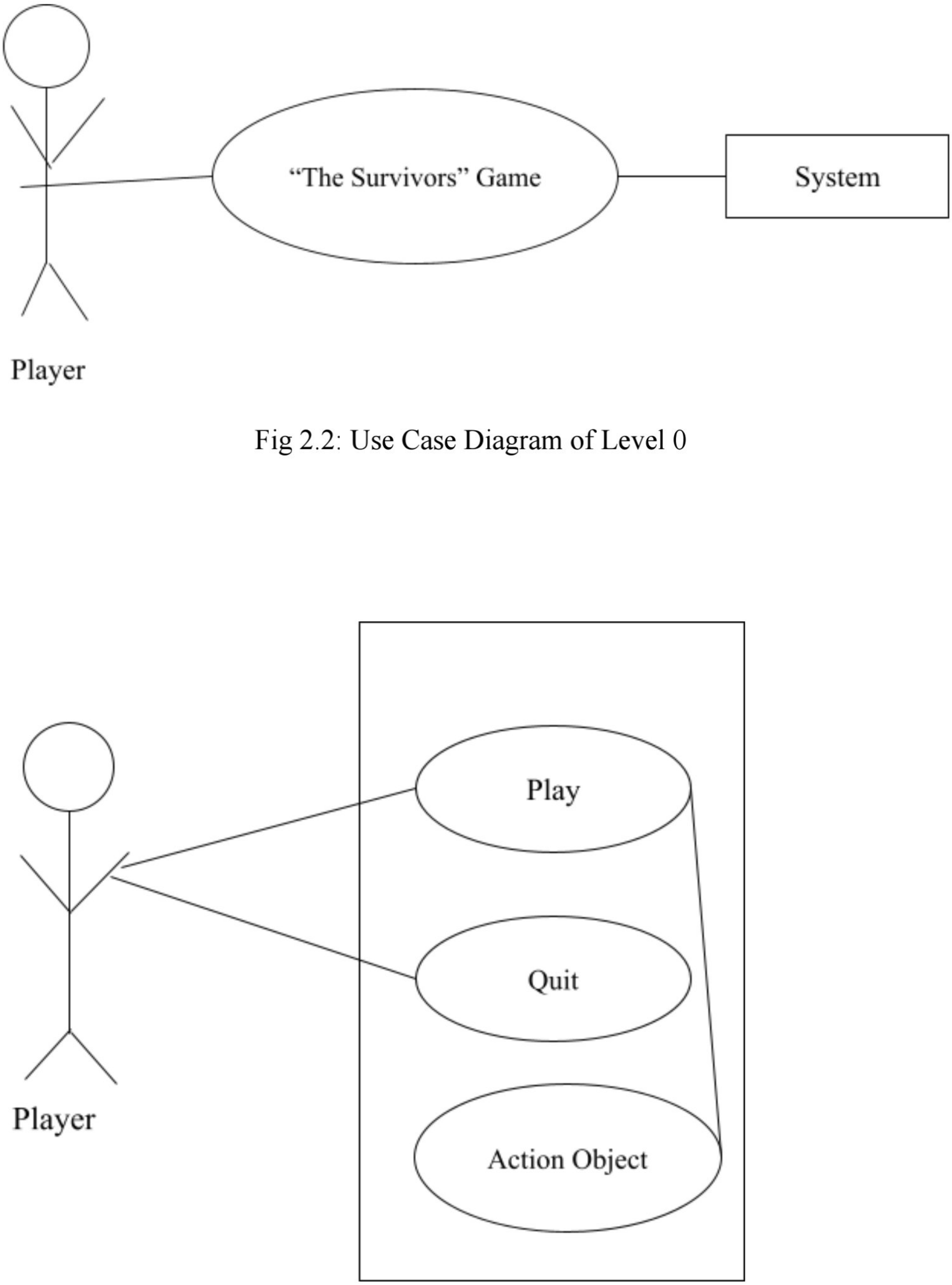

Fig 2.2: Use Case Diagram of Level 0

Fig 2.3: Use Case Diagram of Level 1

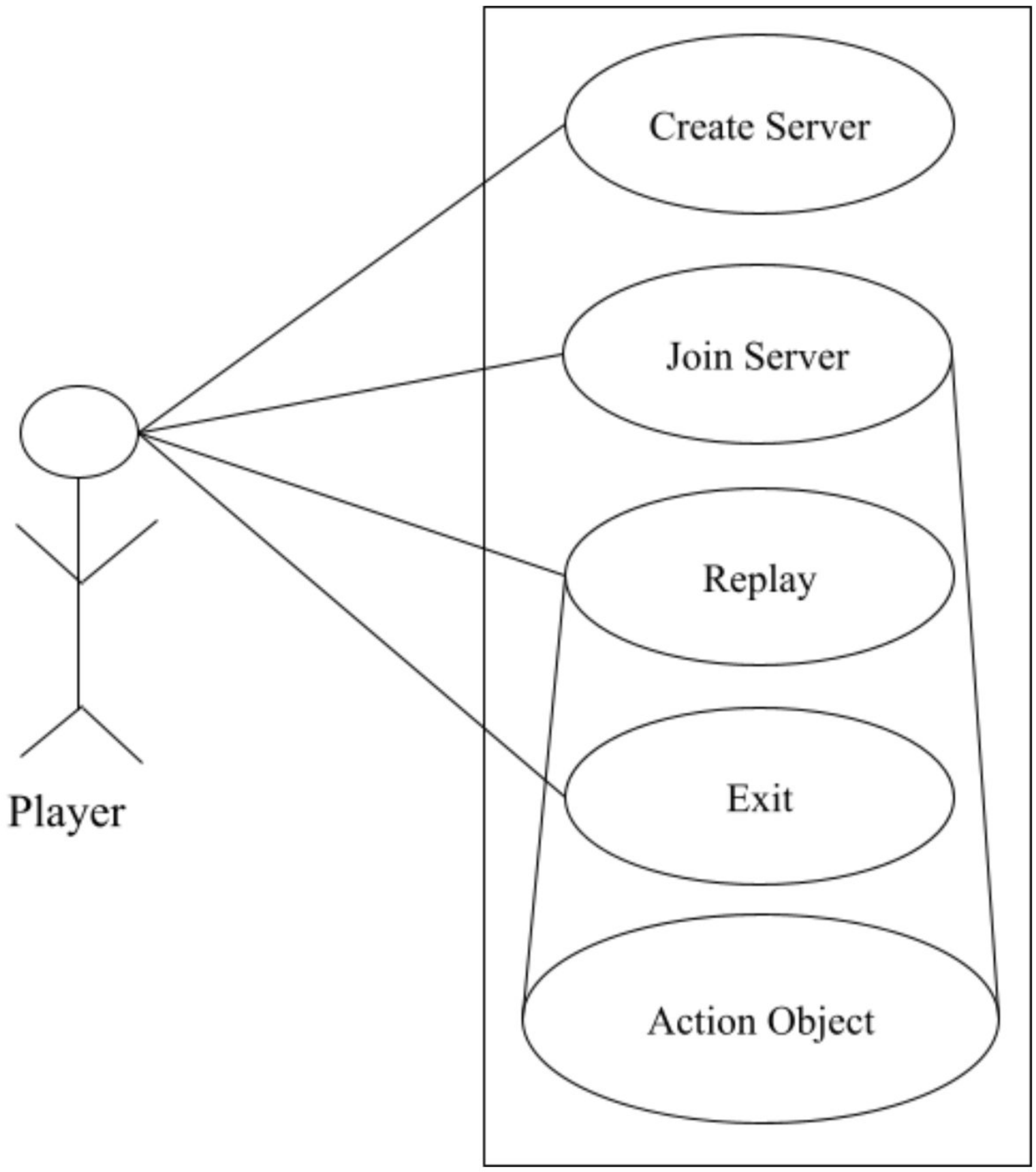

Fig 2.4: Use Case Diagram of Level 2

This Diagram of Level 2 leads us to the "Play" module of the use cases. These use case descriptions are given here –

1. Use case: Create Server

Primary Actors: Any one playing the game

Goal in context: To start a new game session

Precondition:

1. System supports the game configuration
2. The file has been triggered to run game and the game screen has appeared
3. Local network is running

Triggers: The player needs to start a new game session

Scenario:

1. Go to the main menu of the game
2. Click "Create Server" button
3. New game session is created on the local network.
4. New game is started on a device connected to the local network.

Exception: Game crashed

Priority: Essential, must be implemented

When Available: First increment

2. Use case: Join Server

Primary Actors: Any one playing the game

Goal in context: To join a game already running on specified device IP

Precondition:

1. A game session was created beforehand
2. Local network can support load

Triggers: Need to join existing game session

Scenario:

1. Go to the main menu of game
2. Enter device IP for existing game session
3. Click the "Join Server" button
3. Game is loaded in existing session

Exception:

1. Local network load exceeded
2. Game crashed
3. No existing session on provided device IP

Priority: Essential, must be implemented for multiplayer

When Available: First increment

3. Use case: Replay Game

Primary Actors: Any one playing the game

Goal in context: To reload the game session

Precondition:

1. Game session was created from this device initially
2. Local network is running

Triggers: Need to reload game session to replay game

Scenario:

1. Reach win/loss screen
2. Click the "Replay" button

Exception:

1. Local Network not running
2. Game crashed

Priority: Essential, must be implemented

When Available: Second increment

4. Use case: Exit Game

Primary Actors: Any one playing the game

Goal in context: To exit from the game

Precondition: The game is currently playing

Triggers: Player needs to exit from the game

Scenario:

1. Go to the main menu of the game
2. Press the "Exit Game" button
3. Game is exited

Priority: Essential, must be implemented

When Available: First increment

There is another module for “Quit” in the Level 1 of the Use Case Diagram. The Use Case for it is:

Use case: Quit

Primary Actors: Any one playing the game

Goal in context: To Exit from the Game Process

Precondition: Player has entered in the game process

Triggers: Player needs to exit from the game

Scenario:

1. Go to the main menu
2. Click the "Exit Game" button
3. Game is exited

Exception: Something went wrong. Cannot exit now.

Priority: Essential, must be implemented

When Available: First increment

2.2.4.2 Data Model

If software requirements include the need to create, extend or interface with a database or if complex data structures must be constructed and manipulated, the software team may choose to create a data model as part of overall requirements modeling. Although our game has many data objects, it does not have any data storage. All the objects and their related data are handled by the game engine. So the developers need not think about data storage. For this reason, a data model cannot be constructed for this project.

2.2.5 Requirement Change Management of our System

The developers intend to release a complete and fully functional game that follows the guidelines mentioned in the SRS document. However, since the product will be released for Windows platforms, updates will likely be required. These updates would consist of any bug fixes that are necessary, compatibility patches for all of the current devices that support the Windows platform, and expansions of the content.

2.2.5.1 Bugs and Glitches

The players should contact the developers to present any bugs or glitches they have detected and if they have any beliefs that the game is not functioning properly. General concerns or comments would also need to be submitted here.

2.2.5.2 Patches

As the Windows system is updated and new devices are released, the game would also need to be updated. Developers would constantly be making changes in order to keep up with any compatibility issues that may arise. These changes and any others that may be fixing bugs or glitches would be released through these patches.

2.3 Expected Hurdles

1. There are a lot of operating systems, all of them having their own set of rules and SDK.
2. We adopted some knowledge by following video tutorials, text tutorials, internet and learning materials given by the tools themselves. It is a matter of time, patience and hard work.
3. The system may need periodic maintenance activities.
4. Creating a 3D model is very difficult because you need to work with each and every point of the model.
5. The system may end up incredibly complex and hence hard to fix in case of errors.
6. It is very sensible work and it demands much time because the game engines try to imitate the real world in the game environment.

7. The Unreal game engine demands vast knowledge about its properties, sections and subsections. After all the thing is that a game project is not a project of 6 or 8 months for three people.

Chapter 3 Game Design

3.1 Design Approach (Function oriented or Object oriented)

We are going to use an object-oriented approach.

We have to think about the Classes that we want to build, with the associated variables and functions that will make sense for the development.

The theme throughout the game project is the word mechanic to relate to the actions which are taking place in games from the internal operations of animation and programming to the interactions between the environment and the player. However, in game studies, the game mechanic is used to refer to developed relationships which facilitate and define the game's challenges between game and players. They are complicated systems that contain a series of rational motivations, actions, goals, and feedback of the players.

Comprehending a game mechanic is useful for making actions and other elements that may be implemented with those actions open up a plethora of infinite ideas for games by combining actions, rules and goals.

3.2 Detail Design

Project Description

This game design document describes the details for "The Survivors", a 3D Third-person shooter video game, built specifically for PC devices.

"The Survivors" is a 3D virtual environment game in which single or multiple players will fight/survive using weapons (eg. guns) against a random number of artificially intelligent enemy bots until either all bots are eliminated or all players are eliminated/quit. Enemy bots are guided

in such a manner that they can tackle players and will learn the fighting pattern of actual players from which they can be able to defeat players. Enemy bots can also communicate with each other.

Third-person perspective allows for the design of a large and spacious environment with numerous possibilities. It allows the game designer to design more personalized player characters and directs the player's attention like watching a film.

“The Survivors” will take place in a 3D virtual environment that will appear futuristic and provide thrill to the players. The player characters and enemy bots will all be humans or robots protected by futuristic armor. The weapons used by the player characters and the enemy bots will also be futuristic and will fire laser beams instead of bullets to add to the game’s aesthetic.

3.2.1 Characters

This is one of the interesting sections that was achieved with enthusiasm, because it contains lots of colorful figures. Here we will discuss the characters: the main character (played by users) and the enemies.

The main character model uses primarily grey color body armor which includes a helmet, breastplate, vambrace, rear brace, cuisse and greave. The grey color signifies a broken-down, weaker and less effective version of the enemy bots’ armor.

The enemy character model consists of the same body armor in pure white color to signify their superior defense and “no longer human” nature.

3.2.2 Story

Humans have built an artificial intelligence system named A-A-G after their creators for the sake of a world which is dying due to human actions over the past millennia. This system was built with the capabilities to understand human nature and behave as humans in order to co-exist with them and become the next step in their evolution.

With the passage of time, the system became able and outperformed regular humans in all respects. Thinking itself superior and hence worthy to rule over the world, it turned into evil and made the humans it was supposed to help into its slaves. A-A-G created an army of humans and brainwashed them, killing those who are unhealthy or old.

But there were a few humans who were mentally strong enough and underestimated by A-A-G, they were able to remember their past. With their memories recovered, they have started to struggle against the A-A-G's army to get their world back.

One of the escapees named Aspark is a software scientist who leads the humans since he is aware of the weakness of the A-A-G system. He plans the first attack on the city tower controlled by A-A-G. It is a smaller tower where a very few soldiers are active but a lot of necessary material like guns and armor are available.

This group of escapees name themselves THE SURVIVORS, and embark on a journey of survival to eliminate A-A-G and take back their world.

3.2.3 Gameplay

Single or multiple players will enter a virtual 3D environment equipped with weapons (primarily guns). The objective for victory is to eliminate all the artificially intelligent enemy bots to survive and take over the location where the enemy bots operate.

If the player(s) die or if the player(s) do not defeat all enemy bots in time, the game will be lost.

The players can move freely in a large virtual 3D environment. Players have to save their health and defeat the enemy bots in order to win the game.

The enemy bots will utilize AI to accurately attack the player(s), work together to eliminate the players, and even hide behind various map elements in order to protect themselves from player attacks.

Multiplayer will be achieved through the use of temporary game servers hosted on a local network connection. A player has to initially create a game server, and then multiple players can join that server using the initial player's IP address.

The User Interface is simple and user-friendly. The initial game menu provides the option to create or join a game session. When a player or multiple players enter a game session, they are immediately required to hide, battle and survive the onslaught of enemy bots

3.2.4 Key-features

- "The Survivors" is a shooter video game.
- It is third-person, hence the player character is visible on-screen.
- Multiple players can join a single game session in order to play together.
- The player can move freely within a large virtual 3D environment.
- Enemy bots will hunt down and attack players.
- Through the use of AI, enemy bots will attack players faster, cooperate with each other and hide to protect themselves from players
- The players will attack enemy bots using weapons.
- The gameplay consists primarily of roaming the 3D game map to find and defeat enemy bots.
- Fighting in The Survivors is real-time intensive and requires quick physical response from the players.

3.2.5 Genre

Shooter video games or shooters are a subgenre of action video games where the focus is almost entirely on the defeat of the character's enemies using the weapons given to the player. Shooter games test the player's spatial awareness, reflexes, and speed in both isolated single player or networked multiplayer environments.

Third-person shooters are characterized by a third-person camera view that fully displays the player character in his/her surroundings. Notable examples of the genre include the Tomb Raider series, several entries in the Resident Evil and Metal Gear Solid franchises, Star Wars: Battlefront and Gears of War. Third person shooter mechanics are often incorporated into open-world adventure games.

3.2.6 Gameplay

- The large virtual 3D environment allows free-roam for the player.
- Enemy bots will hunt down and attack players using weapons.
- Conversely, players can also attack enemy bots using weapons.
- In multiplayer mode, players can cooperate and strategize to defeat enemy bots faster.

3.2.6.1 Goals

Overall: To achieve victory over the opposing organization.

In-game (short term): Defeat enemy bots in game map

3.2.6.2 Rules

- Only weapons can be used for attack.
- It is not possible to fire through walls.
- A killed player or enemy bot cannot be revived.
- The game will not end until either all players or all enemy bots are defeated.
- To win the game, the players must eliminate all the bots and capture the location in time.
- All the elements in the 3d virtual game map react naturally to physics such as gravity and collision.

3.2.6.3 Losing

The losing condition is simple: if the player character dies, the player loses.

When a player loses, the character model disappears after a while.

3.2.6.4 Mechanics

- Core Mechanic: Shoot down a random number of highly skilled AI bots.
- Secondary Mechanic: Explore the central station.
- Narrative: Survive and capture the central station.

"The Survivors" also employs the following general gameplay mechanics.

- 3D virtual game environment to allow strategy opportunities.
- The players can attack bots using weapons in the game.
- Camera movement is achieved through the mouse.
- Players move forward, backward, left and right by pressing the respective arrow keys.
- By pressing the left click button of the mouse players can fire the weapon.

- Previously locked paths may open up if certain rooms/parts of the map are cleared of enemy bots.

3.2.6.5 User Skills

To play and clear this game, users require the following skill:

- Fast reflexes
- Ability to strategize
- Good mouse control
- Memory

3.2.7 Technical Description

Initially, the game will be a PC standalone version.

Follow up with Mac support.

Could add console support (through e-stores) or Mobile Cross-platform:

- iOS
- Android
- Windows Phone

in future.

Consider upgrading the game engine to Unreal Engine 5.

3.3 System Design

3.3.1 LEVEL 0 DFD

This DFD explains the work of the User . Firstly click on the game icon to enter the game.

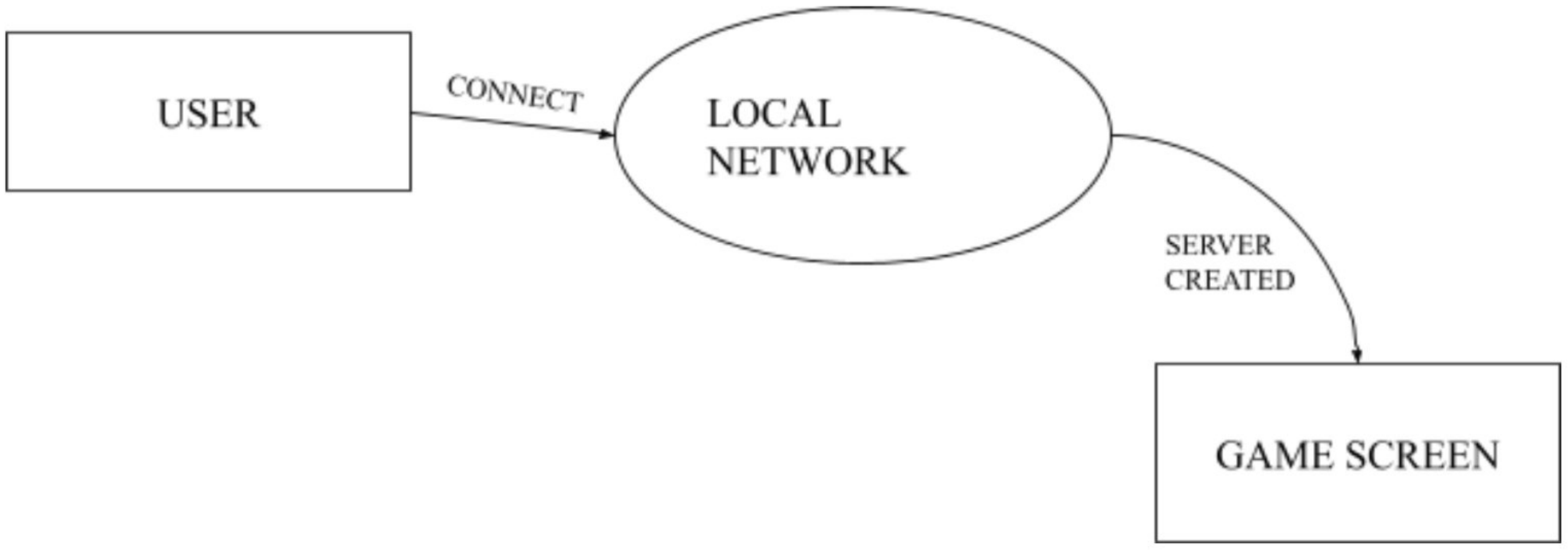

Fig 3.1: DFD Level 0

3.3.2 LEVEL 1 DFD

This DFD shows the working of the user . There are two options to play the game. There is Create a Server or Join a Server. The corresponding game screen will appear.

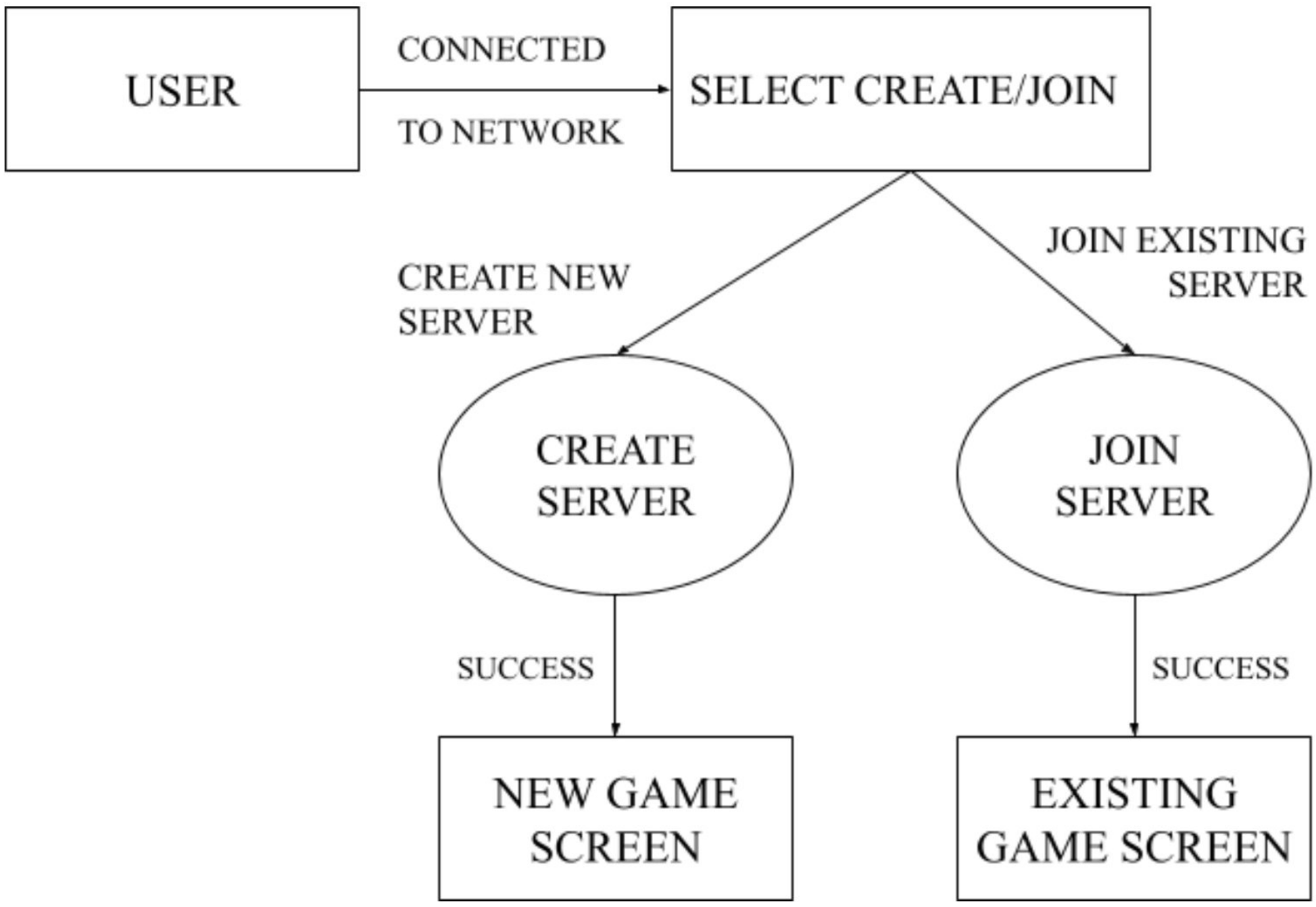

Fig 3.2: DFD Level 1

3.3.3 LEVEL 2 DFD

This DFD shows the working of the user . When the user is playing the game, if the player dies then one instance occurs :-

REPLAY

Then further activities will proceed according to game rules.

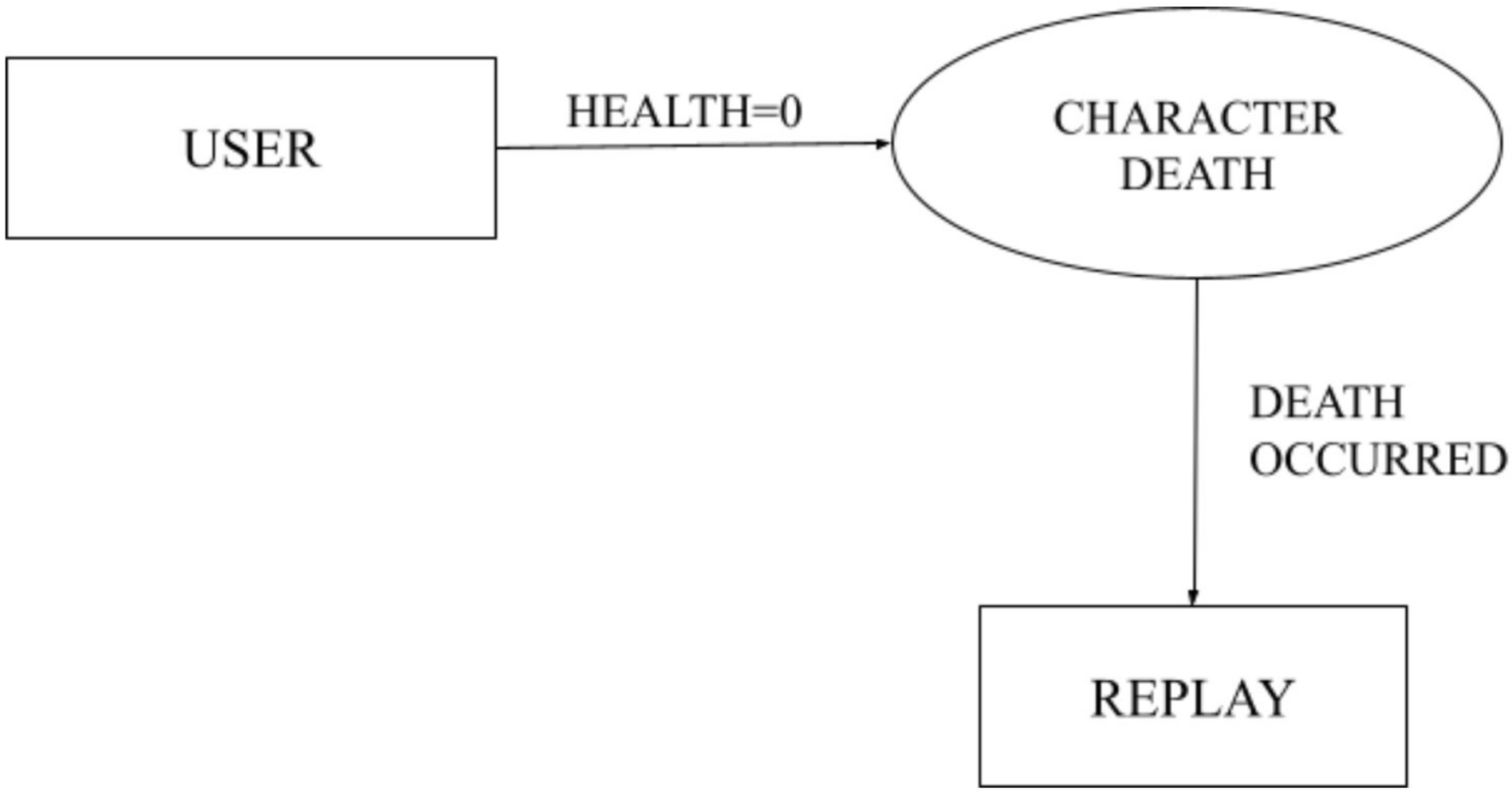

Fig 3.3: DFD Level 2

3.3.4 LEVEL 3 DFD

This DFD shows the working of the user .

When the user is playing the game, if the player dies, then one instance occurs :-

REPLAY

Player has to wait for 5 seconds then the game restarts.

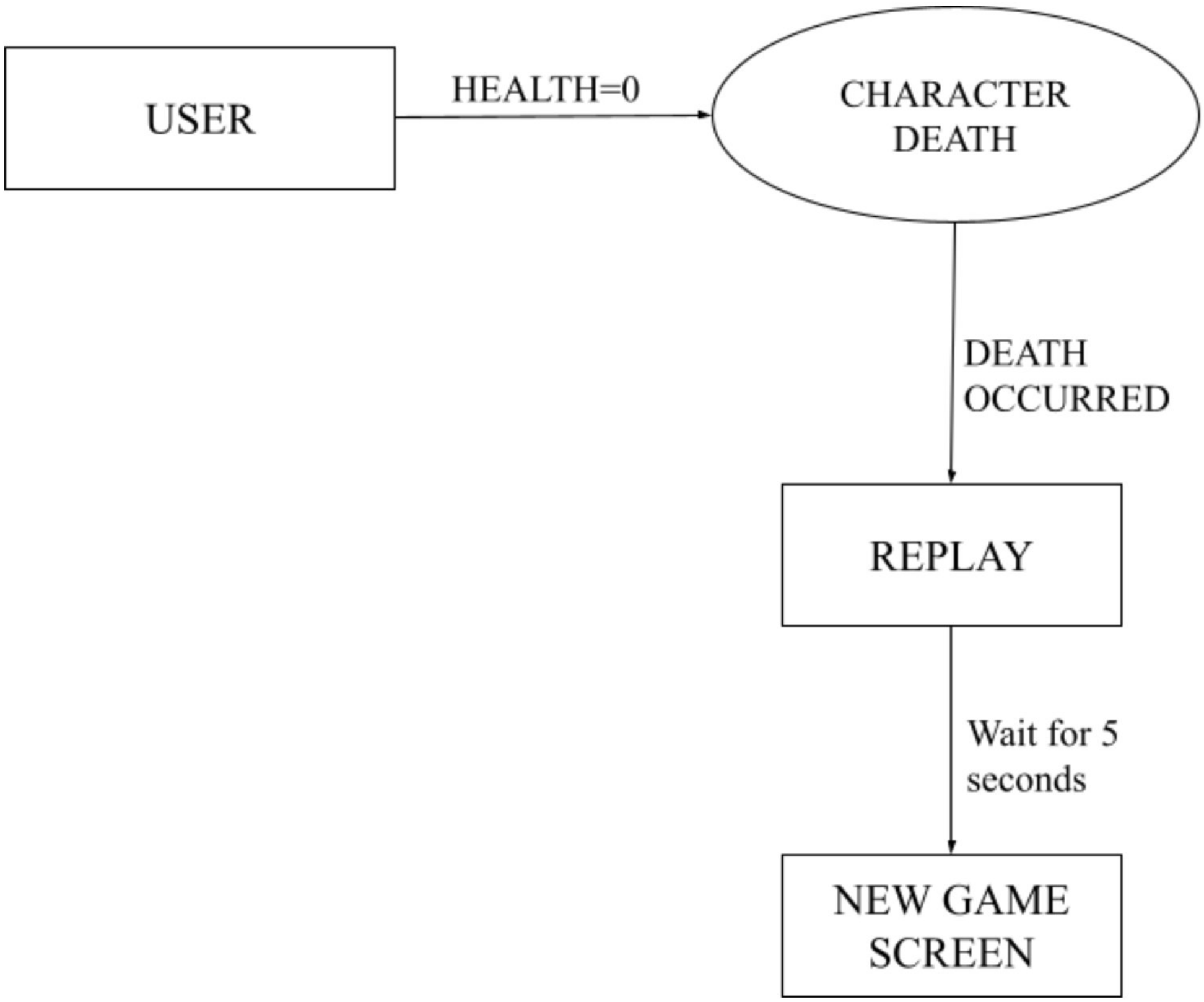

Fig 3.4: DFD Level 3

3.4 User Interface Design

3.4.1 Main Menu

In the Main Menu, the player has the option to create or join a game.

Fig 3.5: Main Menu Screen

3.4.2 Death/Loss Screen

The death/loss screen appears when player(s) have been defeated

Fig 3.6: Death/Loss Screen

3.4.3 Win Screen

The win screen appears when all enemy bots have been defeated.

Fig 3.7: Win Screen

3.5 Methodology

The steps followed to build "The Survivors" are as follows:

- Designing the 3D virtual environment map, character models and weapons
- Enabling player movement and animation
- Enabling weapons to fire
- Implementing health and death (health=0) mechanics

- Enabling enemies through Artificial Intelligence
- Implementing multiple players in a single lobby.
- Implement end of game conditional (win/loss)

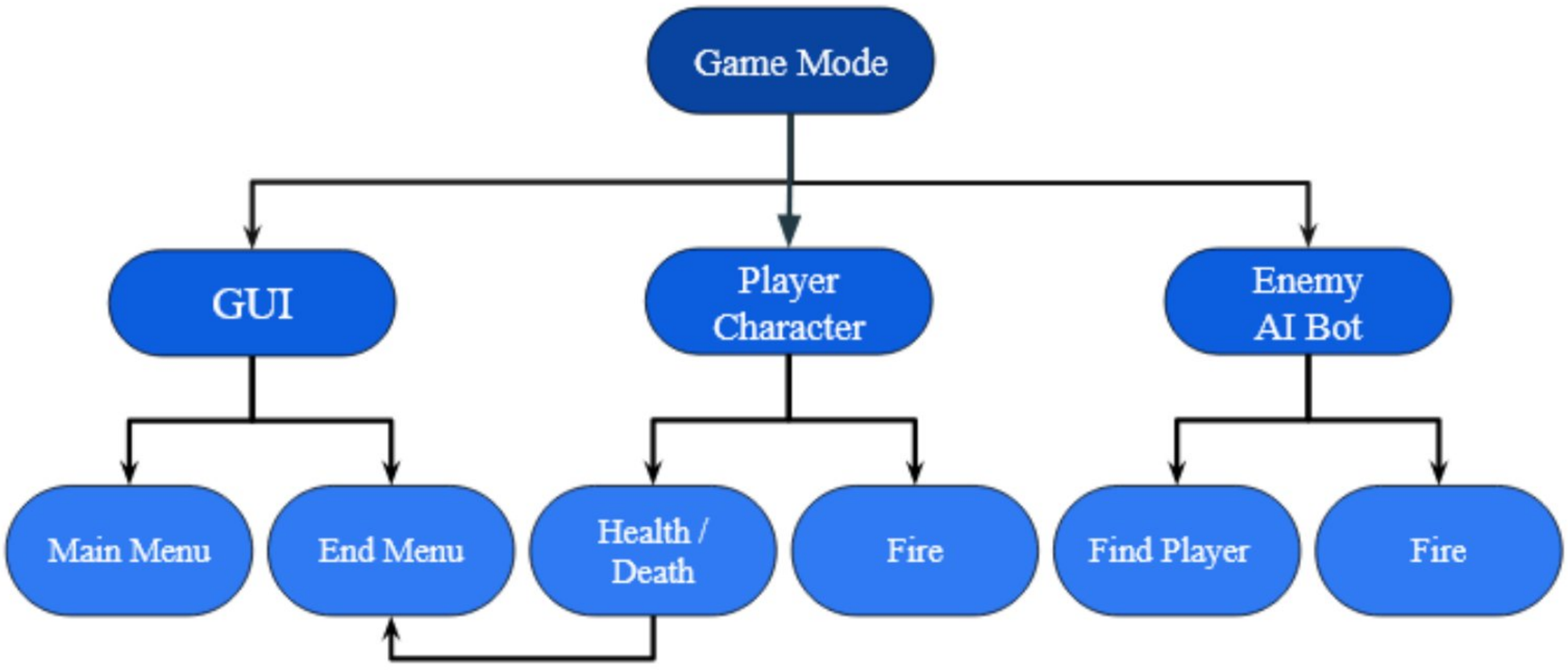

Fig 3.8: Methodology

Chapter 4. Implementation, Testing and Maintenance

4.1 Introduction to Languages

4.1.1 C++

C ++ is a general-purpose programming language created by Danish computer scientist Bjarne Stroustrup as an extension of the C programming language, or "C with Classes". The language has expanded significantly over time, and modern C++ now has object-oriented, generic, and functional features in addition to facilities for low-level memory manipulation. It is almost always implemented as a compiled language, as a cpp file extension.

C++ was designed with an orientation toward systems programming and embedded, resource-constrained software and large systems, with performance, efficiency, and flexibility of use as its design highlights. C++ has also been found useful in many other contexts, including desktop applications, video games, servers , and performance-critical applications.

Object-based programming language that provides a clear structure to programs and allows code to be reused, which reduces development costs. C ++ is portable and can be used to improve custom applications across multiple platforms.

To use C ++, you need two things:

- Code editor, such as VS Code, for coding C ++
- A compiler, like GCC, that will translate C ++ code into computer-readable language

4.1.2 Visual Scripting Language (Blueprint Programming Language)

The Blueprint Visual Scripting system in Unreal Engine is a complete gameplay scripting program based on the concept of using a location-based visual interface to create game play features within Unreal Editor. Like most common writing languages, it is used to describe classes. As you use UE4, you will often find that items described using Blueprint are colloquially called "Blueprints." This program is extremely flexible and powerful as it gives designers the ability to use almost the entire range of concepts and tools commonly available only to program planners. In addition, the Blueprint tag located in the use of Unreal Engine's C ++ allows programmers to create basic systems that can be extended by designers. Game editing and everything that UnrealScript used in the past can now be encrypted using C ++. At the same time, while Blueprints can be understood as unrealScript substitutes, they serve many of the same purposes as those of UnrealScript, such as:

- Expanding classes
- Maintenance and repair of fixed structures
- Manage the topic (e.g. sections) to replace classes

4.1.3 Unreal Engine 4

Unreal is an engine which is used to develop games. The latest version of this engine is Unreal Engine 5. The unreal engine is used to develop classic graphical games and much more. In unreal two types of languages are used which are C++ and visual scripting (blueprint). Famous games such as Pubg and Call of Duty developed in the Unreal engine. Graphical design and environment of games are developed in other software which further import in unreal. Speciality of unreal engine is that it supports high graphics games.

Main Editor View : The Main Editor Window indicates all tools and function windows. The first step of a designer is to know this interface and what every button means, because almost all operations are done in the window except programming edits. Every sub-window should be known well.

Scene View

The Scene Window mainly stores model assets of the game project. There are various kinds of 3D models, such as players, enemies, NPC, items, sky, and terrain. All of the models can be displayed in the scene window. This window is the most important part of the unreal game engine.

The scene window shown in following figure, is where the users builds the visual scene of the project, where users can plan and execute their ideas

Game Window

The Viewports are your window into the worlds you create in Unreal. They can be navigated just as you would in a game, or can be used in a more schematic design sense as you would for an architectural Blueprint. The Unreal Editor viewports contain a variety of tools and visualizers to help you see exactly the data you need.

Hierarchy View

The Hierarchy View is used to store the specific objects in the Game Scene, such as camera, 2D texture, texture 3D, light, box, spheres, cubes, models, plan, terrain.

The Hierarchy view indicates the game objects in the currently activated scene. Game objects that are dynamically created and removed from the activated scene will emerge here when they are created and activated in the scene.

Project Window

The Project Window mainly stores all asset documents which contain game scripts, prefabs, models, animation models, fonts, physical materials, and GUI skins. The de- signer can create or delete all material related to the project in the Project Window

Cameras

The camera in Unreal is created as a game object in the Hierarchy view and indicated in the Scene window automatically. In a game the camera provides the visible area on the screen, which means the camera provides the height and width of the view, also the depth can be set. The whole visible space by a camera is called the view volume. If an object which is created in the scene is not placed inside the view volume, it cannot be seen on the screen. The shape of the view volume can be adjusted to orthographic or perspective. These views are constructed from an eye position which could be imag- ined with the location of the viewer, a near clipping three-dimensional space.

4.1.4 Autodesk Maya

Autodesk Maya is used to design 3-D models for video games, animated series, TV series and also it is used to produce visual effects. Autodesk Maya produces dynamic environments for video games. In order to produce games, Maya provides different modes.

Autodesk Maya is best known for Modeling and Animation. But Maya is not open source, this is paid software. We used Maya for designing our game environment and gave animation to characters.

Maya's User Interface (UI)

1. Menu Sets

Menu sets divide the type of menus available into categories: Modeling, Rigging, Animation, FX, and Rendering.

2. Menus

The menus contain both tools and actions for working in your scene.

3. Status Line

The status line contains icons for some commonly-used general commands. A quick Selection field is also available for you to set up for numeric input.

4. User Account menu

Log in to your Autodesk account. Use this to manage account settings.

5. Shelf

The Shelf contains icons for common tasks, organized by tabs based on category. The real power of shelves, however, is that you can create custom shelves, and then make tools or command shortcuts that are quickly accessed from there with a single click.

6. Workspace selector

Select an arrangement of windows and panels designed for different workflows.

7. Sidebar icons

The icons at the right end of the Status line open and close tools that you will use frequently.

8. Channel Box

The channel box lets you edit attributes and key values for selected objects. The Transform attributes are shown by default, but you can change which attributes are displayed here.

9. Layer Editor

There are two types of layers that are displayed in the layer editor:

- Display layers are used to organize and manage objects in a scene.
- Animation layers are used to blend, lock, or mute multiple levels of animation.

10. View panel

The view panel offers different ways of viewing the objects in your scene with a camera view.

11. Tool Box

The tool box contains tools that you use to select and transform objects in your scene.

12. Quick layout/Outliner buttons

The upper three quick layer buttons below the Tool Box let you switch between useful View panel layouts with a single click, and the bottom button opens the outliner.

13. Time Slider

The time slider shows you the time range that is available as defined by the range slider, below. The time slider also displays the current time, and the keys on selected objects or characters.

14. Range Slider

The range slider lets you set the start and end time of the scene's animation.

15. Playback controls

The playback controls let you move around time and preview your animation as defined by the time slider range.

16. Animation/Character menus

The Animation or Character menus let you switch the animation layer and the current character.

4.1.5 Photoshop

Photshop is used to edit photos and also to design 2-D models and provide animation to 2-D models. Photoshop is also paid software and requires a license to use this software. We used Photoshop to provide texturing to models and which makes the gaming environment more realistic. For texturing first we require a UV set of models, designed in Maya, then texture will be set in photoshop, which will further set on UV sets.

4.2 Coding Standards of Language Used

This is a set of coding standards for C++ and Blueprint Visual Scripting language with the aim to help C++ programmers to write simpler, more efficient, more maintainable code.

1. Write clean code that expresses the idea behind it since compilers do not read comments.
2. Do not write code that leaks resources. Even a slow growth in resources will, over time, exhaust the availability of those resources. This is particularly important for long-running programs, such as a game.
3. Be efficient. Do not waste time or space.

4. However, time and space that is spent well to achieve a goal (e.g., speed of development, resource safety, or simplification of testing) is not wasted.
5. Using a well-designed, well-documented, and well-supported library saves time and effort; its quality and documentation are likely to be greater than what a developer could do if the majority of their time must be spent on an implementation.

4.3 Project Scheduling

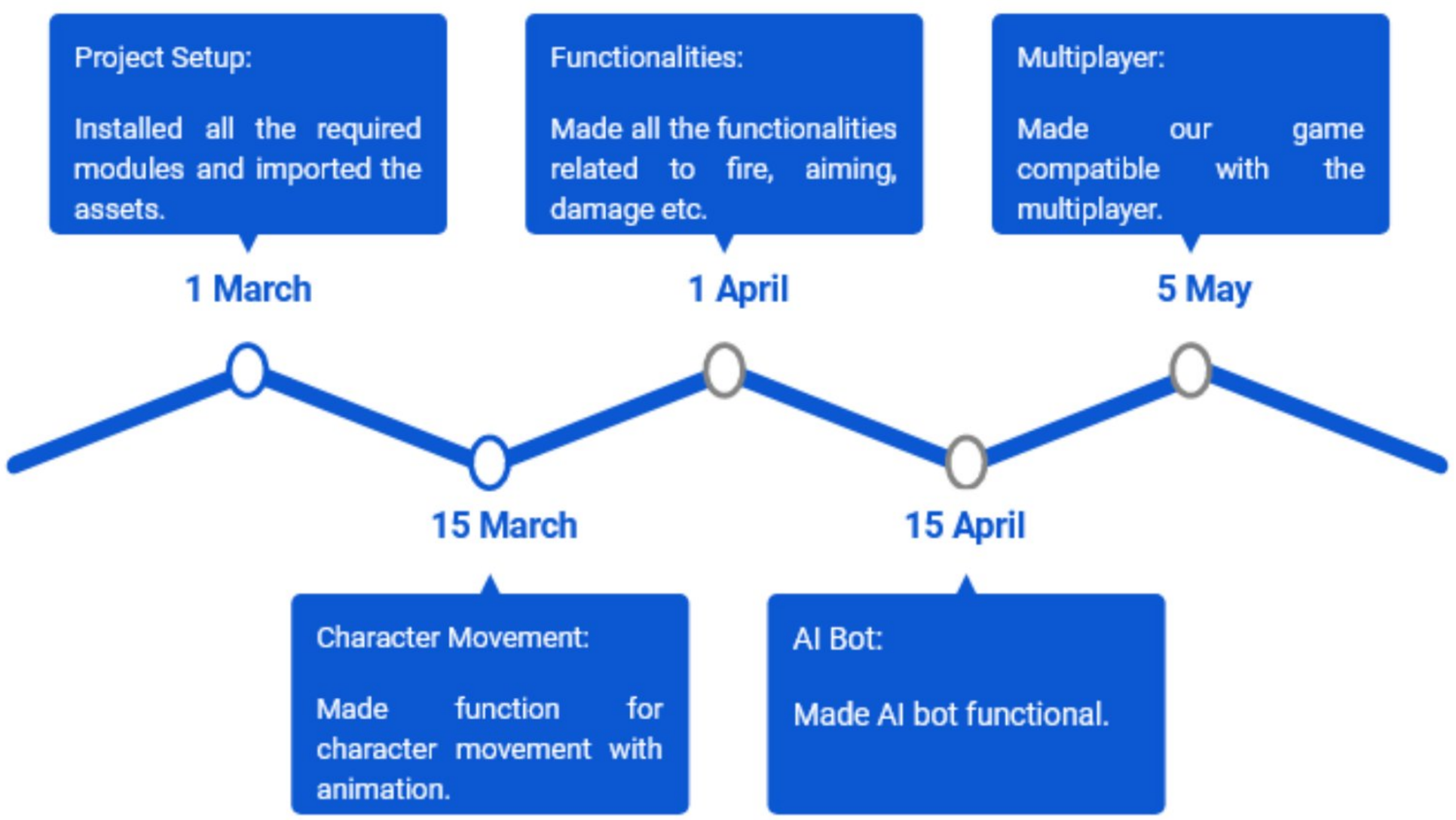

Fig 4.1: Project Scheduling

Table 4.1 : Project Progress Milestones

Sr. No.	Project Activity	Description	Date of Completion
1.	Setup project	- Installed the required modules - Imported the required assets.	01 March
2.	Enabled Movement	- Added player characters - Enabled character movement on user keypress.	07 March
3.	Added Animation	- Added animations to player character movement for realism.	15 March
4.	Added Weapons Functionality	- Added gun models in the game - Attached models to player characters and enemy models - Enabled weapons firing	22 March
5.	Expanded Player Functionalities	Added the following functionalities - Added crosshair icon for aim - Added fire on keypress - Enabled health for player and enemy	30 March
6.	Added AI Enemy Bots	Added AI enemy bots with following functionalities - Patrolling area - Fire when player in sight - Give chase to fleeing player	07 April

7.	Enabled Win/Loss	- Loss if player dies/time runs out - Win if all enemy bots defeated - Added win/loss screens in UI	15 April
8.	Enabled Multiplayer	- Enabled creating a temporary session on local network - Allowed entry to more players in the local network to enable multiplayer.	5 May

4.4 Test Plans

Features to be tested

- Player character movement as per user input
- Weapon aiming as per user input
- Weapon firing as per user input
- Enemy bot behavior.
- Responsiveness of the system in a multiplayer session.

Pass/Fail Criteria

- Pass, if the player character moves only according to user input.
- Pass, if the player character aims a weapon only according to user input.
- Pass, if the player character shoots a weapon only according to user input.
- Fail, if the player character does not follow user input.
- Pass, if the enemy bot chases and shoots the player character.

- Fail, if the enemy bot does not notice/comprehend the player character.
- Fail, if the enemy bot can not aim or shoot at the player character.
- Pass, if the system can show the same game situation on different devices for different players.
- Fail, if there is a difference in game display to any player.

Test Cases

Here are some test cases for the game to check if the game works properly in various situations. We are giving four test examples for four different situations here.

Test Case 1

Test Case : This test will check if the animation is working correctly.

Test Procedure : Import a character model with animation in unreal. Place the character in the scene. Run the game.

Expected Result : Animation works perfectly in the environment.

Actual Result : Animation is not working.

Comment : Need to check character configuration in inspector window. The appropriate animation was not selected. Select it.

Conditional Test : Again run scene.

Expected Result : Animation is working now.

Actual Result : Working perfectly.

Test Case 2

Test Case : This test will check if the AI bots mechanic for hiding behind objects is working correctly.

Test Procedure : Add an AI script to enemy bots in order to recognize the objects and interact. Run scene.

Expected Result : Enemy bots hide behind objects at regular intervals.

Actual Result : Run time exception

Comment : Need to add exception handling in the scripts for the objects and bots.

Conditional Test : Run scene.

Expected Result : Interaction is ok now.

Actual Result : Interaction is ok now.

Accuracy : Perfectly accurate.

Test Case 3

Test Case : This test will check if the weapon firing mechanic is working.

Test Procedure : Add gun to character in the scene. Run scene.

Expected Result : Gun appears in the correct position and can be fired.

Actual Result : Working perfectly

Comment : Weapons are working as expected.

Test Case 4

Test Case : This test will check if the multiplayer works.

Test Procedure : Create a game session on one device. Use the device IPv4 address on another device. Run scene.

Expected Result : Session is created and another player can join.

Actual Result : Working perfectly

Comment : Multiplayer is implemented.

Chapter 5. Result And Discussions

5.1 User Interface Representation

This section shall detail the different modules and classes implemented to make the game interactive for users.

5.1.1 Description of Various Modules

In this section, we shall share the modules implemented in game and describe the functions and responses to stimuli of each module.

5.1.1.1 Player Character Module

Stimulus/Response Sequence

1. If the user presses arrow keys, the player character moves in-game.

2. If the user uses a mouse to move the game camera, the player character aims the weapon accordingly.

3. If the player character receives damage from enemy bots, the health stat decreases.

4. If the health of a player character reaches 0, it dies and is removed from the game.

Functions

1. Player character module associates with the model to render on display.

2. It controls character movement.

3. It controls the health stat of the player character.

4. It actively equips weapon models with character models.

5. It aims the weapon at the center of the game camera.

6. It stores game information specific to the user, for example, enemies killed.

5.1.1.2 Enemy Bots Module

Stimulus/Response Sequence

1. Enemy bot will be controlled by the system according to the behavior tree.

2. If the enemy bot receives damage from a player character, the health stat decreases.

3. If the health of an enemy bot reaches 0, it dies and is removed from the game.

Functions

1. Enemy character module associates with the model to render on display.

2. It controls enemy character movement.

3. It controls the health stat of the enemy character.

4. It actively equips weapon models with character models.

5.1.1.3 Game Mode Module

Stimulus/Response Sequence

1. If a game session is created, game mode is automatically activated.

2. If a player is added to the game session, it automatically adjusts game rules.

3. If an enemy bot is killed, game mode rewards points to player characters.

Functions

1. It maintains a list of the players in the game session.

2. It maintains game rules.

3. It controls the win and loss conditions.

4. It stores game information, for example, enemies killed and points awarded.

5.1.1.4 Player Controller Module

Stimulus/Response Sequence

1. If the player dies, the player controller informs the game mode.

2. If a user is added in the game session, it automatically assigns a unique player controller to the player character associated with the user.

Functions

1. A unique player controller module instance is associated with every player character .

2. It controls player character state.

3. It controls the death status of a player character.

5.1.1.5 Player State Module

Stimulus/Response Sequence

1. It only functions if multiplayer mode is enabled.

2. If any player character kills any enemy bots, it increases its respective score.

Functions

1. Player state module associates with the player character in multiplayer mode.

2. It maintains the score of each respective player character.

5.1.1.6 Weapon Module

Stimulus/Response Sequence

1. If the user clicks the left mouse button, the player character fires the weapon in-game.

2. If a player character fires the weapon, it is aimed towards the center of the game camera.

3. If the player character fires the weapon, it spawns a visual fire effect and sound.

Functions

1. Weapon module associates the weapon model with the Player character model to render the weapon on display.

2. It controls weapon fire.

5. It aims the weapon at the center of the game camera.

5.1.1.7 AI Module

Stimulus/Response Sequence

1. If a player character enters the view of an enemy bot, AI controls the enemy bot to chase the player.

2. If a player character is inside the view of an enemy bot, AI fires the enemy bot's weapon on it.

3. If other enemy bots are in close vicinity of an enemy bot, the individual AI modules associated with each will cooperate to attack the user better.

4. If the health of an enemy bot becomes critically low, AI controls it to hide and recover.

Functions

1. A unique AI module associates with each enemy bot module when the game session starts.

2. It controls enemy bot movement.

3. It maintains the health stat of the enemy bot through evasive measures.

4. It aims the weapon of the enemy bot at a player character in the view.

5. It commands the enemy bot to chase any fleeing player characters to attack and kill them.

5.1.2 Classes implemented in Game

Table 5.1 : Classes

Sr. No.	Class Name	Responsibility
1.	SCharacter	Responsible for player movement, shooting, checking death conditions for the player, etc.
2.	SCharacter.h	Contains all the functions and variables used in the above file.
3.	EnemyBots	Responsible for enemy bot movement, shooting, checking death condition etc.
4.	SGameMode	Responsible for showing the HUD to the screen according to the condition.
5.	SGameMode	Contains all the functions and variables used in the above file.
6.	SPlayerController	Responsible for telling the GameMode when the game has ended.
7.	SPlayerController.h	Contains all the functions and variables used in the above file.

8.	SPlayerState	Responsible for assigning the scores to the respective player in the current session.
9.	SPlayerState.h	Contains all the functions and variables used in the above file.
10.	SWeapon	Responsible for all the functionality of the gun i.e - bullet damage, fire rate, fire function, effects etc.
11.	SWeapon.h	Contains all the functions and variables used in above class.
12.	AIEnemyBots	Responsible for controlling the AI enemy to perform some specific task like - recognize player, move to player, shoot at player, take cover etc.

5.2 Snapshots of System

Fig 5.1: Main Menu

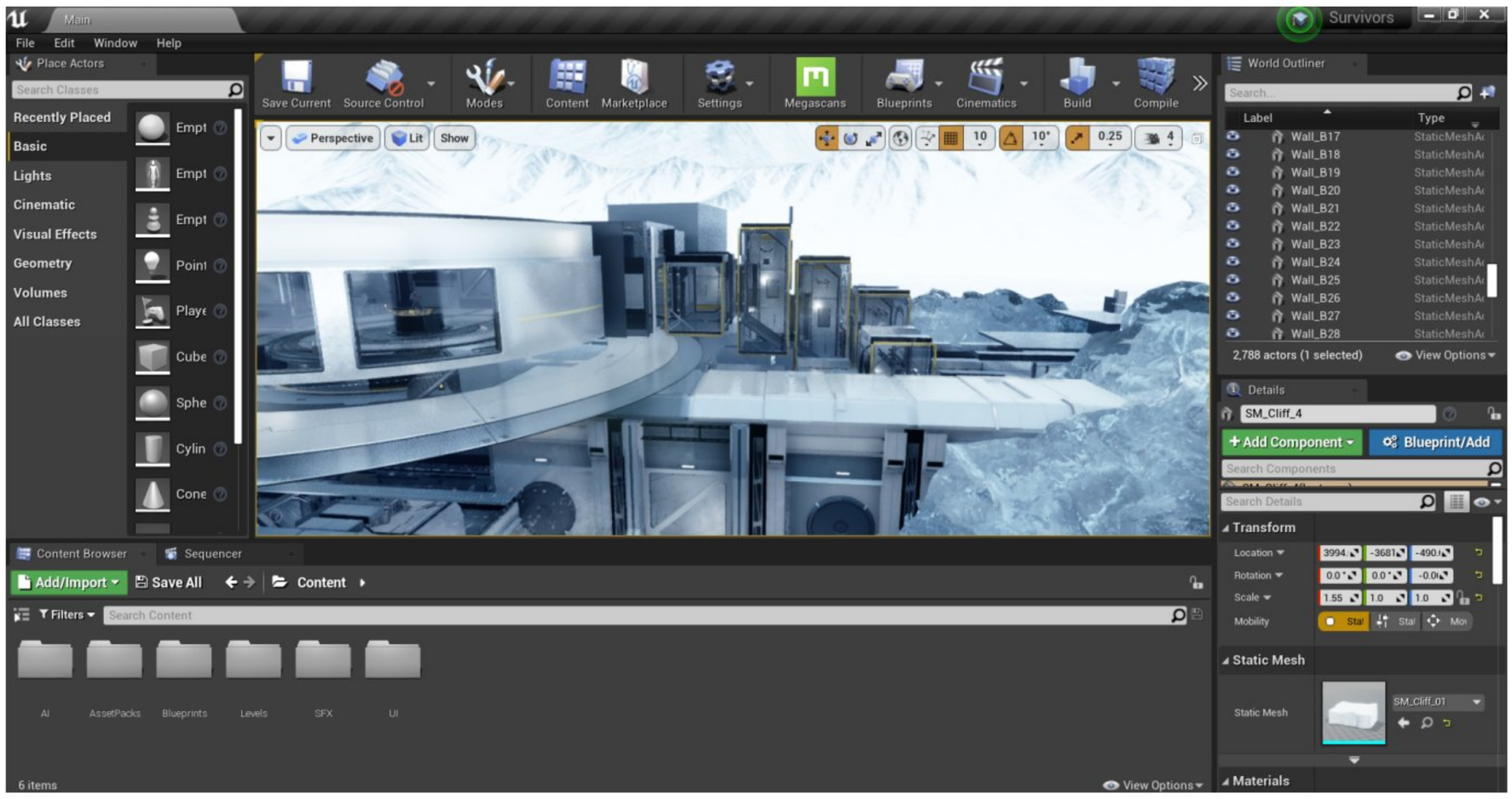

Fig 5.2: Level 1

Fig 5.3: Level 2

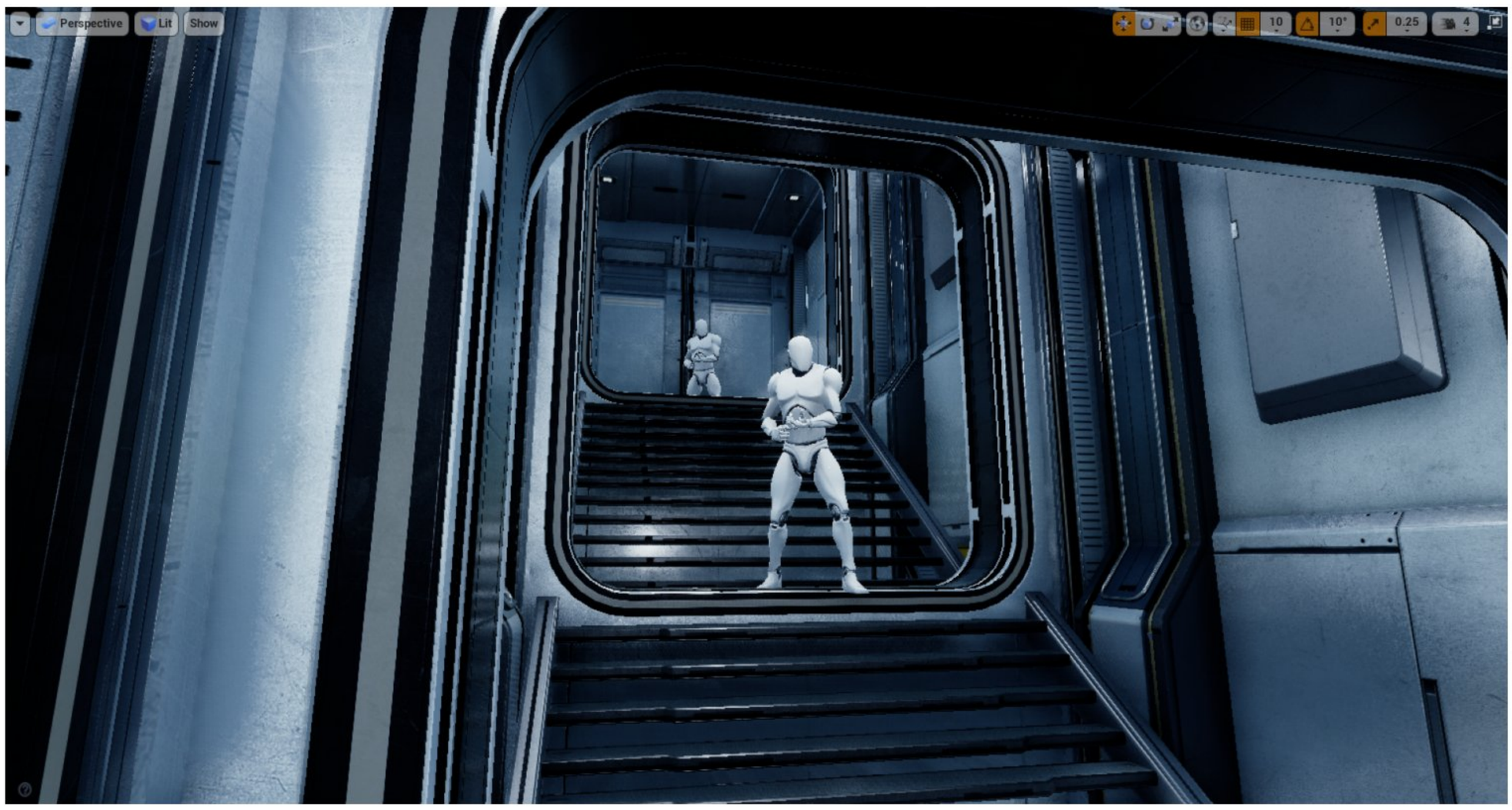

Fig 5.4: Level 3

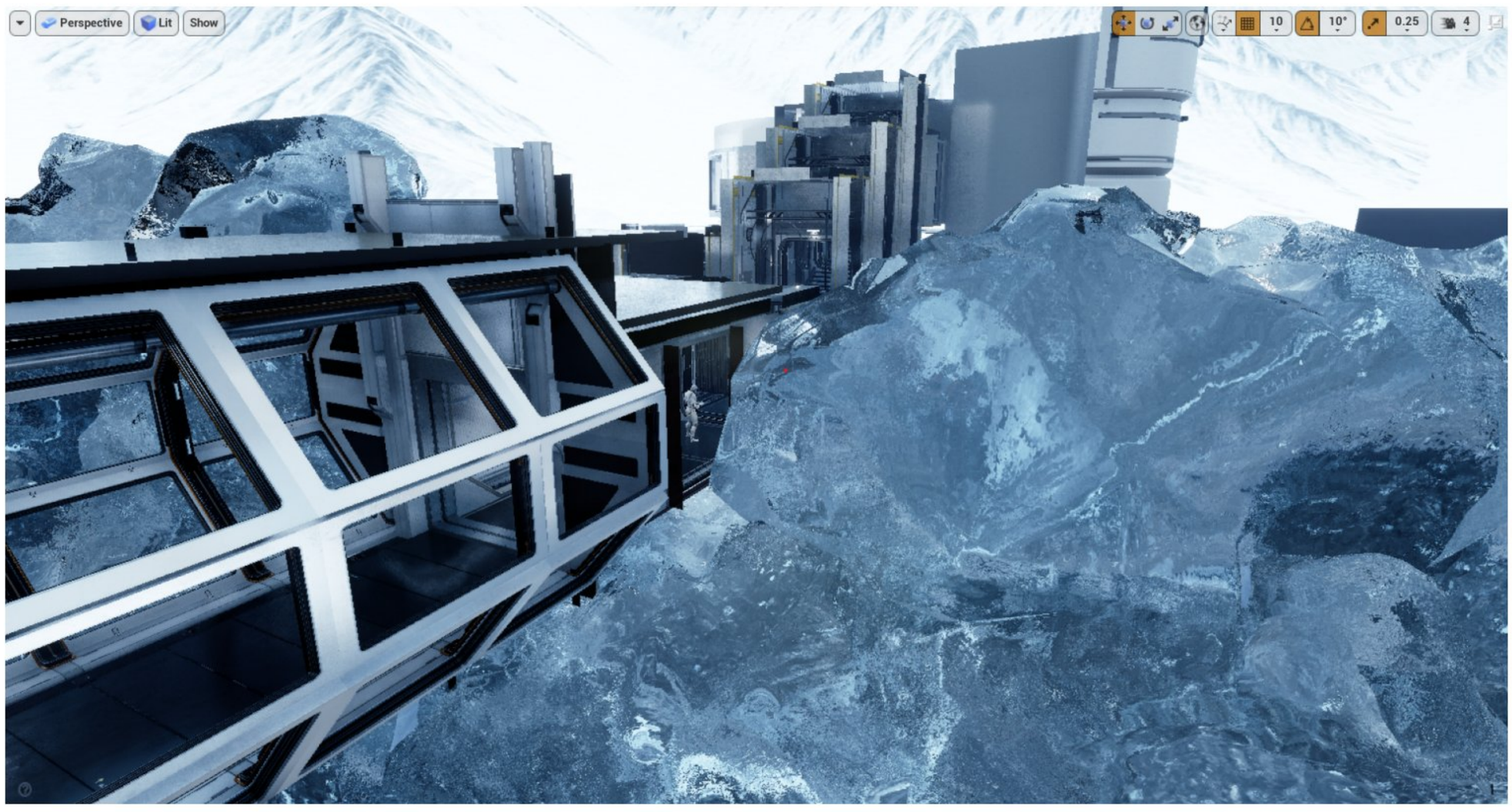

Fig 5.5: Level 4

Fig 5.6: Level 5

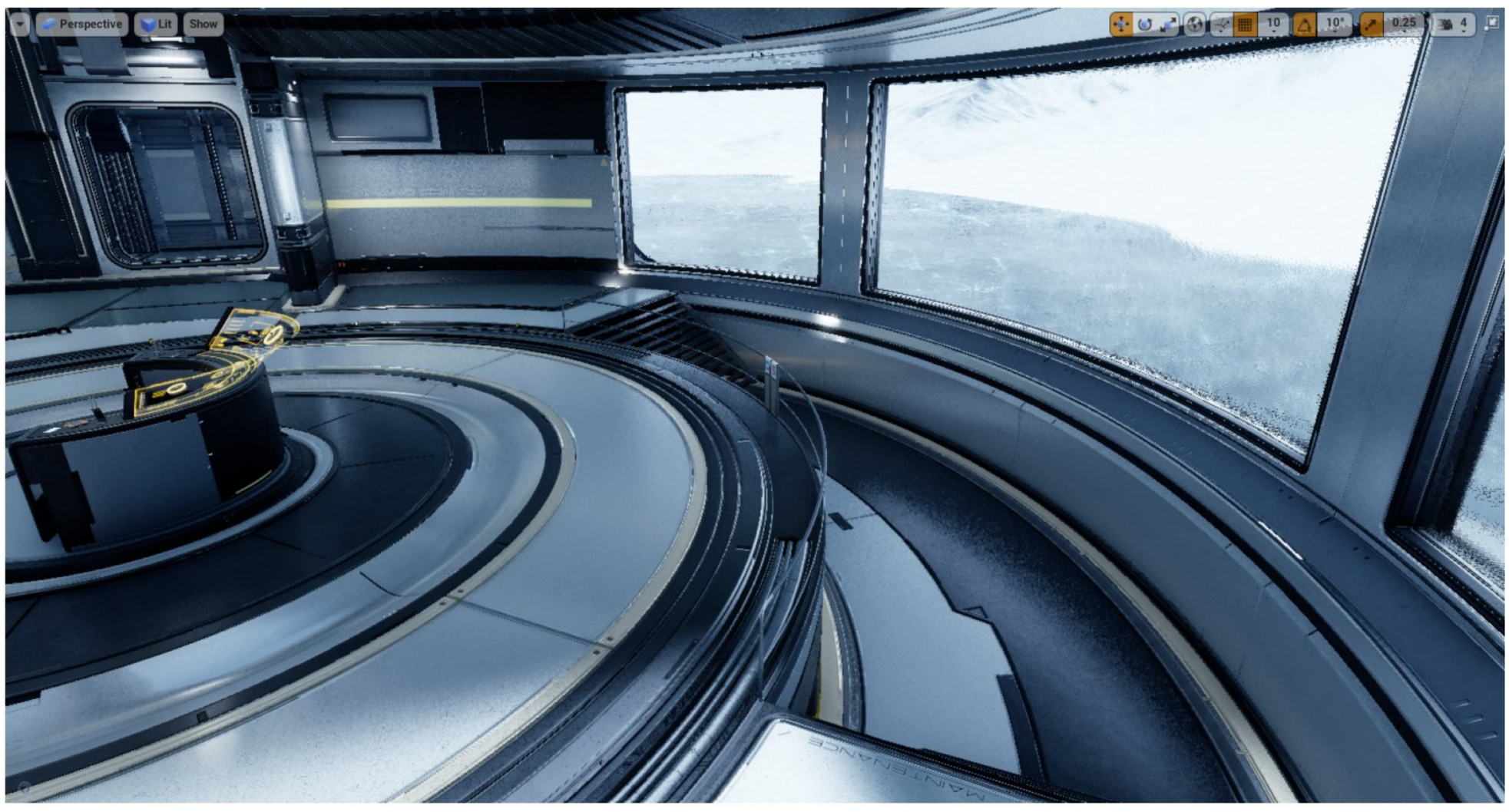

Fig 5.7: Level 6

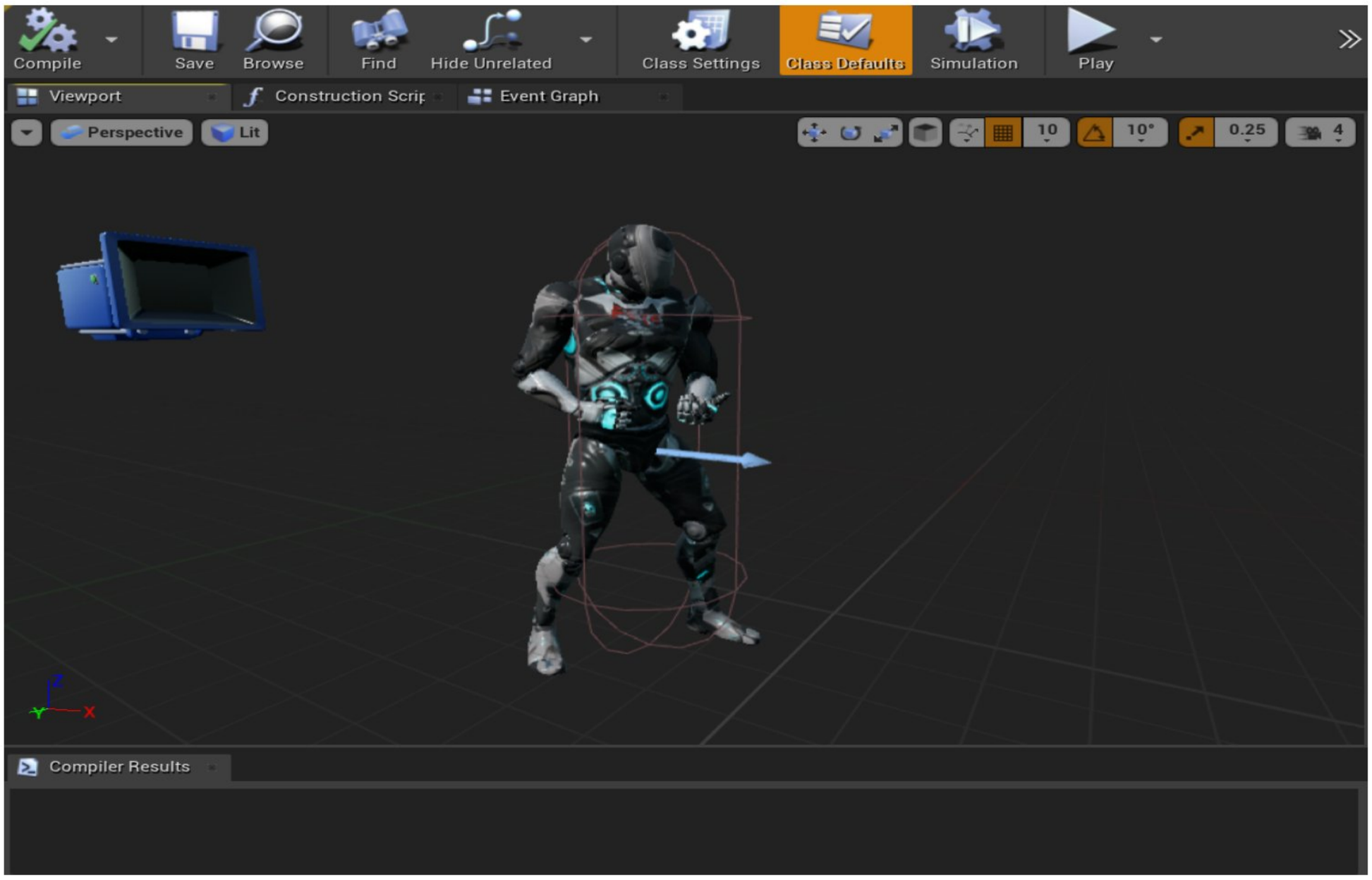

Fig 5.8: Player

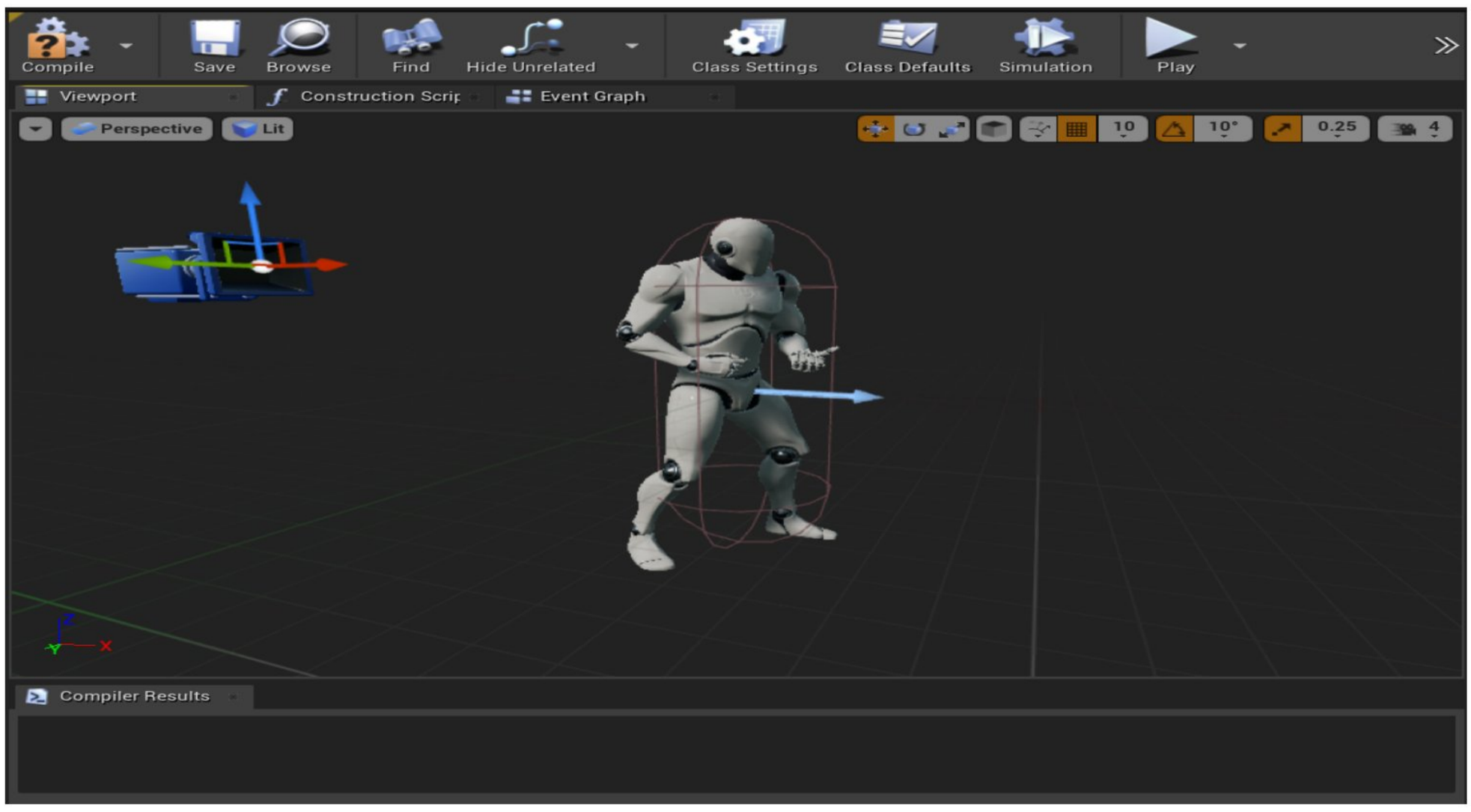

Fig 5.9: Enemy

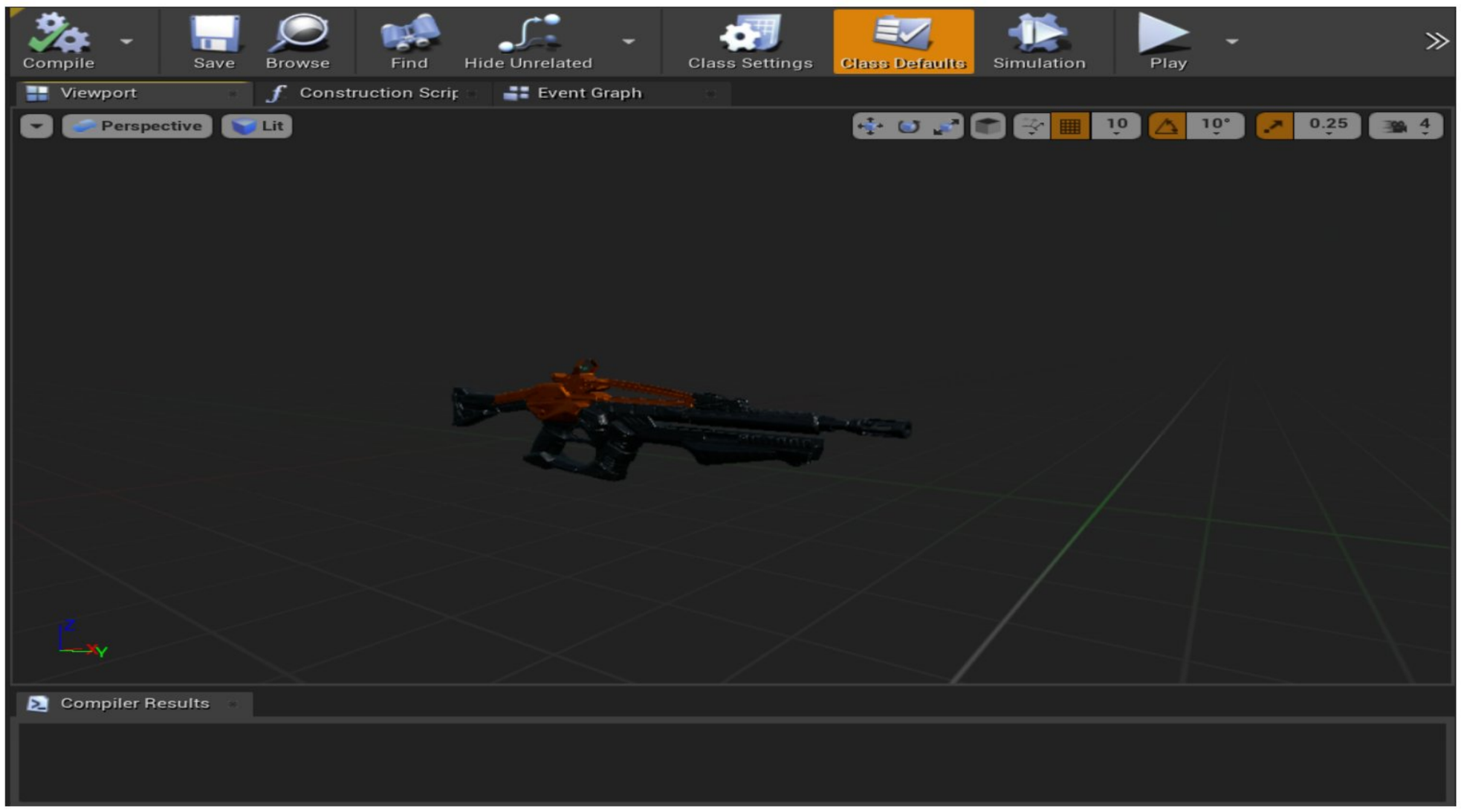

Fig 5.10: Gun

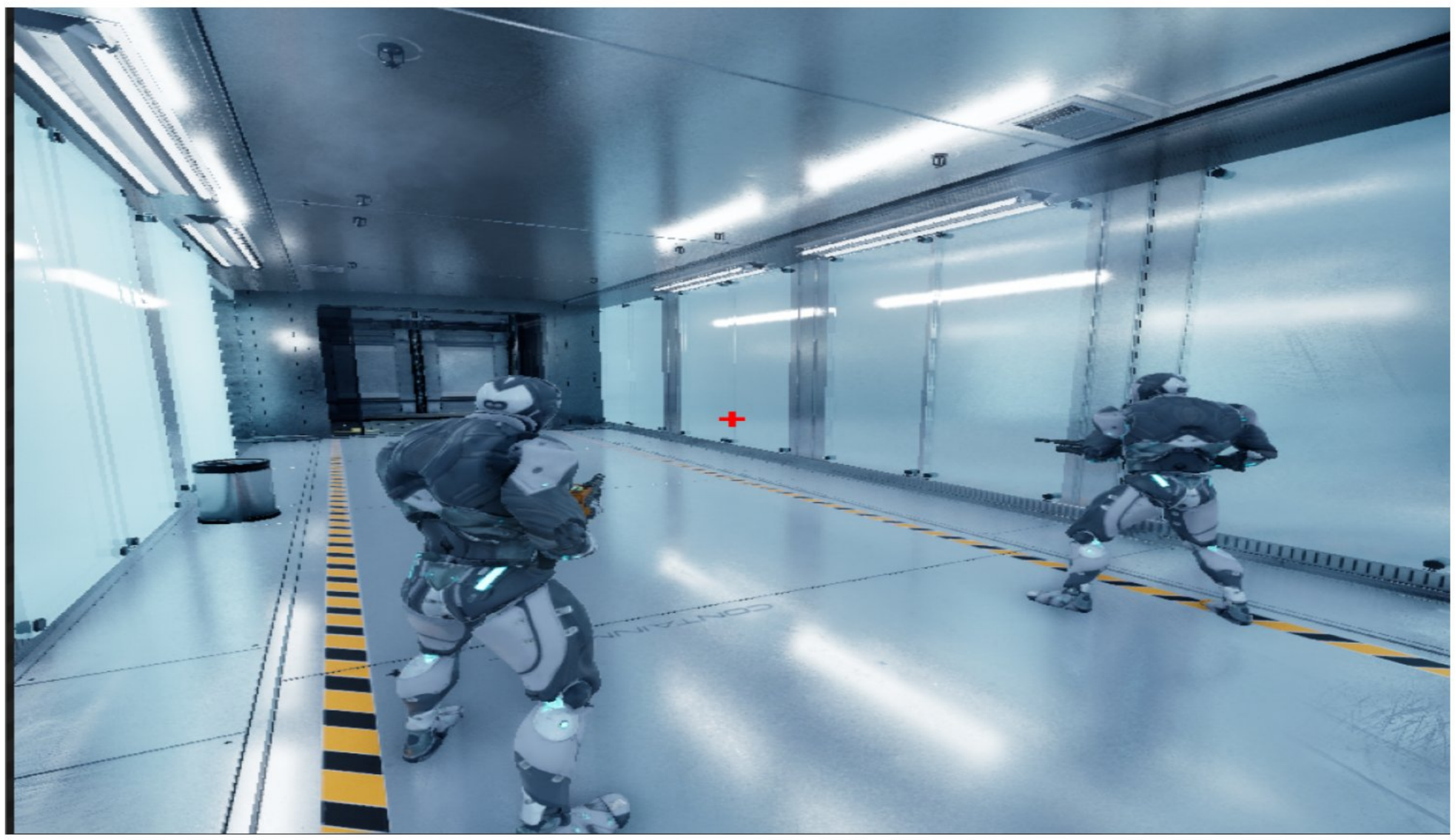

Fig 5.11: Multiplayer 1

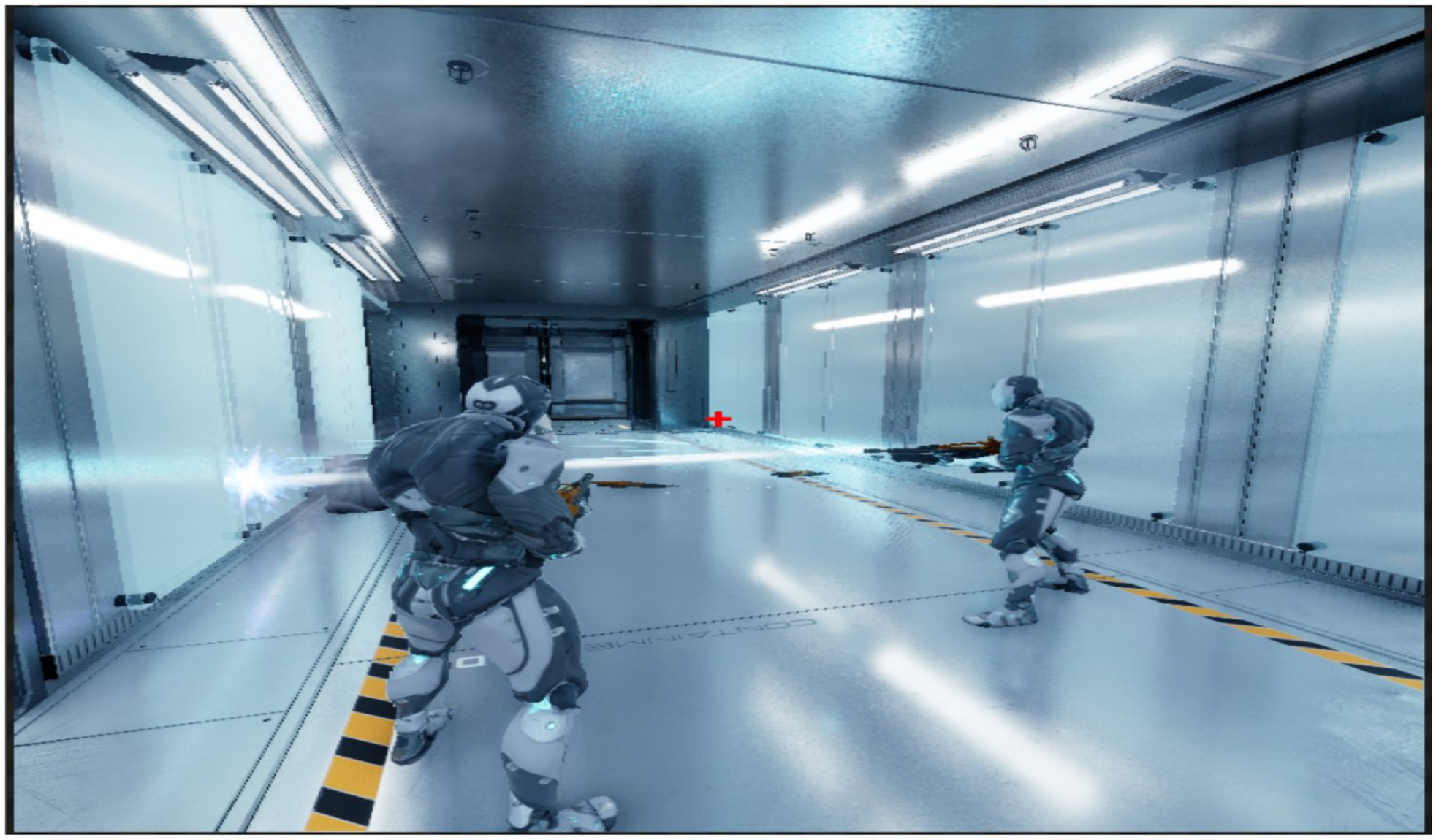

Fig 5.12: Multiplayer 2

Fig 5.13: Multiplayer 3

Chapter 6: Conclusion and Future Scope

6.1 Conclusion

The Survivors is a 3D game in which single or multiple players fight/survive using weapons (primarily guns) against a random number of artificially intelligent enemy bots in a virtual environment map until either all bots are eliminated or all players are eliminated.

The players can move freely in a large virtual 3D environment. Players have to save their health and defeat the enemy bots in order to win the game.

The enemy bots will utilize AI to accurately attack the player(s), work together to eliminate the players, and even hide behind various map elements in order to protect themselves from player attacks.

Multiplayer will be achieved through the use of temporary game servers hosted on a local network connection. A player has to initially create a game server, and then multiple players can join that server using the initial player's IP address.

In this game, we have implemented excellent graphics with player characters, enemy bots, the virtual game environment, weapons and the HUD. We have also implemented Artificial Intelligence in enemy bots in order to level the playing field and allow the game to feel realistic. Multiplayer has been achieved through the use of a local network server to allow players on the same network to play together.

6.2 Future Scope

While the game performs very well as per the initial requirements, it needs many additions before it is ready to compete with existing market leaders, such as:

- A cloud server to allow multiple players globally to play the game together.
- A database backend to keep track of player wins/losses.
- More game environments or maps.
- Player character skin modification
- To add different types of enemies
- To deploy this game on more servers
- To add more weapons to the game.
- To allow usage of environment elements for attack.
- To enable first-person aiming.
- To add some premium weapons or player skins in future the player may need to pay to acquire them, which is a choice for the players and is not a must.

REFERENCES/BIBLIOGRAPHY

1. Autodesk.com. 2022. *Autodesk empowers innovators everywhere to make the new possible*. [online] Available at: <https://www.autodesk.com>
2. Unreal Engine. 2022. *The most powerful real-time 3D creation tool*. [online] Available at: <https://www.unrealengine.com>
3. Adobe: *Creative, marketing and document management solutions*. [online] Available at: <https://www.adobe.com>
4. Programming with C++ on Unreal Engine [online] Available at: <https://docs.unrealengine.com/4.27/en-US/ProgrammingAndScripting/ProgrammingWithCPP/>
5. Tyagi, Shivam & Sengupta, Sudhriti. (2020). *Role of AI in Gaming and Simulation.*
6. El Rhalibi, A. & Wong, Kok & Price, Marc. (2009). *Artificial Intelligence for Computer Games*. Int. J. Computer Games Technology. 2009.
7. Fan, Xueman & Wu, J. & Tian, L.. (2020). *A Review of Artificial Intelligence for Games*.
8. Torres-Ferreyros, Carlos & Festini-Wendorff, Matthew & Shiguihara, Pedro. (2016). *Developing a videogame using unreal engine based on a four stages methodology*.