

Tunisian Republic
Ministry of Higher Education
And Scientific Research

University of Sfax
Higher Institute of Computer Science and
Multimedia in Sfax



*License in Computer Science
And Multimedia*

N° d'ordre : **2024-162**

END OF STUDIES PROJECT

Presented to

**Higher Institute of Computer Science and Multimedia in
Sfax**

In order to obtain the

LICENSE

In Computer Science and Multimedia

VIDEO GAME DEVELOPMENT

By

Rami Chaaben

Soutenu le 30/05/2024, in front of the jury composed of

Mr. Hamrouni Zied Meddeb

President

Ms. Abdelmoulah Ons

Reporter

Mr. Ghorbel Hatem

Supervisor

Mr. AmenAllah Ben Achour

Company Guest

2023-2024

Dedication

From the bottom of my heart, I dedicate this work to the following people: My cherished family, who has not only helped to make this project possible, but who has also served as my mentors, my guides, my instructors, and my friends. I cannot begin to articulate the love and appreciation I have for everything you have done. Their unwavering faith in me and their unwavering commitment to extending a helping hand whenever I needed it have been priceless. I am deeply grateful to have them as an integral part of my life, guiding and supporting me in every step of the way. Their presence has been an immeasurable gift, and I cannot express my gratitude enough for their unwavering belief in me. Together, my family and these remarkable individuals have been my pillars of strength, and without their encouragement and assistance, this achievement would not have been possible. Thank you, from the bottom of my heart, for being there every step of the way.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Mr. Hatem Ghorbel, Assistant Professor at ISIMS, for his dynamism, determination, and enthusiasm in directing me, as well as for his patience and fruitful advice which led to the successful completion of this work.

I would also like to thank the distinguished members of the jury who have honored me by examining and evaluating this work.

Finally, I would like to thank the staff and personnel of the Higher Institute of Computer Science and Multimedia in Sfax for the daily efforts they make to improve working conditions during our training period.

Table of Contents

Chapter I: General project context.....	
1: Introduction	11
2: Company Presentation.....	11
2.1: Presentation of CGI Studio	11
2.2: Activities of CGI Studio	12
2.3: Organization of CGI Studio	12
3: Project presentation	13
3.1: Inspirations.....	13
3.2: Work methodology and planning.....	14
3.2.1: Adopted methodology: Agile	15
3.2.2: The Scrum methodology.....	15
3.2.3: Why Scrum for videogame development	15
3.2.4: The actors of Scrum in the project	16
4.1: Work Environment	16
4.2: Software environment	16
4.3: Programming and modeling languages.....	17
Conclusion.....	18
Chapter II: Specification of requirements and general architecture.....	
1: Introduction	19
2: Game summary.....	19
3: Actor's specification.....	19
4: Backlog Product	20
5: Functional and non-functional needs.....	21
5.1: Functional needs	21
5.2: Non-functional needs	22
6: Modeling of functional needs.....	23
6.1: General architecture of Unity.....	23
6.2: General use case diagram	24
6.3: General class diagram	26
Conclusion	26

Chapter III Sprint 1: Game mechanics and the player character

1: Introduction	27
2: Backlog	27
3: Use Case Diagram	28
3.1: Use case of "Move"	29
3.2: Use case of "Control Camera"	30
3.3: Use case of "Collect Loot"	30
3.4: Use case of "Change Weapons"	31
3.5: Use case of "Attack"	32
3.6: Use case of "Get into a vehicle"	32
3.7: Use case of "buy Items"	33
3.8: Use case of "Upgrade Stats"	34
4: Game mechanics	35
4.1: Activity diagram.....	35
4.2: Missions.....	37
4.2.1: Main missions	37
4.2.1.1: Checkpoint	37
4.2.1.2: Radio tower	38
4.2.1.3: Outpost	40
4.2.2: Side missions.....	41
4.2.2.1: Races	41
4.2.2.2: Collectibles	43
4.3: Progression system.....	43
5: Sprint 1 realization	45
5.1: Unity asset store.....	45
5.2: Player character.....	45
5.3: Game's Scene	46
5.4: Gameplay.....	49
Conclusion	51

Chapter IV Sprint 2: AI Character

1: Introduction	52
2: Backlog	52
3: Use Case Diagram	53
3.1: Use case of "Move"	54
3.2: Use case of "Attack"	55

ISIMS	
3.3: Use case of “Stay Idle”	55
4: Sprint 2 realization	56
4.1: Artificial Intelligence in videogames.....	56
4.2: AI uses in “Wild World”	56
4.2.1: Movement and navigation between destinations.....	56
4.2.2: Attack, Aim, and Shoot	57
4.3: AI animation	58
Conclusion	59
Chapter V Sprint 3: Main Menu.....	
1: Introduction	60
2: Backlog	60
3: Use Case Diagram	60
4: Sprint 3 realization	61
4.1: Main Menu	61
4.2: Help Menu	62
Conclusion	62
General Conclusion	63
Webography	64

Table of Figures

Figure 1- Company's logo	11
Figure 2- Organization of CGI Studio	12
Figure 3- Minecraft's logo	14
Figure 4- Far Cry's logo	14
Figure 5- Scrum Process	15
Figure 6- Unity architecture's diagram	23
Figure 7.1- General use case diagram	24
Figure 7.2- General use case diagram	25
Figure 8- General class diagram	26
Figure 9- Use case diagram of player character	28
Figure 10- Use case diagram of "Move"	29
Figure 11- Use case diagram of "Control Camera"	30
Figure 12- Use case diagram of "Collect Loot"	30
Figure 13- Use case diagram of "Change Weapons"	31
Figure 14- Use case diagram of "Attack"	32
Figure 15- Use case diagram of "Get into a vehicle"	32
Figure 16- Use case diagram of "Buy Items"	33
Figure 17- Use case diagram of "Upgrade Stats"	34
Figure 18- Activity diagram	36
Figure 19- Checkpoint	37
Figure 20- Radio tower	38
Figure 21- Satellite dish	39
Figure 22- Outpost	40
Figure 23- Racetrack 1	41
Figure 24- Racetrack 2	42
Figure 25- Chest	43
Figure 26- Upgrade menu	44
Figure 27- Unity asset store	45
Figure 28- Low poly ultimate pack	45
Figure 29- Starter assets - first-person	46
Figure 30- Game's map	47
Figure 31- Example of rural areas	47
Figure 32- Example of urban areas	48
Figure 33- Standard Assets	49

Figure 34- Input system	49
Figure 35- User Interface	50
Figure 36- Use case diagram of AI character	53
Figure 37- Use case diagram of “Move”	54
Figure 38- Use case diagram of “Attack”	55
Figure 39- Use case diagram of “Stay Idle”	55
Figure 40- NavMesh	57
Figure 41- Example of AI character aiming	58
Figure 42- Example of AI character animation	59
Figure 43- Use case diagram of the user	60
Figure 44- Main menu	61
Figure 45- Help menu	62

Table of Tables

Table 1- Table of work environment	16
Table 2- Table of software environment	16
Table 3- Table of programming and modeling languages	18
Table 4- Table of backlog product	20
Table 5- Table of Sprint 1 backlog	27
Table 6- Table of use case “Move”	29
Table 7- Table of use case “Control Camera”	30
Table 8- Table of use case “Collect Loot”	31
Table 9- Table of use case “Change Weapons”	31
Table 10- Table of use case “Attack”	32
Table 11- Table of use case “Get into a vehicle”	33
Table 12- Table of use case “Buy Items”	33
Table 13- Table of use case “Upgrade stats”	34
Table 14- Table of checkpoint mission.....	38
Table 15- Table of outpost mission	40
Table 16- Table of race 1 mission	41
Table 17- Table of race 2 mission	42
Table 18- Table of upgrade menu	44
Table 19- Table of user interface.....	50
Table 20- Table of Sprint 2 backlog	52
Table 21- Table of use case “Move”	54
Table 22- Table of use case “Attack”	55
Table 23- Table of use case “Stay Idle”	56
Table 24- Table of Sprint 3 backlog	60
Table 25- Table of use case of the user	61
Table 26- Table of main menu	61

General Introduction

Videogames are a form of interactive digital entertainment played using electronic devices such as computers, consoles, or mobile phones. They are characterized by their use of digital graphics, sound, and often complex narratives and gameplay mechanics. The development of videogames has evolved significantly since their inception in the early 1970s, moving from simple, text-based adventures and arcade games to the immersive, visually stunning, and highly interactive experiences available today.

Videogames can be played solo, cooperatively, or competitively, either locally or online. They are enjoyed by a diverse audience, spanning all age groups and demographics. The industry has grown into a massive global enterprise, influencing culture and even education, with applications in training, therapy, and more.

Chapter I: General project context

1.Introduction

First-person shooter (FPS) games are a subgenre of action videogames centered around gun and other weapon-based combat in a first-person perspective, where the player experiences the game through the eyes of the protagonist. This perspective creates an immersive experience, making the player feel as though he is in the game world.

2.Company Presentation

2.1 Presentation of CGI Studio



Figure 1: company's logo

This is a professional IT firm specializing in videogame development. The company is attentive to market needs. CGI STUDIO is located at 1 Moussa Ben Noussair Street in Menzel Temime, Nabeul. It was established on July 5, 2018, by Mr. Amen Allah Ben Achour.

2.2 Activities of CGI Studio

- *Mobile Applications
- *3D Animation
- *Augmented reality and virtual reality applications
- *Videogame development
- *Interior design
- *Artificial intelligence

2.3 Organization of CGI Studio

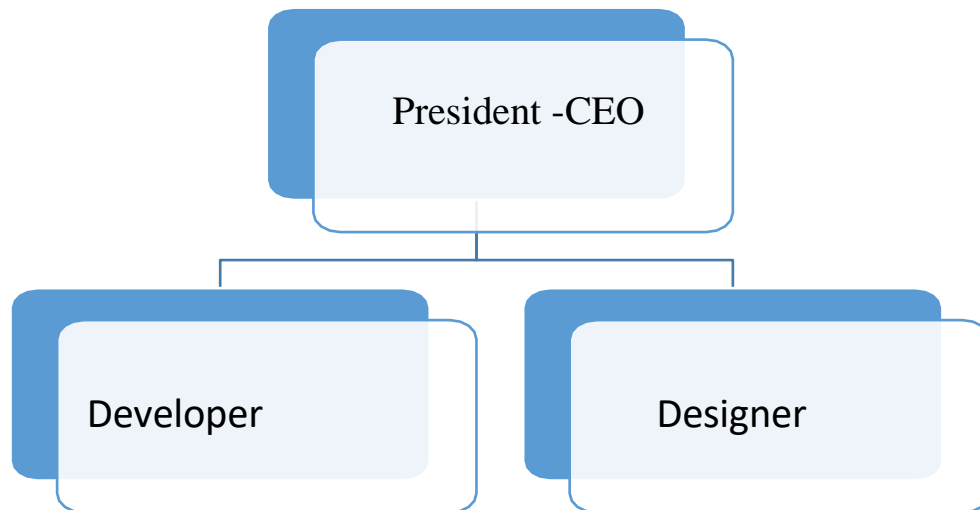


Figure 2: Organization of CGI Studio

President's responsibilities include:

- *Preparing work and implementing the decisions of the board of directors.
- *Representing the company to third parties and being responsible for signing civil, administrative, and judicial documents.
- *Exercising authority over all personnel.

The developer's responsibilities include:

- *Establishing the specifications.
- *Translating the specifications into code with the goal of making it increasingly fast and efficient.
- *Programming the necessary interfaces and associated tools.

The designer's responsibilities include:

- *Interpreting requests made orally or in writing from a specifications document.
- *Imagining potential transformations of submissions.
- *Proposing improvement solutions for the proposals.
- *Creating digital 3D models, animations, and files for machining.
- *Making projects visually understandable.

3.Project Presentation

First-person shooter (FPS) games are a subgenre of action videogames centered around gun and other weapon-based combat in a first-person perspective, where the player experiences the game through the eyes of the protagonist. This perspective creates an immersive experience, making the player feel as though he is in the game world.

In this context, and as part of our final year project, we have decided to develop a 3D game called "Wild World." In this open-world game, the player navigates through a vast, immersive environment filled with diverse landscapes and challenges. Player faces enemies equipped with advanced artificial intelligence, creating a dynamic and unpredictable gameplay experience.

3.1 Inspirations

As with any other form of creation, it is impossible to create a videogame without inspiration. Consequently, we conducted extensive research on existing games like our future product. Our goal is to create a 3D open-world videogame in first-person perspective, centered around themes of adventure and exploration. Research has been conducted to assist us in our project.

Minecraft

Minecraft is a sandbox videogame developed by Mojang Studios. The game was created by Markus "Notch". In Minecraft, players explore a procedurally generated 3D block world with virtually infinite terrain, where they can discover and extract raw materials, craft tools and items, and build structures and machines.[1]



Figure 3: Minecraft's logo

Far Cry

Far Cry is an open-world first-person shooter game series developed by Ubisoft. The series is known for its expansive and immersive environments, compelling storytelling, and dynamic gameplay. Most of the time players play as a protagonist trying to survive in dangerous environments, fight its enemies and defeat their powerful leaders. This Far Cry belongs to the Action-Adventure game series that presents us with both exploration, combat, and survival.[2]



Figure 4: Far Cry's logo

3.2 Work Methodology and Planning

Methodologies impose discipline on software development processes with the goal of making the project more predictable and its execution more efficient. This goal is theoretically achieved by following a rigorous sequence of tasks and adhering strictly to a detailed schedule, which we will present in the following sections. In this section, we will introduce the methodology we have carefully chosen for the development of our project, as well as the reasons why we have chosen it and its advantages specifically for our project. Let us not forget that the objective of this process is to produce a high-quality videogame that meets the needs of users and is delivered within the predefined deadlines and costs.

3.2.1 Adopted Methodology: Agile

The Agile methodology is an approach that enables the realization of a project by breaking it down into small parts. The idea is to achieve the result defined by the project objectives through small, controlled, and practical steps. It has been widely adopted in software development projects due to its pragmatism, which is focused on the outcome and the final product.[3]

3.2.2 The Scrum Methodology

Scrum is an agile process that focuses on delivering valuable outcomes, emphasizing teamwork and accountability. Typically, the size of a Scrum team consists of 5 to 9 experts who collaborate on a project. The Scrum methodology includes a product owner, a Scrum master, and a development team.[4]

Scrum Process

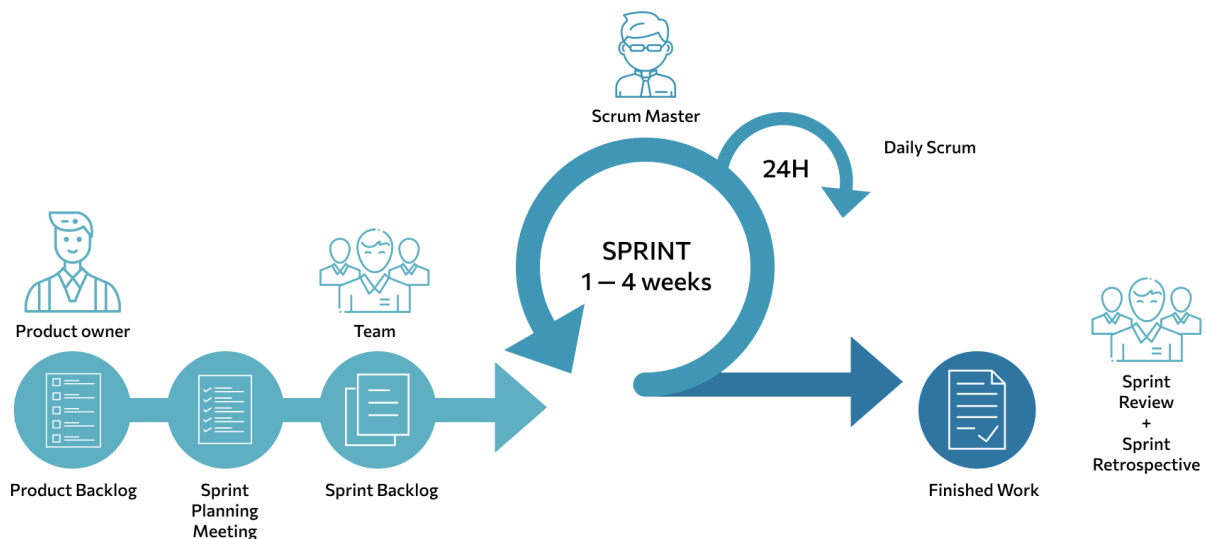


Figure 5: Scrum Process

3.2.3 Why Scrum for videogame development?

When we mention Scrum or agile development, it's about reevaluating plans and creating interactive phases of work. This type of work is highly applied in videogame development projects. The reason is the flexible approach that minimizes the risks of making mistakes. This way, the team can follow market changes and adapt the product they're building to what is required by the target audience.

Moreover, trends within this industry change. Thus, when creating games, you must keep in mind the requirements of the gaming industry because there are always new features to integrate.

Most importantly, by choosing a Scrum team for videogame development, developers become more engaged and focus on creating the game engine without wasting efforts, time, and resources.

3.2.4 The actors of Scrum in the project

The actors of Scrum in the project are:

Product Owner and Scrum Master: Amen Allah BenAchour



Developers: Rami Chaaben

4.1 Work Environment

Device Name	RAM	CPU	GPU
Asus TUF Gaming A15	16GB	AMD Ryzen 5 4600H 4.0GHz	Nvidia GTX 1660 TI And Radeon Graphics
Xiaomi Note 12	6GB	Snapdragon 685 Octa-core 2.8GHz	Adreno 610

Table 1: Table of work environment

4.2 Software Environment

Tool	Description
	Unity, developed by Unity Technologies is an embraced game engine that allows developers to create types of interactive content, including 2D and 3D games as well, as simulations.[5]
	Adobe Photoshop is a recognized software created by Adobe Inc. It is mainly used for design editing photos and creating art. Ever since it was first launched in 1988 Photoshop has become the go to choose for professionals and hobbyists who want to manipulate and improve images create artwork, design graphics and much more.[6]



	<p>Microsoft Visual Studio, created by Microsoft is an integrated development environment (IDE) that offers developers a range of tools, features and resources to simplify software development, testing, debugging and deployment across platforms and technologies.[7]</p>
	<p>Blender is a free and open-source 3D creation suite. It supports the entirety of the 3D pipeline—modeling, rigging, animation, simulation, rendering, compositing, motion tracking, and video-editing. It's used for creating animated films, visual effects, art, 3D printed models, interactive 3D applications, and videogames.[8]</p>
	<p>Diagrams.net is an open-source online application for creating diagrams, an easy-to-use tool that offers many features such as real-time collaborative editing, online storage on Google Drive or OneDrive, and export options to PNG, JPG, SVG, and more.[9]</p>
	<p>Mixamo a 3D computer graphics technology company. Based in San Francisco, the company develops and sells web-based services for 3D character animation. Mixamo's technologies use machine learning methods to automate the steps of the character animation process, including 3D modeling to rigging and 3D animation.[10]</p>

Table 2: Table of software environment

4.3 Programming and Modeling Languages



Tool	Description
	<p>C# (commonly known as "C sharp") is a programming language that was created by Microsoft as a component of their.NET initiative.[11]</p>
	<p>UML, or Unified Modeling Language, is a standardized modeling language used in software engineering to visualize, specify, construct, and document the artifacts of a software system. It provides a set of graphical notation techniques to create abstract models of specific aspects of a system.[12]</p>

Table 3: Table of Programming and Modeling Languages

Conclusion:

In this chapter, we placed the project in its general context. We presented its objectives, as well as the adopted methodology, which is Scrum. Finally, we described the work environment.

Chapter II: Specification of Requirements and General Architecture

1.Introduction

The first phase of the videogame development cycle is the analysis and requirements specification phase. We developed both functional and non-functional needs. Additionally, we covered different use cases for our project.

2. Game summary

The game is a mobile game on Android belonging to the categories of open world, first-person, adventure, and survival, with low-poly graphics. The game centers on the oppressive regime of mafia. Player takes on the role of a fighter to liberate their homeland from the tyranny. Player's goal is to explore the world, engage in combat, and utilize an array of weapons and vehicles.

3. Actors' specification

The game includes three actors: "Player Character," "User," and "AI Character."

Player Character: Is the main actor. Their primary goal is to move around their environment and do objectives. They have the freedom to perform actions such as running, jumping, or getting into a vehicle.

AI Character: Is a bot who can be a police officer, a businessman, a woman, etc. Their role is to move around limited areas in the scene with the ability to react to our main actor by attacking them or do random activities such as walking, swimming, driving, etc.

User: Is the user of the game, the one who launches the application and navigates through the main menu and accesses other interfaces.

4.Backlog Product

ID	User Story	Priority
1	As a Player Character, I can move	++
2	As a Player Character, I can run	++
3	As a Player Character, I can jump	++
4	As a Player Character, I can control the camera	++
5	As a Player Character, I can Collect Bullets	+
6	As a Player Character, I can Collect money	+
7	As a Player Character, I can buy Items	++
8	As a Player Character, I can upgrade my stats	++
9	As a Player Character, I can attack the AI Character	+++
10	As a Player Character, I can aim with a gun	+++
11	As a Player Character, I can shoot with a gun	+++
12	As a Player Character, I can get into a vehicle	++
13	As a Player Character, I can drive a vehicle	++
14	As a Player Character, I can get out of the vehicle	++
15	As a Player Character, I can stay in the “Idle” state	++
16	As a Player Character, I can capture an outpost	+++
17	As a Player Character, I can capture a radio tower	+++
18	As a Player Character, I can capture a checkpoint	+++
19	As a Player Character, I can race against time	++
20	As a Player Character, I can race against AI Character	++
21	As an AI Character, I can find the path to the player	++
22	As an AI Character, I can attack the Player Character	++
23	As an AI Character, I can aim with a gun	++
24	As an AI Character, I can move	+
25	As an AI Character, I can run	+
26	As an AI Character, I can stay in the “Idle” state	+
27	As a User, I can start the game	++
28	As a User, I can save the game	++
29	As a User, I can load a save game	++
30	As a User, I can quit the game	+

Table 4: Table of Backlog Product

5. Functional and non-functional needs

5.1 Functional needs

The functional needs that the Player Character can perform:

❖ **Move:**

Movement is done using buttons and specific touch areas that allow walking, running, and jumping in all directions.

❖ **Control the camera:**

The player can control the camera using a button, allowing it to rotate in all directions.

❖ **Change weapons:**

The player can change his weapon through toolbar.

❖ **Drive a vehicle:**

The player can get into, drive, and leave the vehicle.

❖ **Stay in the “Idle” state:**

“Idle” is a transitional state close to immobility it activates when the player is not moving or perform any action.

The functional needs that the AI Character can perform:

❖ **Find the path to the player:**

The AI Character can find the shortest path to the player using the NavMesh and NavMesh Agent components of artificial intelligence.

❖ **Attack:**

The AI character can attack the player if he gets in range. Each attack reduces the player’s health points.

❖ **Move:**

The movement of the AI character is done using an artificial intelligence agent, allowing it to move naturally within a NavMesh. It can walk and run.

.

❖ **Saty in the “Idle” state:**

“Idle” is a transitional state close to immobility it activates when the AI character is not moving or perform any action.

The functional needs that the User can perform:

❖ **Start the game.**

❖ **Save/Load the game.**

❖ **Pause the game.**

❖ **Quit the game.**

5.2 Non-functional needs

After defining the functional needs of the game, we must necessarily consider the non-functional needs.

❖ **Performance:**

The game should run smoothly and responsively, providing a smooth gaming experience without lags or delays, even on lower-end devices.

❖ **Security:**

The game should implement appropriate security measures to protect player data and ensure the integrity of the gaming experience, including protection against cheating or unauthorized access.

❖ **Graphics and Visuals:**

The game should have high-quality graphics and immersive visuals to enhance the player experience.

❖ **Offline Capability:**

The game should be fully played offline, allowing players to enjoy the game without requiring an internet connection.

❖ **Storage and Resource Management:**

The game should have efficient resource management to minimize storage space requirements on the player's device and optimize memory usage for smooth performance.

6. Modeling of functional needs

6.1 General architecture of Unity

the general architecture and relationships between the basic classes of Unity are depicted in figure 5 below:

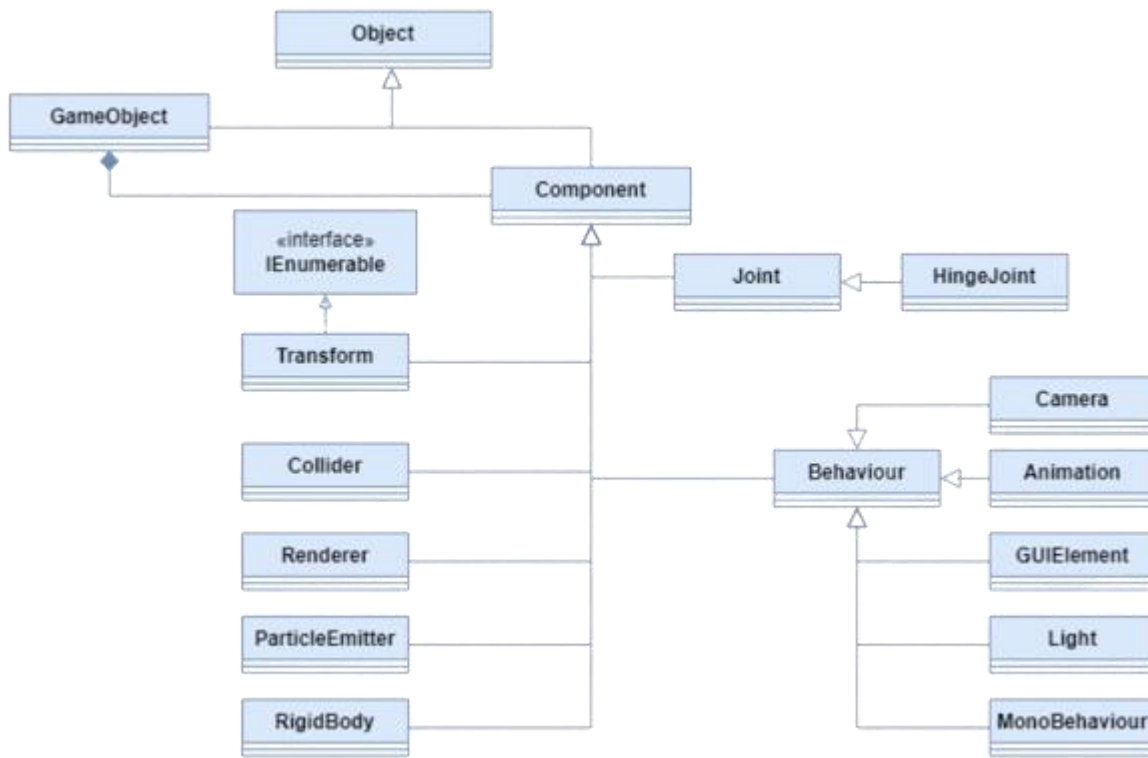


Figure 6: Unity architecture's diagram

- **Object:** Abstract class from which **GameObject** and **Component** inherit
- **GameObject:** Base class for all entities in Unity scenes.
- **Component:** Base class for all components that can be attached to **GameObjects**.
- **Behaviour:** A type of **Component** that can be enabled or disabled.
- **Joint:** Base class for all Joints
- **Transform:** Allows fixing the position, rotation, and scale of an object.
- **Rigidbody:** Controls the position of an object through physics simulation.
- **Renderer:** Makes an object appear on the screen.

- **Collider:** Base class for all colliders. Examples of classes inheriting from Collider include SphereCollider, BoxCollider, and MeshCollider.
- **ParticleEmitter:** Script interface for particle emitters.
- **Camera:** A camera is a device through which the player views the world.
- **Light:** Script interface for light components.
- **Animation:** The animation component is used to play animations.
- **GUIElement:** Base class for images and text strings in graphical interfaces. Examples of classes inheriting from GUIElement include GUIText and GUITexture.
- **HingeJoint:** Connects two rigid bodies, forcing them to move as if connected by a hinge.

6.2 General use case diagram:

The functionalities available to the user before starting the game are depicted in Figure 6.1 below:

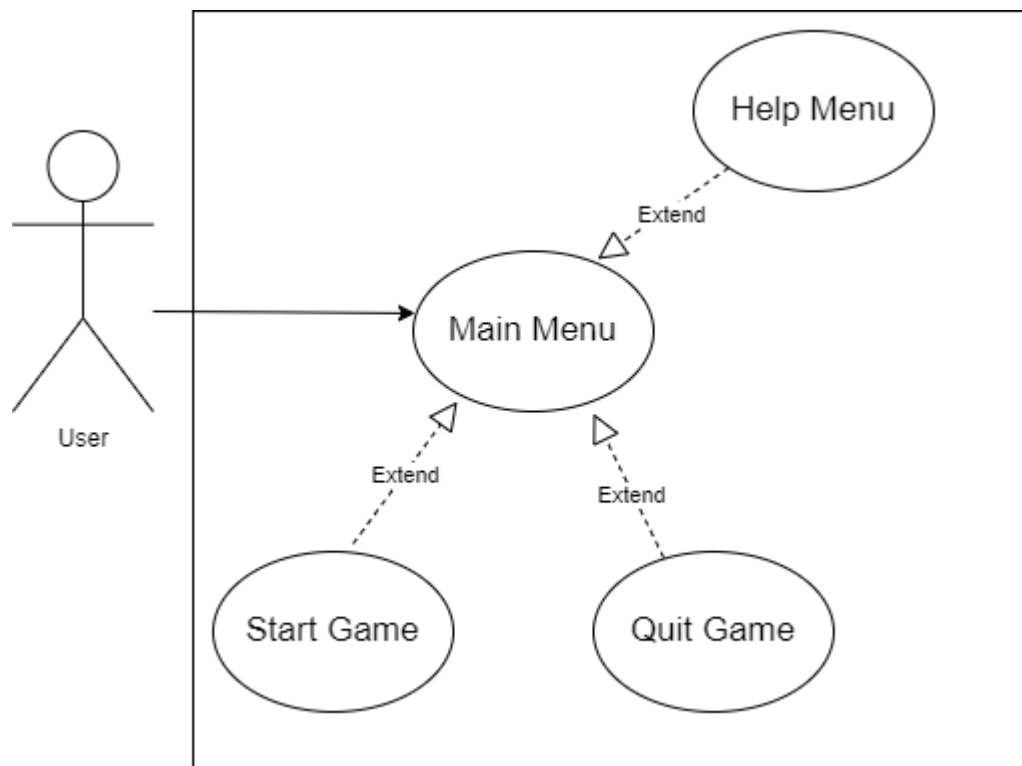


Figure 7.1: General Use Case Diagram

Once the game start, the user gains access to the game features depicted in figure 6.2 below:

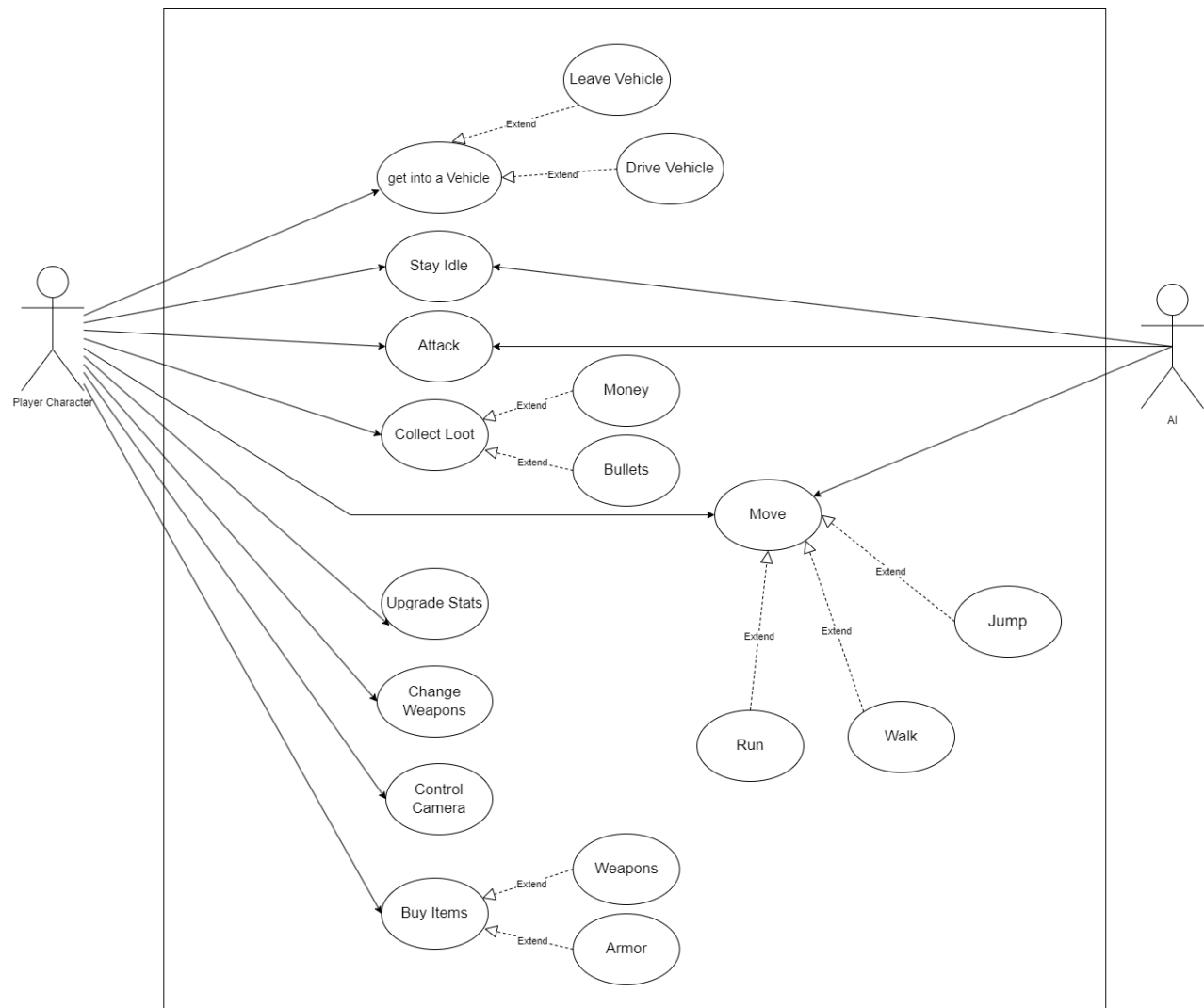


Figure 7.2: General Use Case Diagram

6.3 General class Diagram

this class diagram of the game contains the most important classes presented in the project:

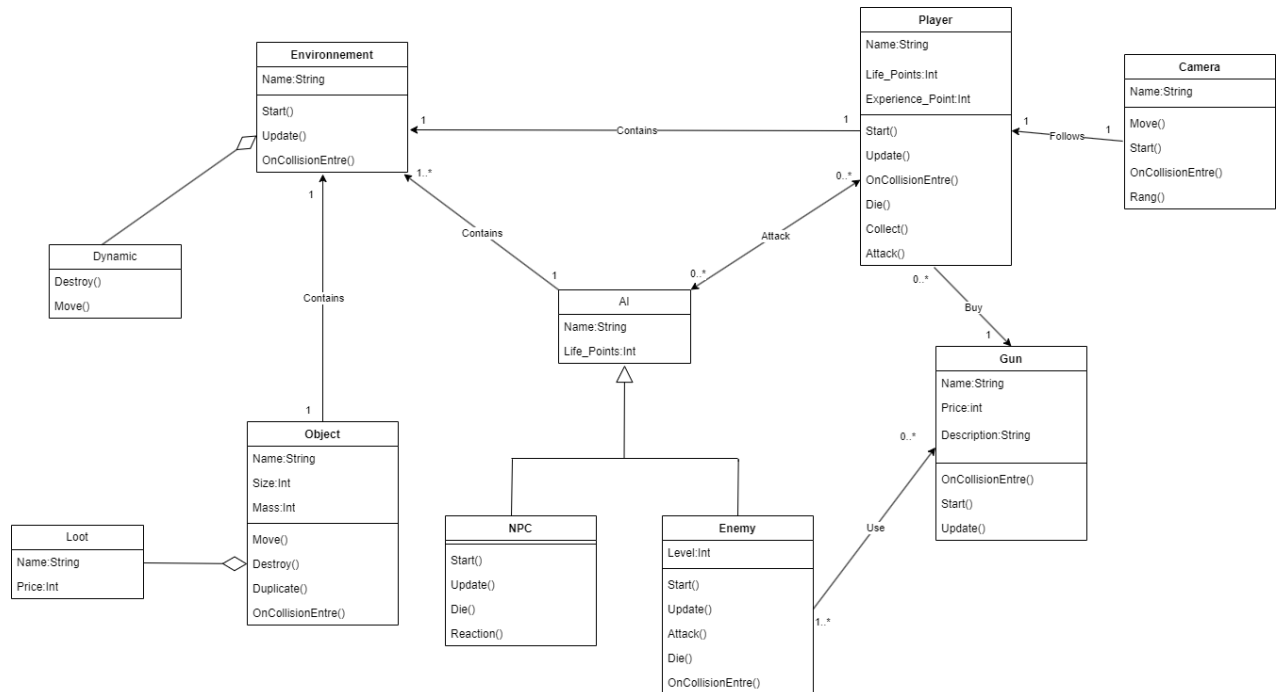


Figure 8: General class diagram

When we look at the diagram, we can immediately notice that the **Player** class is very important because it is obviously the **GameObject** of the player character. This **GameObject** has components such as the **Player** script, which controls variables such as health and experience points. There is also the **MainCamera** **GameObject**, which follows the player wherever he goes.

Conclusion:

In this chapter, we analyzed and specified the functional and non-functional needs of our project and the product backlog. We then presented the architecture with UML diagrams.

Chapter III Sprint 1: Game mechanics and the player character

1.Introduction

This chapter will present a study of the functional needs of the player character related to gameplay itself, the design, and we will conclude the chapter with the implementation.

2.Backlog

ID	User Story	Priority
1	As a Player Character, I can move	++
2	As a Player Character, I can run	++
3	As a Player Character, I can jump	++
4	As a Player Character, I can control the camera	++
5	As a Player Character, I can Collect Bullets	+
6	As a Player Character, I can Collect money	+
7	As a Player Character, I can buy Items	++
8	As a Player Character, I can upgrade my stats	++
9	As a Player Character, I can attack the AI Character	+++
10	As a Player Character, I can aim a gun	+++
11	As a Player Character, I can shoot a gun	+++
12	As a Player Character, I can get into a vehicle	++
13	As a Player Character, I can drive a vehicle	++
14	As a Player Character, I can get out of the vehicle	++
15	As a Player Character, I can stay in the “Idle” state	++
16	As a Player Character, I can capture an outpost	+++
17	As a Player Character, I can capture a radio tower	+++
18	As a Player Character, I can capture a checkpoint	+++
19	As a Player Character, I can race against time	++
20	As a Player Character, I can race against AI Character	++

Table 5: Table of sprint 1 backlog product.

3. Use case diagram

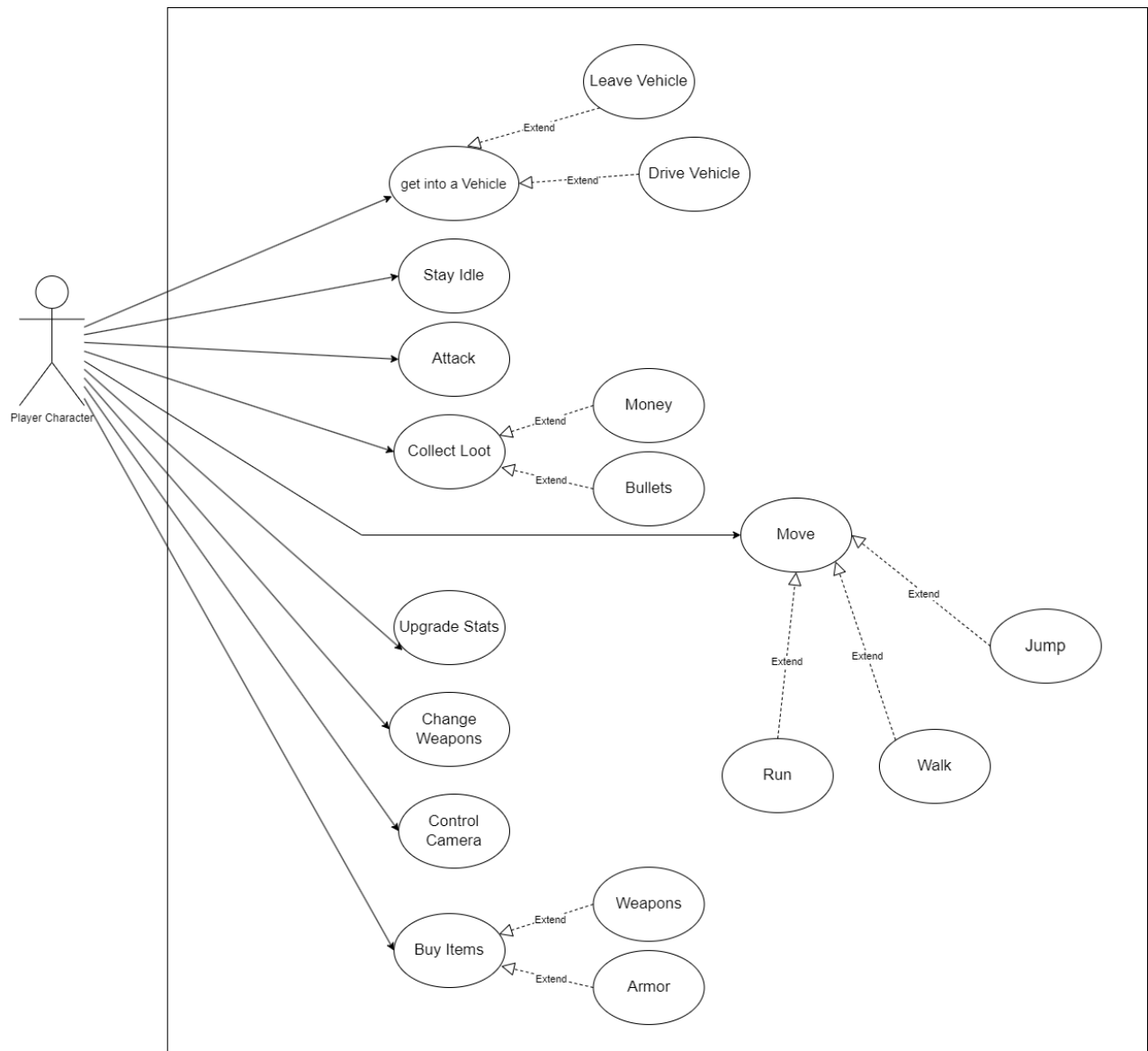


Figure 9: Use case diagram of player character.

3.1 Use case of “Move”:

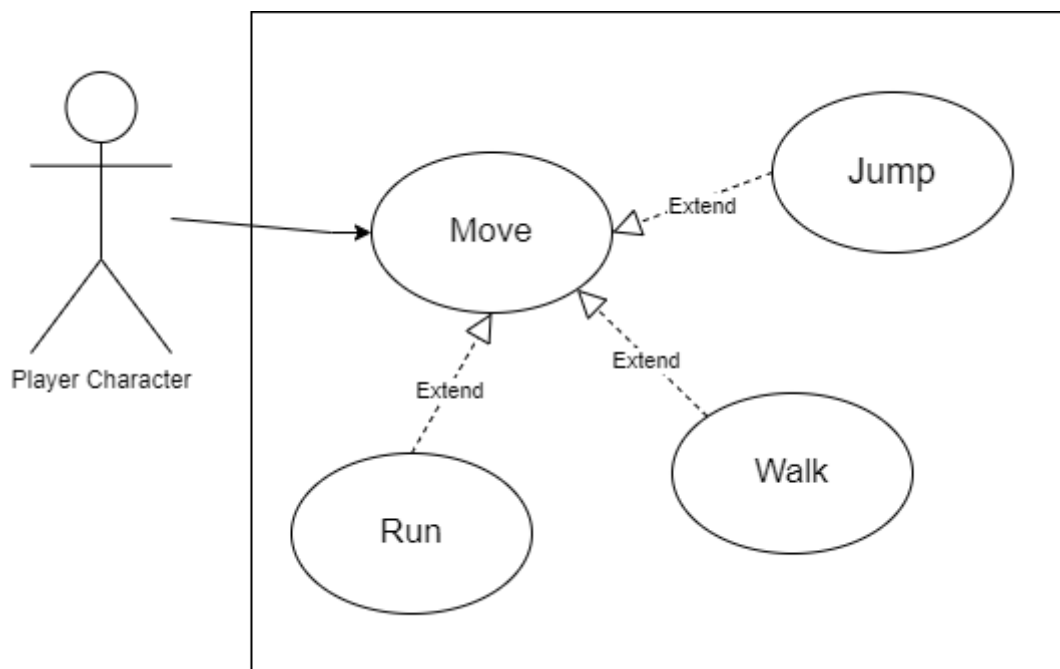


Figure 10: Use case diagram of “Move”.

Use Case	Move
Actor	Player Character
Precondition	The game is started
Postcondition	The Player Character is moving
Description	By pressing specific buttons, the Player Character can move within the game scene: he can walk, run, and jump in all directions.

Table 6: Table of use case “move.”

3.2 Use case of “Control Camera”:

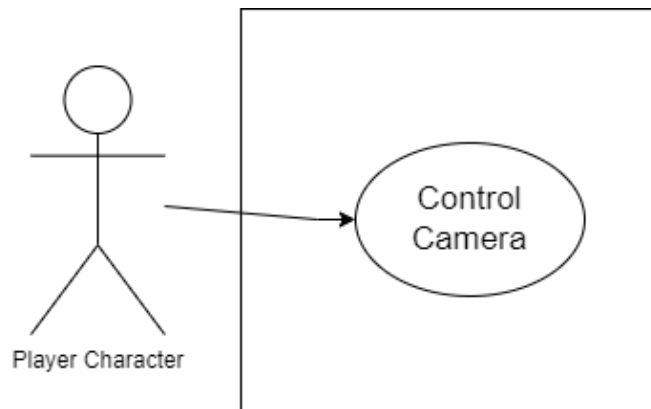


Figure 11: Use case diagram of “Control camera”.

Use Case	Control Camera
Actor	Player Character
Precondition	The game is started
Postcondition	Changing the camera’s position
Description	By pressing specific button, the Player Character can change the camera’s position in the direction that he desires.

Table 7: Table of use case “Control camera.”

3.3 Use case of “Collect Loot”:

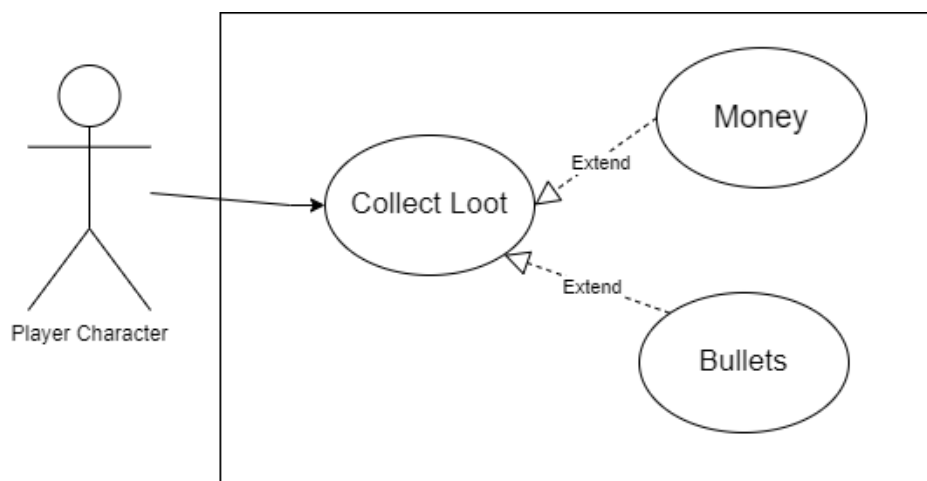


Figure 12: Use case diagram of “Collect loot.”

Use Case	Collect Loot
Actor	Player Character
Precondition	The game is started
Postcondition	Objects are collected
Description	The Player Character can collect money and bullets by approaching them or by pressing specific button.

Table 8: Table of use case “Collect Loot.”

3.4 Use case of “Change Weapons”:

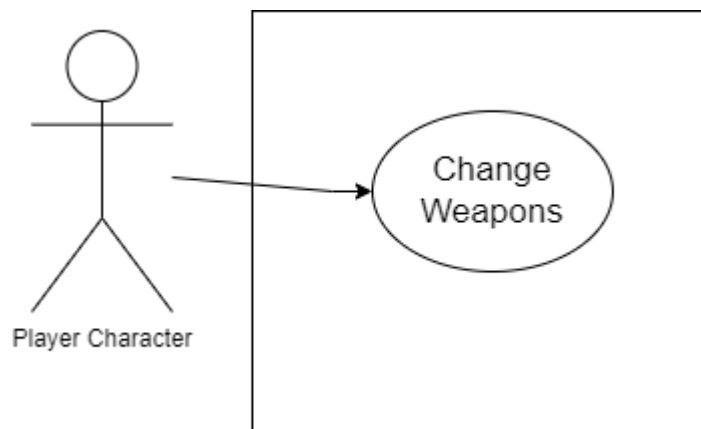


Figure 13: Use case diagram of “Change Weapons.”

Use Case	Change Weapons
Actor	Player Character
Precondition	The game is started
Postcondition	Weapon is changed
Description	The Player Character can change between four weapons by pressing specific button.

Table 9: Table of use case “Change Weapons.”

3.5 Use case of “Attack”

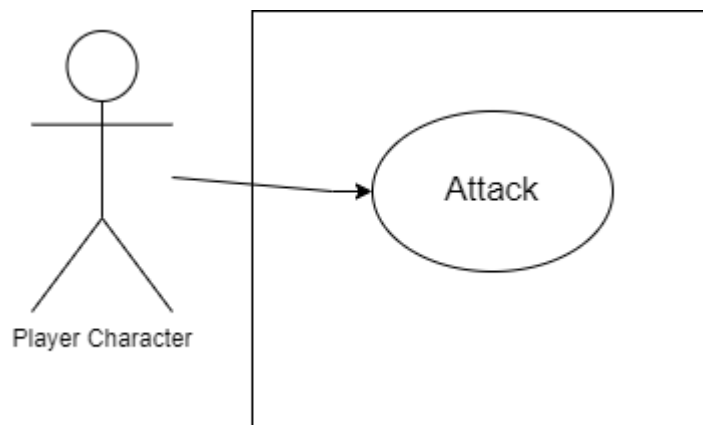


Figure 14: Use case diagram of “Attack.”

Use Case	Attack
Actor	Player Character
Precondition	The game is started
Postcondition	The attack is finished
Description	The Player Character can press button to shoot bullets through the gun.

Table 10: Table of use case “Attack.”

3.6 Use case of “Get into a vehicle”:

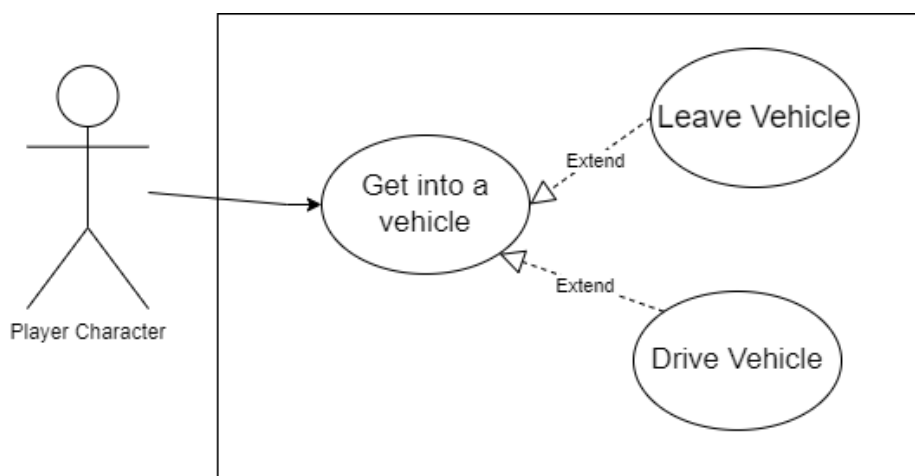


Figure 15: Use case diagram of “Get into a vehicle”.

Use Case	Get into a vehicle
Actor	Player Character
Precondition	The Player Character stands near a vehicle.
Postcondition	The Player drive or out of the vehicle
Description	The Player Character can drive or get out the vehicle.

Table 11: Table of use case “Get into a vehicle.”

3.7 Use case of “Buy Items”:

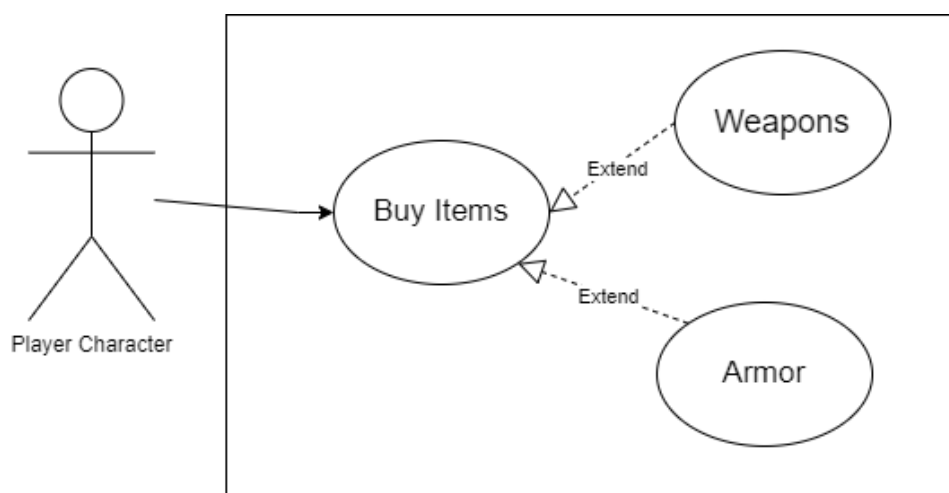


Figure 16: Use case diagram of “Buy Items”.

Use Case	Buy Items
Actor	Player Character
Precondition	The Player Character stands in front of a gun shop
Postcondition	Items are purchased or not
Description	The Player Character can buy items by pressing specific buttons.

Table 12: Table of use case “Buy Items.”

3.8 Use case of “Upgrade Stats”:

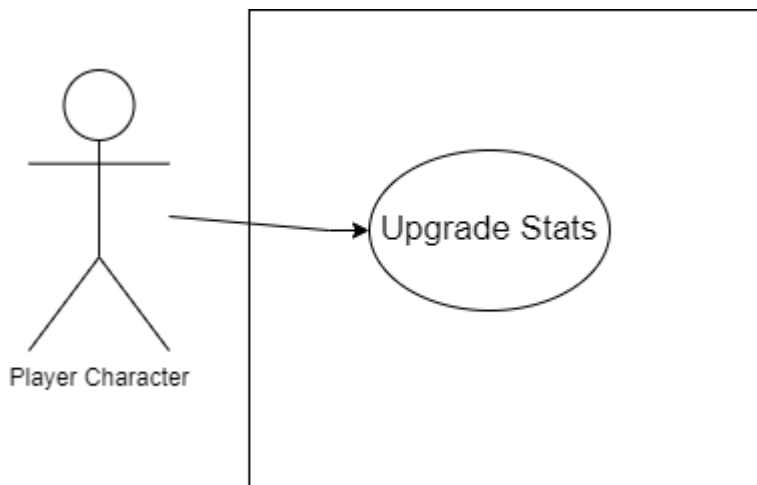


Figure 17: Use case diagram of “Upgrade stats”.

Use Case	Upgrade Stats
Actor	Player Character
Precondition	The Player Character opens the upgrade’s menu
Postcondition	Stats are upgraded or not
Description	The Player Character can upgrade his stats such as health, armor, damage, speed, etc.

Table 13: Table of use case “Upgrade stats.”

4.Game mechanics

4.1 Activity diagram:

In this section, we will present the activity diagram related to the player, encompassing the needs we have outlined in this sprint.

In UML language, an activity diagram provides a view of a system's behavior by describing the sequence of actions in a process. Activity diagrams are like information processing flowcharts because they show the flows between actions in an activity. However, activity diagrams can also show simultaneous parallel flows and alternative flows.

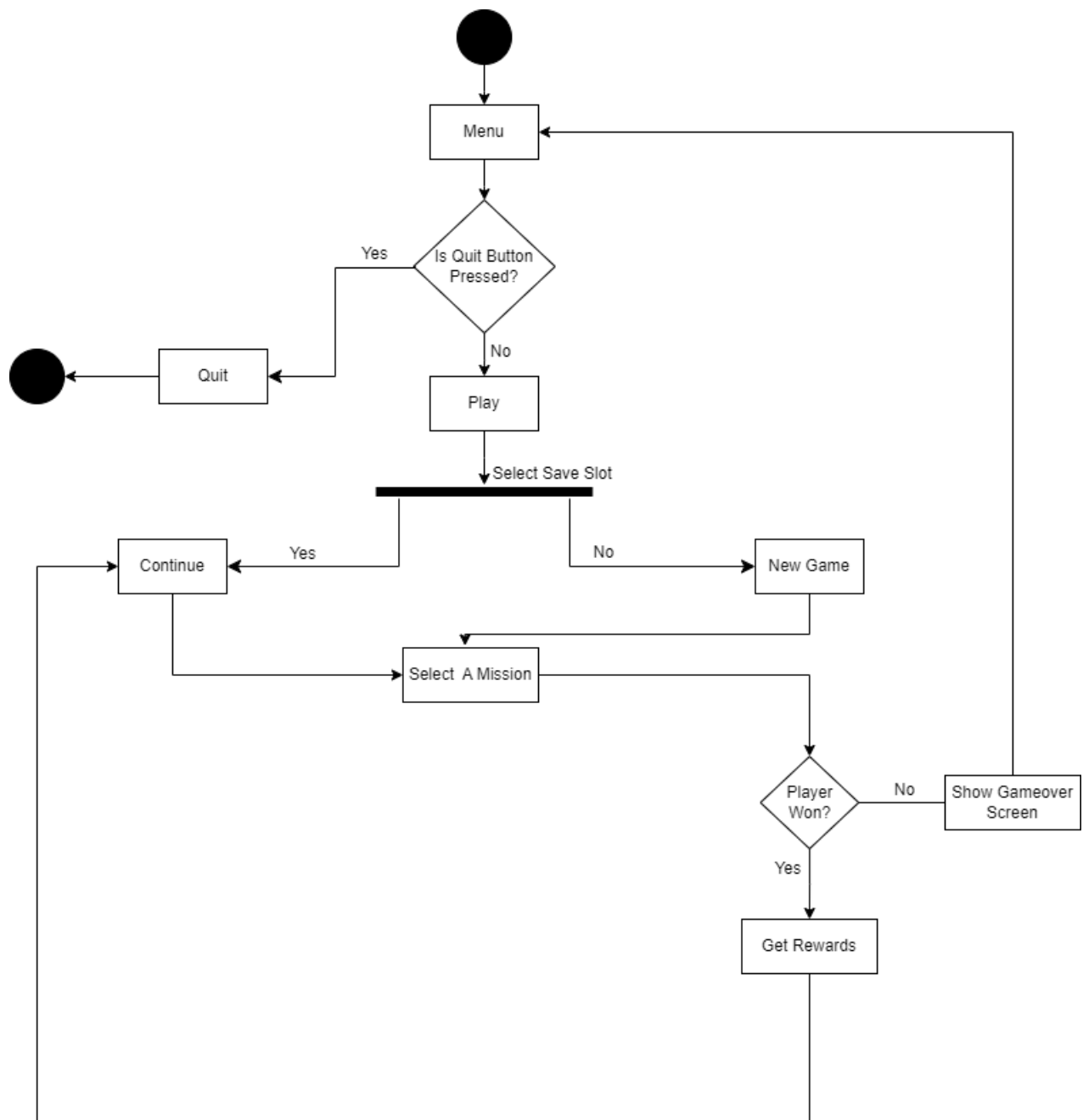


Figure 18: Activity diagram.

4.2 Missions

In a first-person game (FPS), missions play a crucial role in providing structure to the gameplay. In this game, there are two types of missions: main missions and side missions. Main missions focus on introducing and expanding upon the game's mechanics. Moreover, main missions are typically more challenging, providing a sense of achievement and progression. Through these missions, players are encouraged to explore the game's environment, experiment with different strategies, and fully engage with the mechanics at their disposal.

In this game there are 3 types of main mission:

4.2.1 Main Missions

4.2.1.1 Checkpoint

In this mission, the player's objective is to secure a checkpoint. To achieve this, the player must destroy the gate using bullets or a car and eliminate the police man. Overcoming these challenges requires tactical thinking, precise execution, and effective use of available resources. Capturing the checkpoint not only gives the player experience points and money but also makes his movement in the game's world easier.

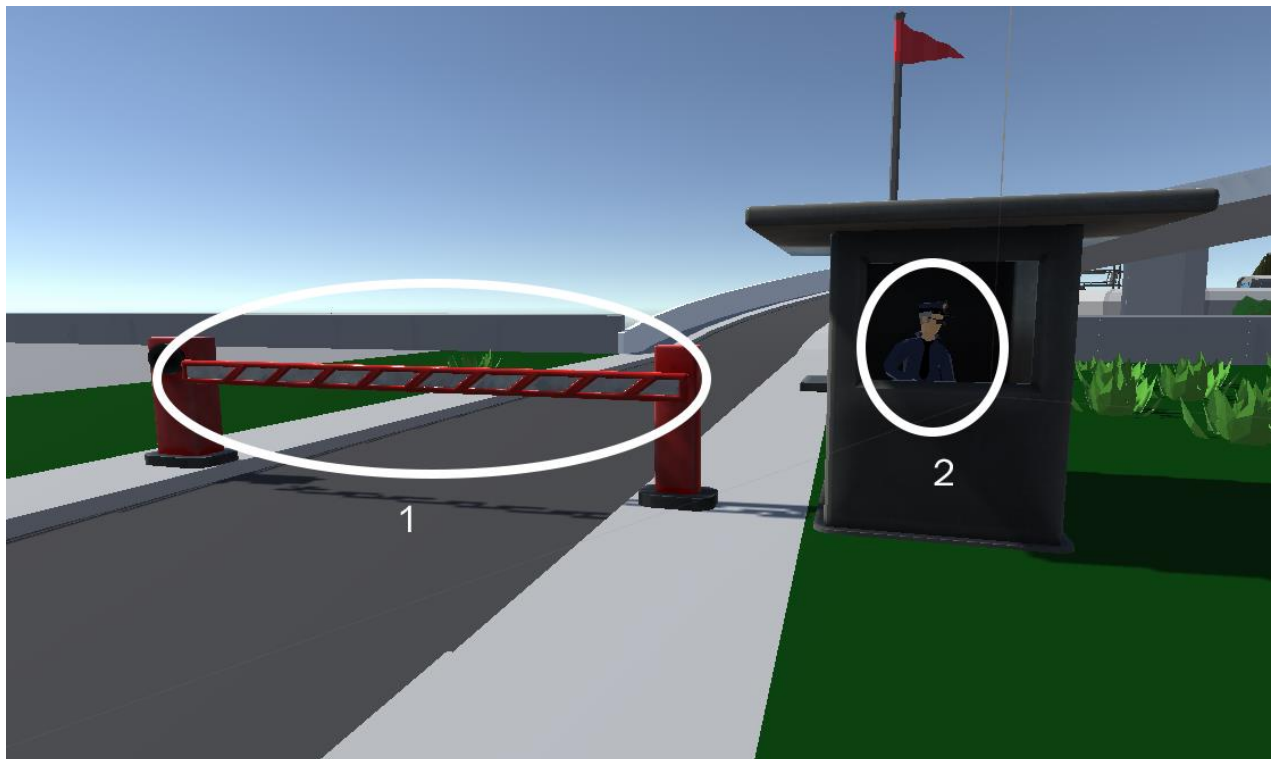


Figure 19: Checkpoint

ID	Description
1	Checkpoint's gate
2	Police man

Table 14: Table of checkpoint mission

4.2.1.2 Radio Tower

The radio tower missions are a pivotal aspect of the game's exploration and progression. In these missions, the player must reach the top tower. The primary objective is to destroy the satellite dish located at the top. This mission gives the player the capability to discover the game's environment from the top.

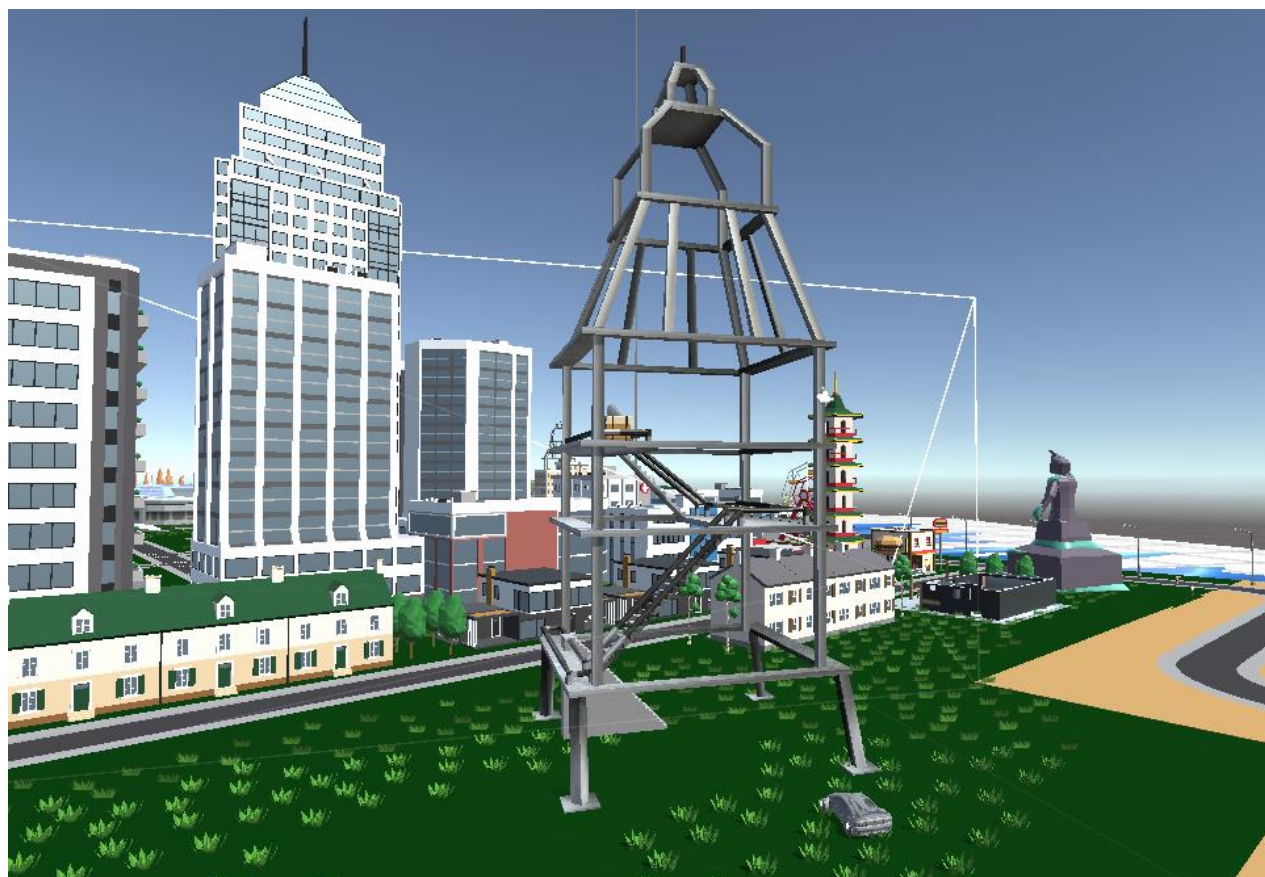


Figure 20: Radio tower..

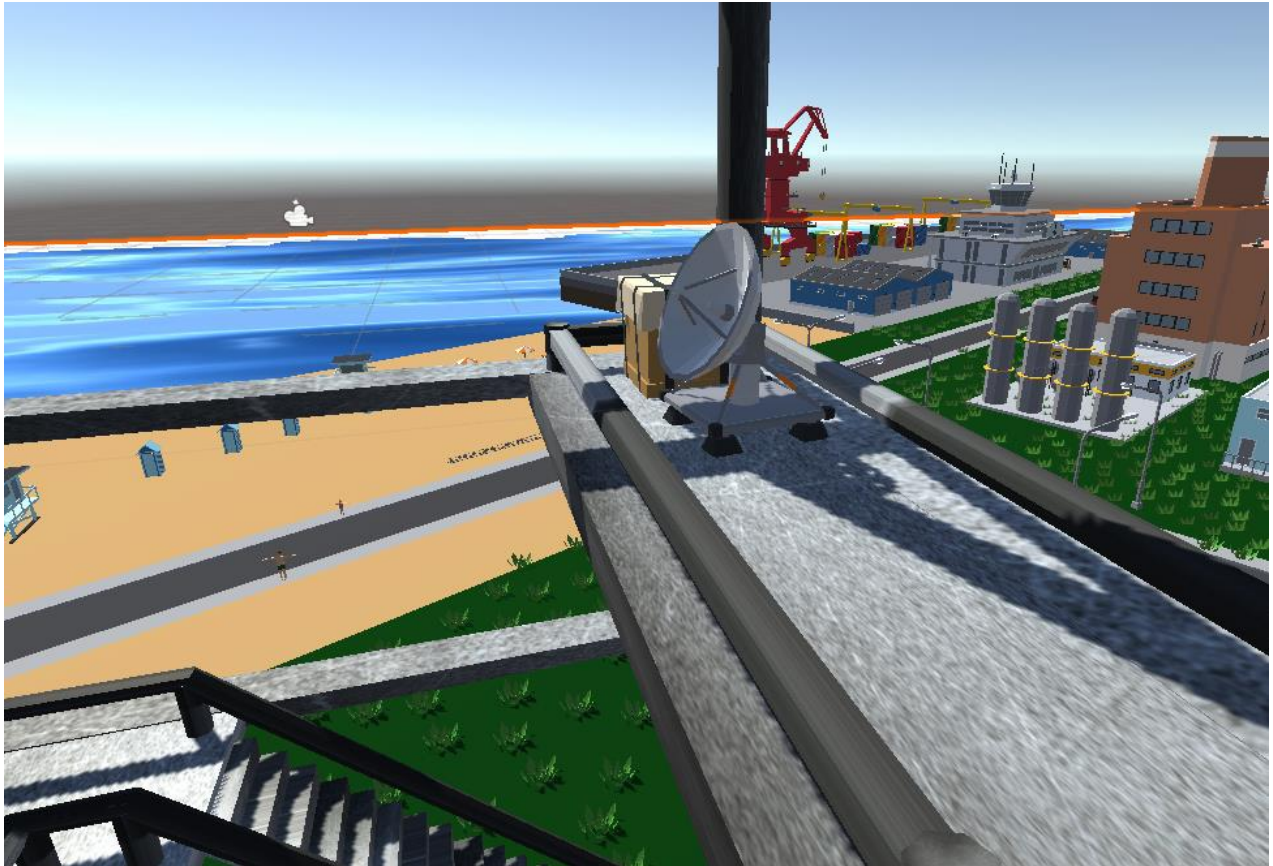


Figure 21: Satellite dish.

4.2.1.3 Outpost

Outpost missions are a critical element of the game's open-world experience, offering players the opportunity to liberate enemy-held territories. In this missions the player's objective is to assault a fortified outpost, eliminate the army, and destroy a satellite. To accomplish this, players must employ combat skills.

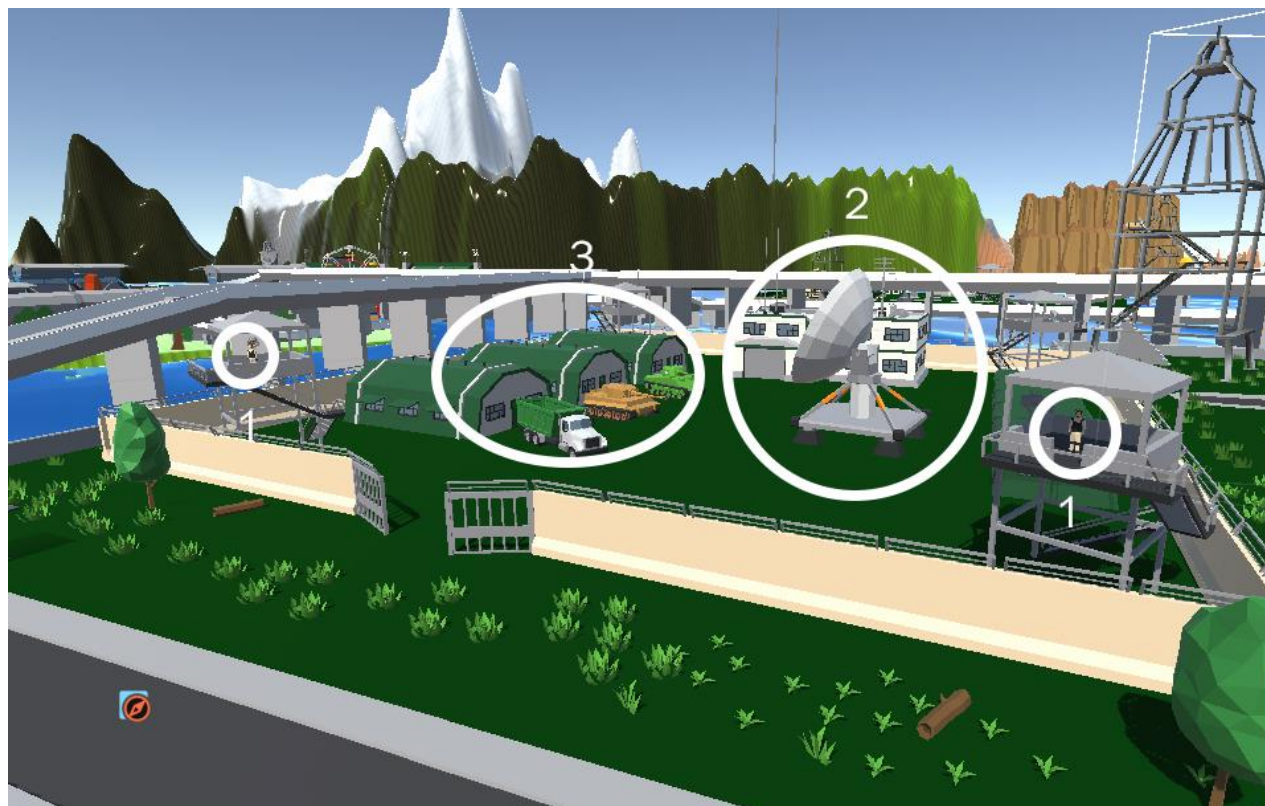


Figure 22: Outpost.

ID	Description
1	Snipers
2	Outpost's Satellite
3	Military barracks

Table 15: Table of outpost mission

4.2.2 Side missions

On the other hand, side missions offer optional tasks that players can undertake. These missions provide additional challenges, rewards, and opportunities to explore the game world more thoroughly.

4.2.2.1 Races

Race missions challenge the player to test their driving skills. This mission can be against the clock or head-to-head with AI characters, each offering a unique thrill. Completing these races not only provides a rush of adrenaline but also rewards players.

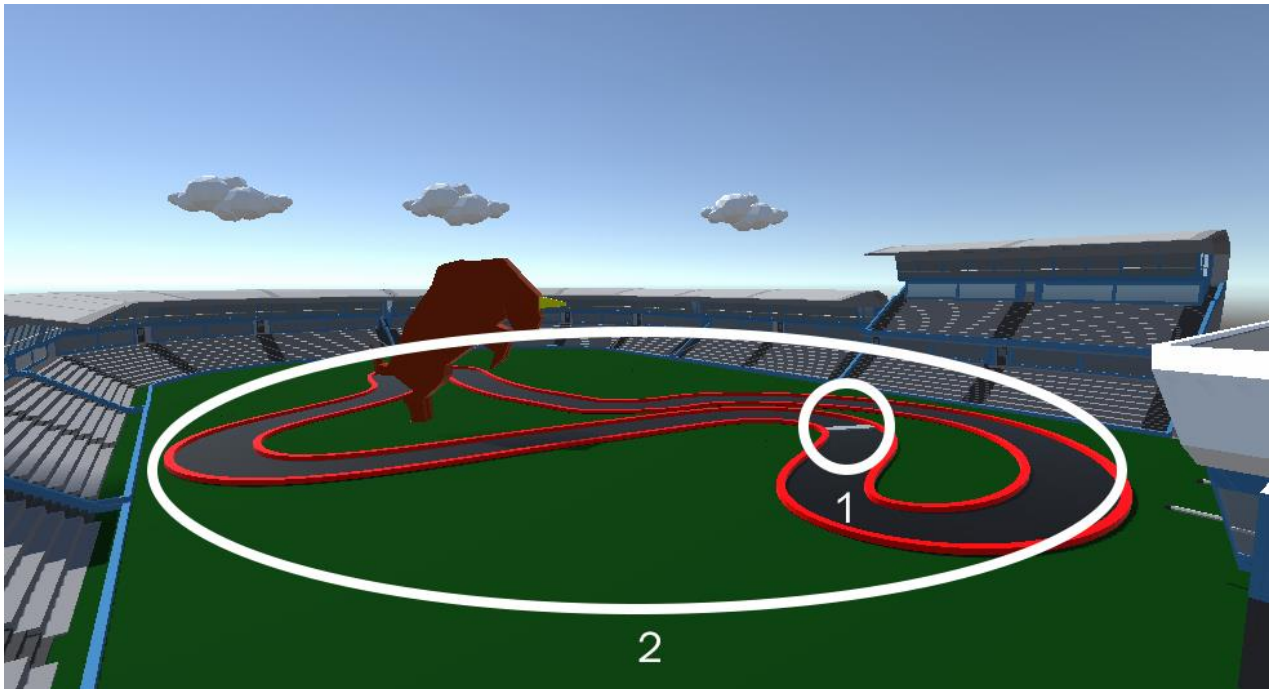


Figure 23: Racetrack 1.

ID	Description
1	Start/Finish Line
2	Racetrack

Table 16: Table of race 1 mission

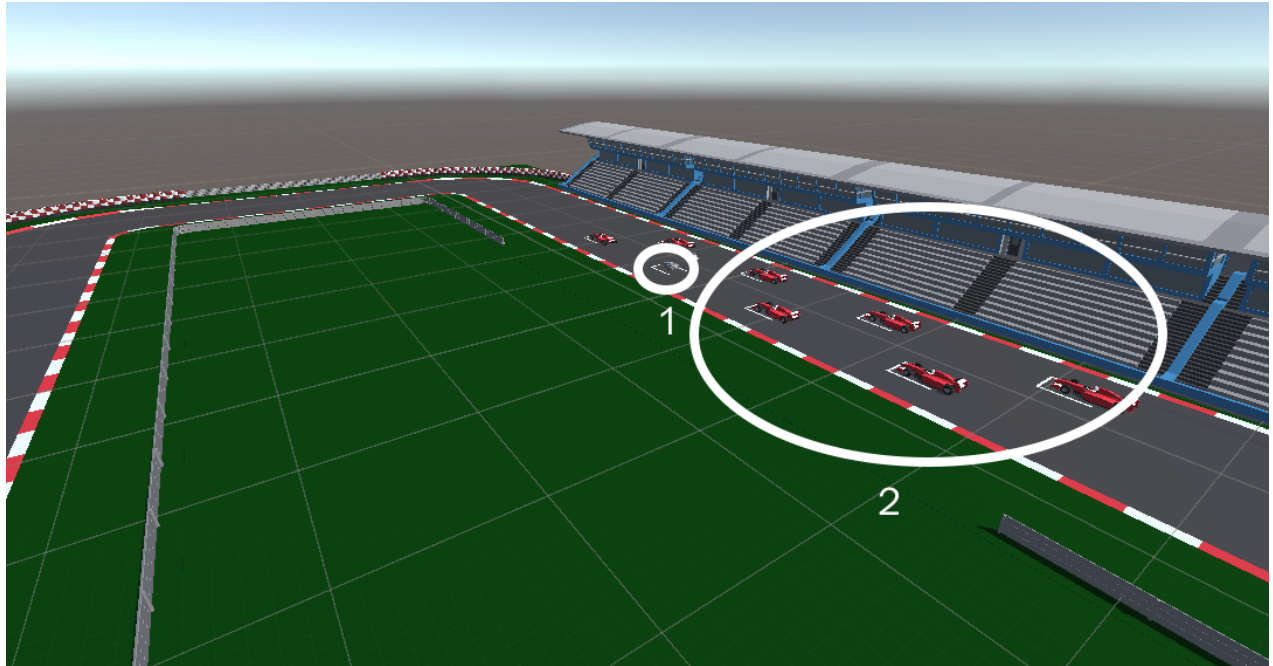


Figure 24: Racetrack 2.

ID	Description
1	Player's car
2	AI Characters

Table 17: Table of race 2 mission

4.2.2.2 Collectibles

Collectibles add an extra layer of exploration and reward to the game. Players can hunt for and gather loot from various chests can be found across the game's map. These chests contain valuable items such as weapons, ammunition, experience points and money. Each chest is strategically placed, encouraging players to thoroughly explore the game's environments.



Figure 25: Chest.

4.3 Progression system

The progression system in videogames is important because it keeps players motivated and excited. As you play, you can unlock new skills, levels, or rewards, making you feel like you're constantly improving. This system gives the game a clear path of growth, making it more fun and engaging. Without it, players might get bored quickly since there wouldn't be much to achieve or look forward to. So, a good progression system makes the whole gaming experience more enjoyable and satisfying.

In this game, the progression system revolves around money and levels. Every activity you complete rewards you with money and experience points. The experience points help you level up, and each new level grants you skill points (SP). You can use these SP to upgrade your stats like health bar, damage, jumping, speed, etc., making the game easier and offering different ways to play. With the money you earn, you can buy new weapons and armor, which also enhance your experience and abilities. This system keeps the gameplay exciting and rewarding, as you're constantly improving and unlocking new content.

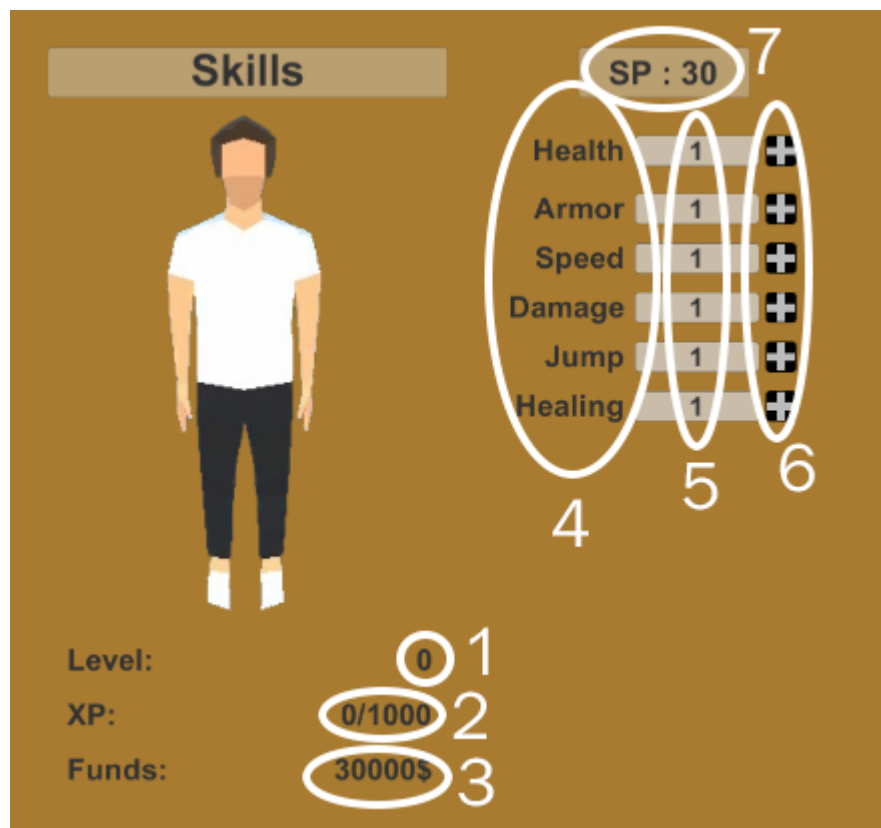


Figure 26: Upgrade menu.

ID	Description
1	Level
2	Experience points
3	Money
4	Stats' name
5	Stats' level
6	Upgrade buttons
7	Skill points (SP)

Table 18: Table of upgrade menu.

5. Sprint 1 realization

5.1 Unity Asset Store

The Unity Asset Store contains a library of free and commercial resources created by Unity Technologies and community members. A wide variety of resources are available, including textures, models, animations, entire project examples, tutorials, and editor extensions. In this project we imported models, textures, and materials from the Unity Asset Store, such as those for houses, buildings, cars, and accessories, to create a realistic scene without spending much time on modeling. [13]



Figure 27: Unity asset store.

5.2 Player Character

For the player character, we used the Low Poly Ultimate Pack for the player model and StarterAssets - FirstPerson for the player controller. The Low Poly Ultimate Pack, available on the Unity Asset Store, provides a collection of simple, stylized 3D models that are easy to integrate and perform well in-game. The StarterAssets - FirstPerson is a Unity Asset that includes a ready-made first-person controller, making it straightforward to implement smooth and responsive player movement.[14][15]

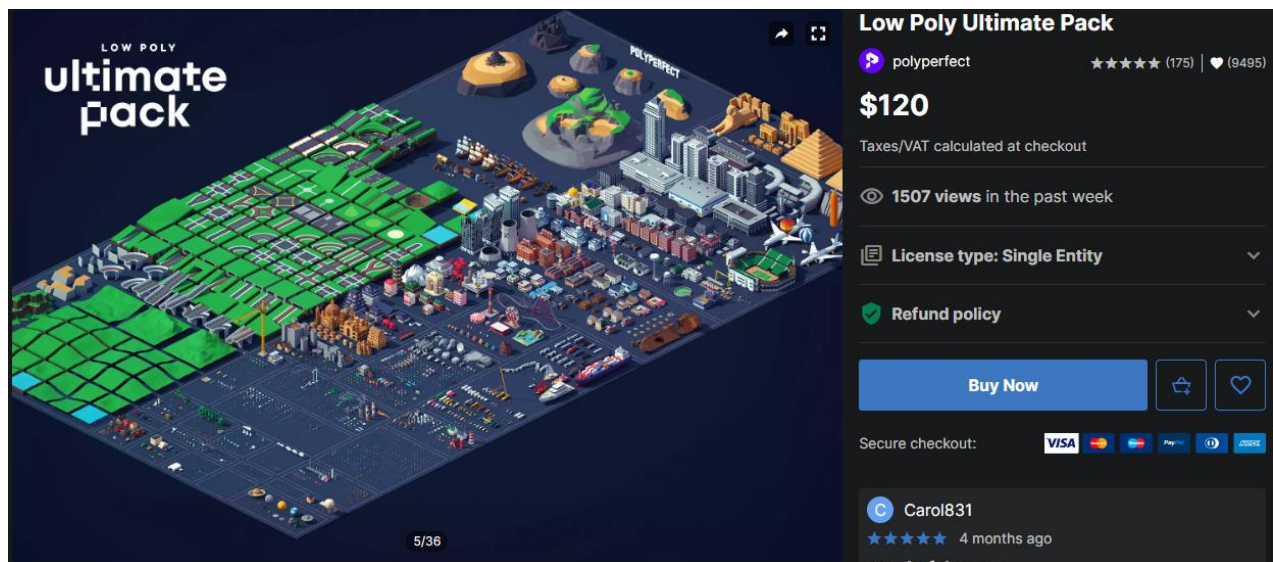


Figure 28: Low poly ultimate pack.

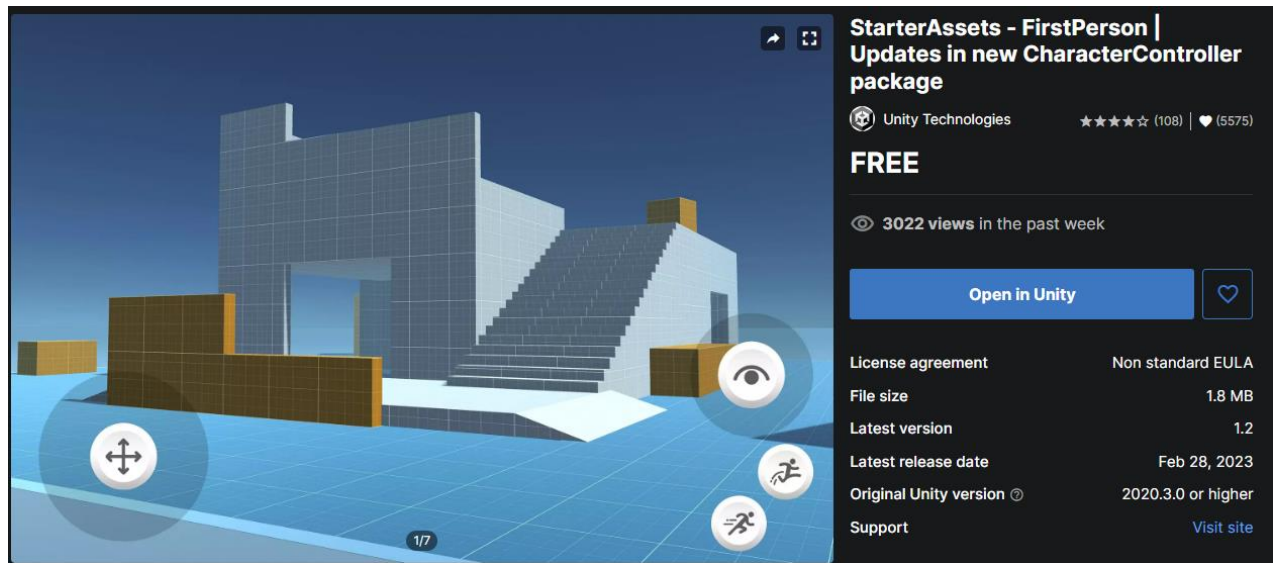


Figure 29: StarterAssets - FirstPerson.

5.3 Game's Scene

Terrain:

Unity provides a selection of tools for creating environmental features such as terrains and vegetation.

The game's terrain contains rural and urban areas. I used Far Cry's franchise maps as a reference and added some special touches.

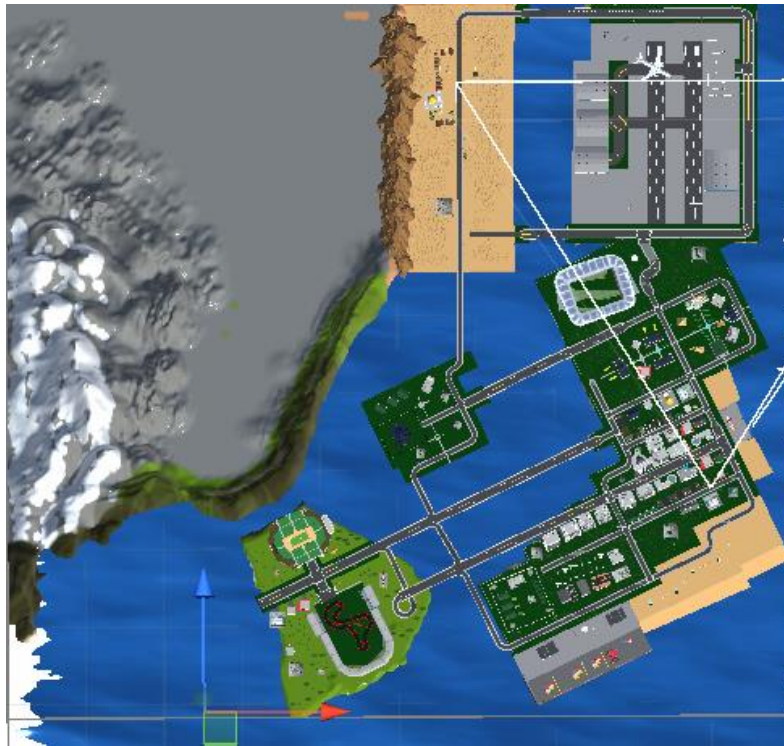


Figure 30: Game's map.

Rural areas:

Rural areas are located outside cities and towns. These areas often offer a peaceful environment.



Figure 31: example of rural areas.

Urban areas:

Urban areas include cities and towns, offering diverse opportunities for employment, education, and entertainment.



Figure 32: example of urban areas.

Vehicles:

For the vehicle controller, we used Unity Standard Assets. Unity Standard Assets are a collection of pre-made assets provided by Unity, including scripts, models, and controllers, designed to help developers quickly implement common game features. Using these assets made it easy to set up realistic and functional vehicle controls in the game.

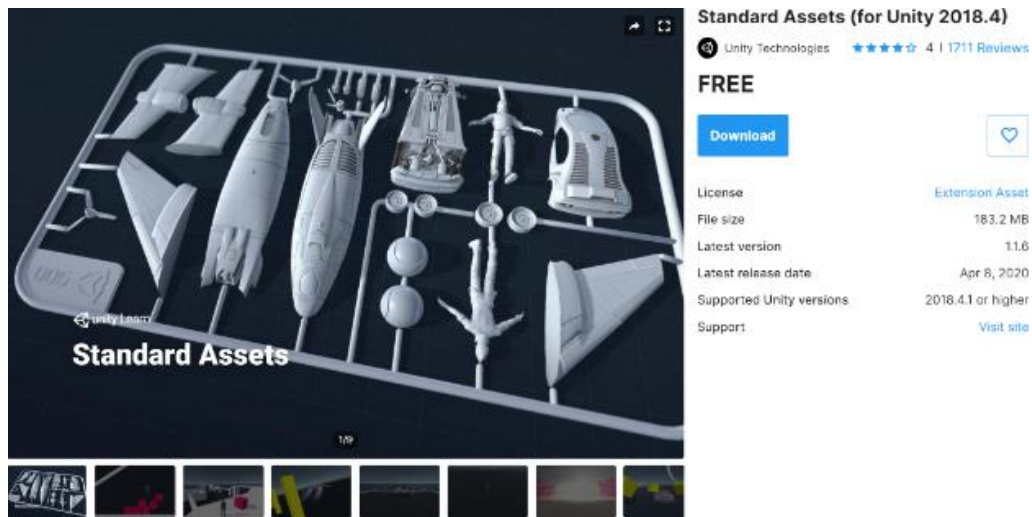


Figure 33: Standard Assets

5.4 Gameplay

Input System

The Input System package implements a system that allows any type of input device to control Unity content. It is designed to be a more powerful, flexible, and configurable replacement for Unity's classic Input Manager (the `UnityEngine.Input` class). The following figure provides a simplified summary of our Input System configuration without going into code details.



Figure 34: Input System.

Interface

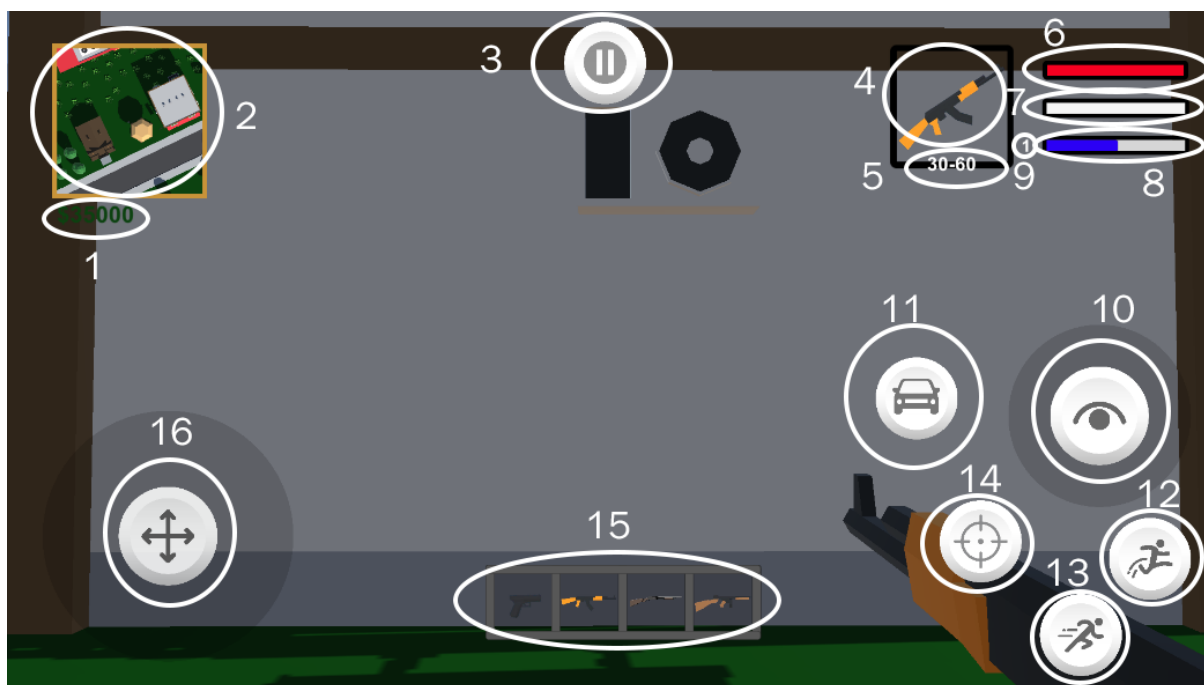


Figure 35: User Interface.

ID	Description
1	Money
2	Mini map
3	Pause menu button
4	Selected weapon
5	Weapon's ammo
6	Health's bar
7	Armor's bar
8	Level's bar
9	Level
10	Control camera analog
11	Get in/out of vehicle button
12	Jump button
13	Sprint button
14	Shoot button
15	Toolbar
16	Moving analog

Table 19: Table of user interface.

Conclusion

In this chapter, we focused on the first sprint of the project.

We began by identifying the backlog for sprint 1, then refined the use cases as we studied the sprint. Next, we modeled the first part of our system by creating sequence diagrams for each use case.

Finally, we presented the implementation: we introduced the player character for the first time then we showcased the terrain with its animated and realistic landscapes, as well as the gameplay interfaces. Additionally, we provided all types of missions, and its rewards and how the progress system is working.

Chapter IV Sprint 2: AI Character

1.Introduction

This chapter will cover the functional needs of the player character, focusing on gameplay and design. We will conclude with the implementation details.

2.Backlog

ID	User Story	Priority
21	As an AI Character, I can find the path to the player	++
22	As an AI Character, I can attack the Player Character	++
23	As an AI Character, I can aim with a gun	++
24	As an AI Character, I can move	+
25	As an AI Character, I can run	+
26	As an AI Character, I can stay in the “Idle” state	+

Table 20: Table of sprint 2 backlog.

3. Use case diagram

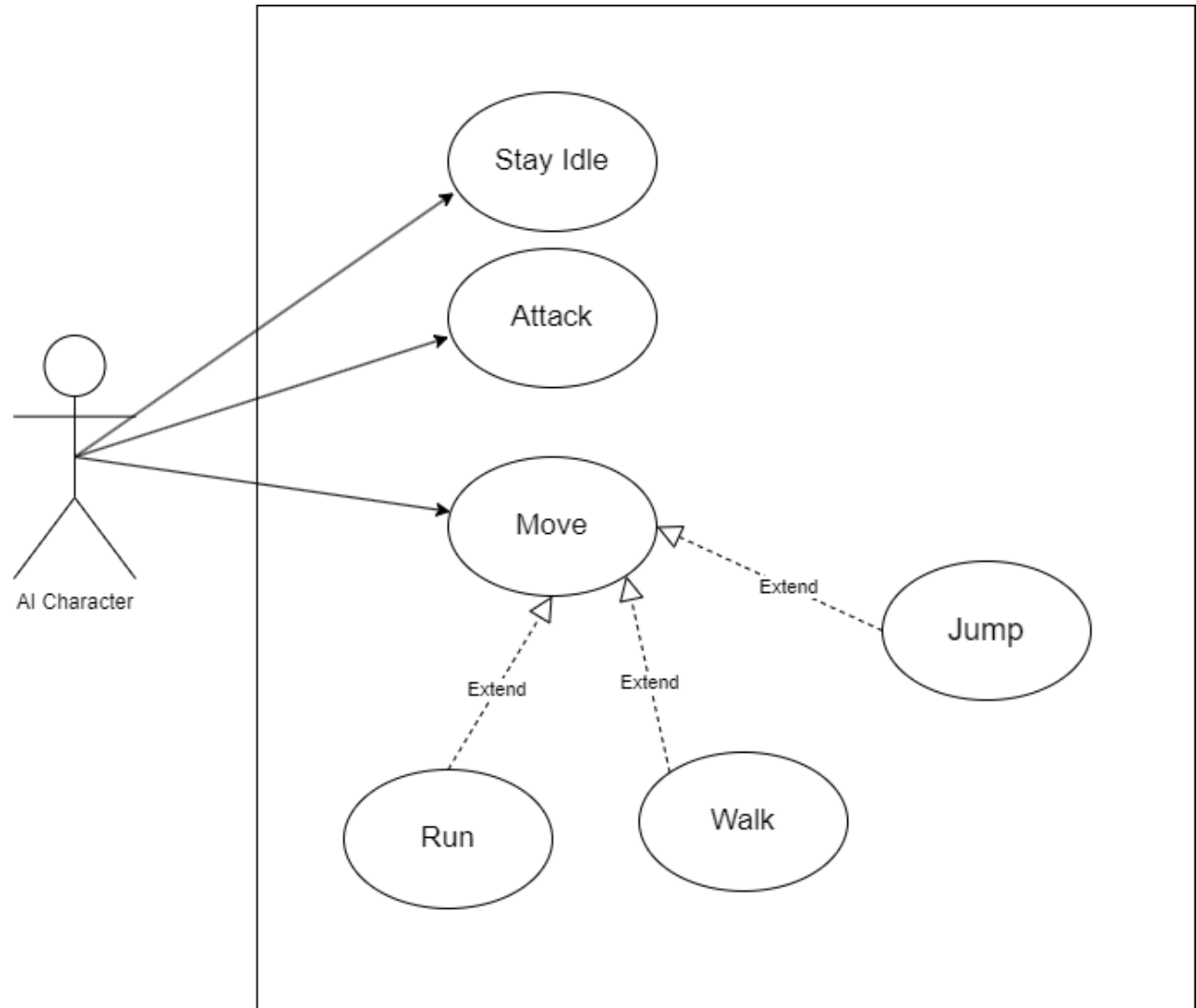


Figure 36: Use case diagram of AI character.

3.1 Use case of “Move”:

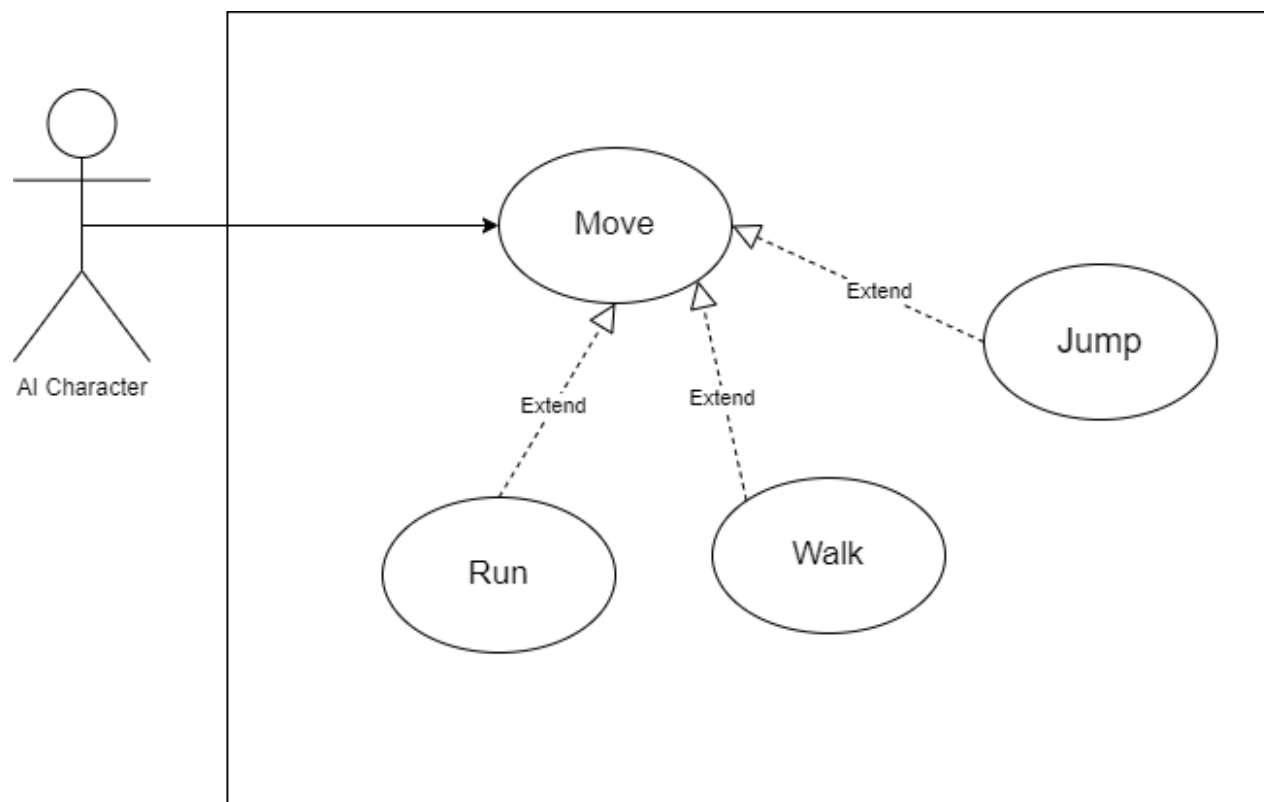


Figure 37: Use case diagram of “Move”.

Use Case	Move
Actor	AI Character
Precondition	The game is started
Postcondition	The AI Character is moving
Description	The AI Character moves within the game scene

Table 21: Table of Use case “Move”.

3.2 Use case of “Attack”:

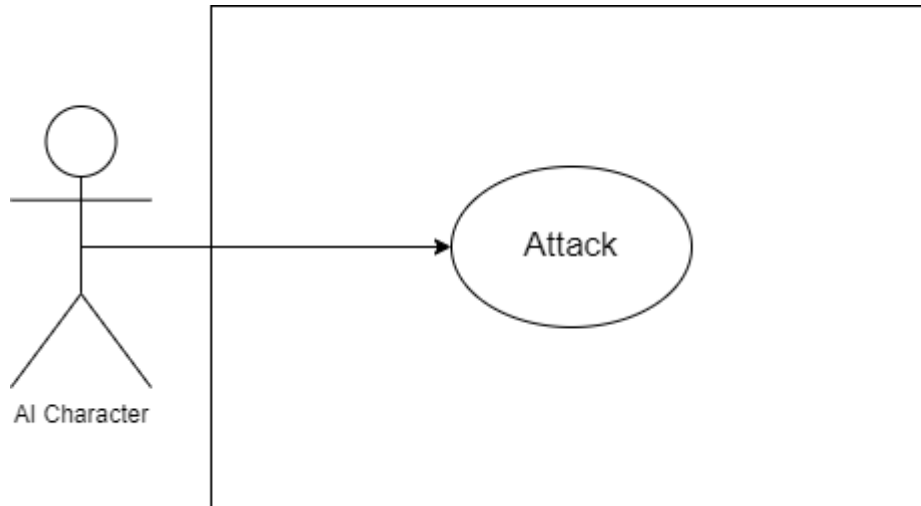


Figure 38: Use case diagram of “Attack”.

Use Case	Attack
Actor	AI Character
Precondition	The Player Character approaches the AI Character.
Postcondition	The Attack is finished
Description	When the Player Character approaches the AI Character, it will attack him.

Table 22: Table of Use case “Attack”.

3.3 Use case of “Stay Idle”:

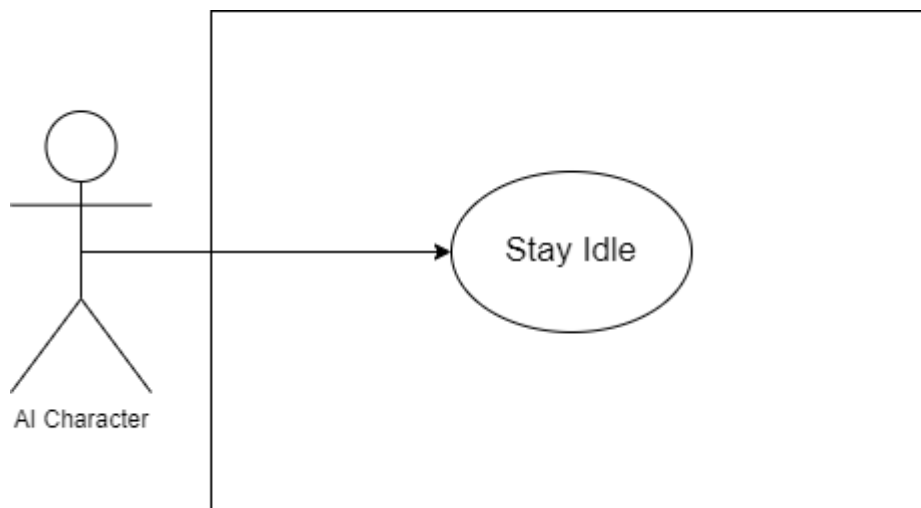


Figure 39: Use case diagram of “Stay Idle”.

Use Case	Stay Idle
Actor	AI Character
Precondition	The game is started
Postcondition	The AI Character Stay in the “Idle” state
Description	When the game starts, AI characters that are far from the player character stay idle, waiting for him to activate their other states.

Table 23: Table of Use case “Stay Idle”.

4. Sprint 2 realization

4.1 Artificial Intelligence in videogames

Artificial intelligence (AI) in videogames makes the game more fun. It helps non-player characters (NPCs) act like real people and interact logically. It also balances the game to be fair for the player. Many think recent games have super advanced AI, but developers often keep AI simple to control the player's experience better. The goal of AI is to keep players engaged and having fun, not to create unbeatable opponents.

When you start a new game, would you rather be constantly beaten or matched with something at your skill level so you can learn and improve? Most players prefer the second option.

This means we don't need the smartest AI in games. We need AI that makes the game enjoyable and fun to play.

4.2 AI Uses in “Wild World”

In our game, you'll encounter AI in different situations such as:

4.2.1 Movement and navigation between destinations:

The movements of bots in the scene, obstacle avoidance, moving from one point to another, rotating towards the player's position, and random patrols.

Unity's NavMesh navigation system allows characters to intelligently navigate the game world using navigation meshes created from the scene's geometry.

Unity's NavMesh consists of the following elements:

-NavMesh (short for Navigation Mesh) is a data structure that describes the walkable surfaces of the game world and enables finding a path from one accessible location to another. It's built or baked from the level geometry.

-The NavMesh Agent component helps create characters that move towards their goal. Agents navigate the game world using NavMesh and know how to avoid each other and moving obstacles.

-The Off-Mesh Link component incorporates navigation shortcuts that can't be represented by walkable surfaces. For example, jumping over a fence or opening a door before passing through it can be described as Off-Mesh links.

-The NavMesh Obstacle component describes moving obstacles that agents must avoid when navigating the world. For example, a barrel controlled by the physics system. As the obstacle moves, agents do their best to avoid it. Once the obstacle becomes stationary, it creates a hole in the NavMesh so that agents can change their path to go around it. If the stationary obstacle blocks the path, agents can find a different route.

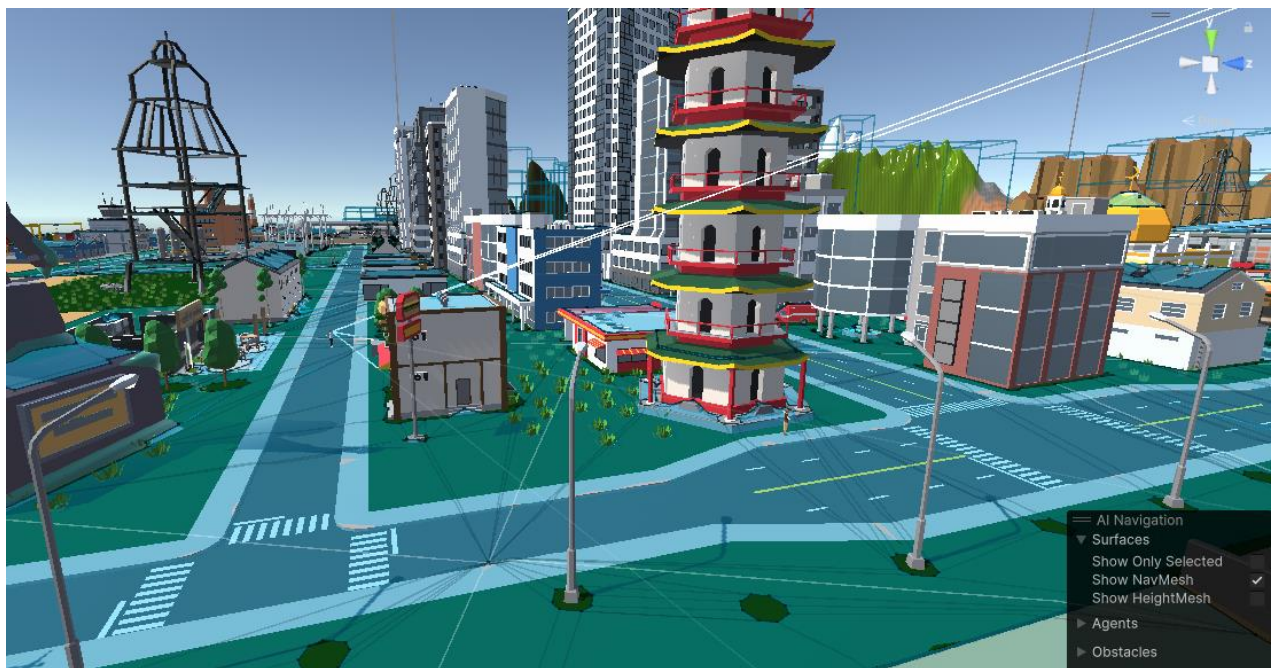


Figure 40: NavMesh.

4.2.2 Attack, Aim, and Shoot:

Aiming and shooting bullets towards the player's position, checking for collisions, creating bullets, and removing them upon collision.



Figure 41: Example of AI character aiming.

4.3 AI animation

Unity features a rich and sophisticated animation system that includes:

- Easy workflow and setup for animations on all Unity elements, including objects, characters, and properties.
- Support for imported animation clips and animations created within Unity.
- Humanoid animation retargeting the ability to apply animations from one character model to another.
- Convenient preview of animation clips, transitions, and interactions between them. This allows us to prototype and preview animations before hooking up game code.
- Animation of different body parts with different logic.
- Overlay and masking functions.

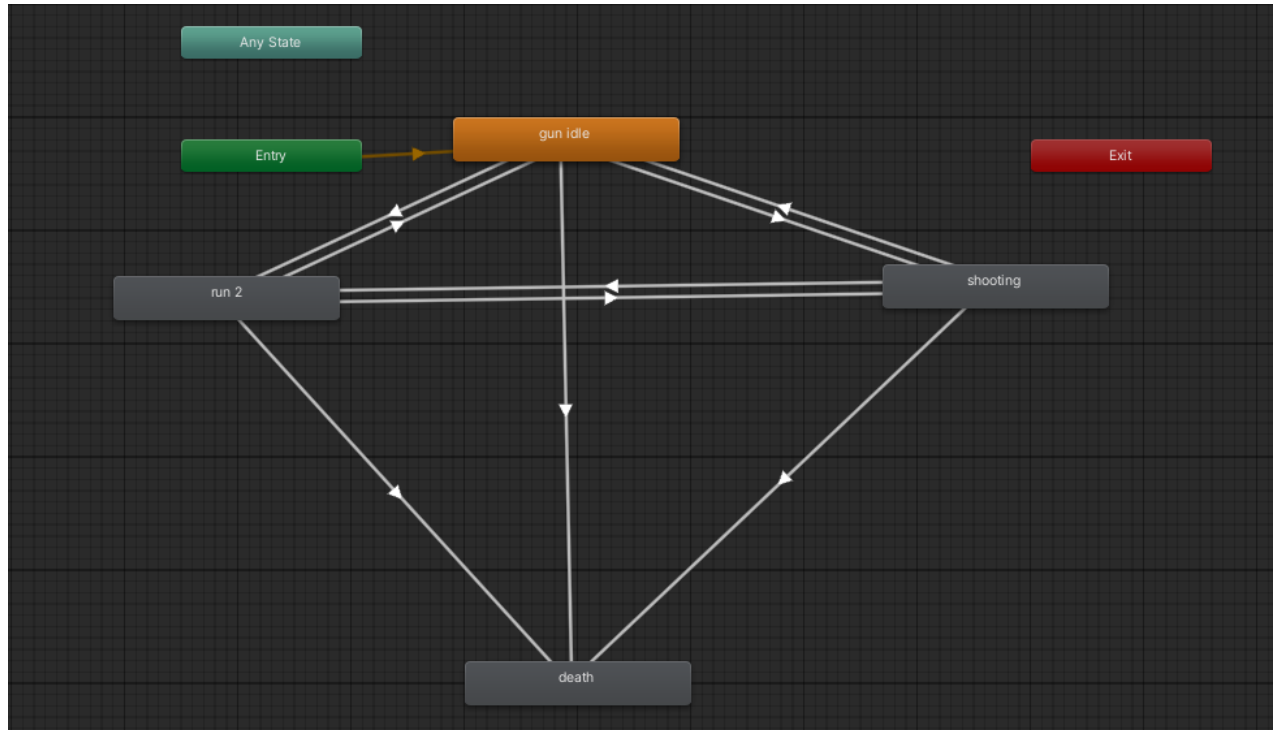


Figure 42:example of AI character animation.

This diagram illustrates how animation functions, it begins with the idle animation. When the player approaches, the animation changes based on the situation.

Conclusion

In this chapter, we focused on the second sprint. Following the same approach as the previous chapter, we began by identifying the backlog for sprint 2 and refining the use cases. Then, we modeled the second part of our system. Finally, we presented the implementation with a particular emphasis on the use of AI in the field of videogames and especially in our project.

Chapter V Sprint 3: Main Menu

1.Introduction

This chapter covers our final sprint. We will present the product backlog, a detailed study of the player's functional needs. Then, we will discuss the design and finally the implementation.

2.Backlog

ID	User Story	Priority
27	As a User, I can start the game	++
28	As a User, I can save the game	++
29	As a User, I can load a save game	++
30	As a User, I can quit the game	+

Table 24: Table of sprint 3 backlog product.

3.Use case diagram

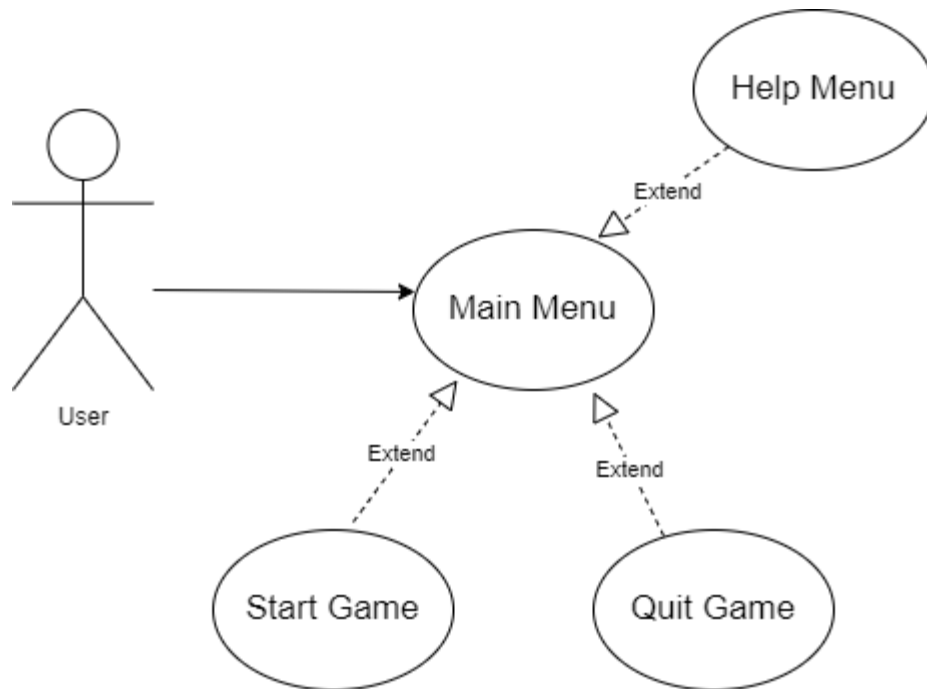


Figure 43: Use case diagram of the user.

Use Case	Main Menu
Actor	User
Precondition	The game startup
Postcondition	Transition to the main menu
Description	The user accesses the main menu, allowing him to choose between starting a new game, continuing a previous save, accessing the help interface, or quitting the game.

Table 25: Table of Use case of the user.

4. Sprint 3 realization

4.1 Main Menu



Figure 44: Main menu.

ID	Description
1	Continue button
2	New game + button
3	Help menu button
4	Quit button

Table 26: Table of main menu.

4.2 Help Menu



Figure 45: Help menu.

The help interface, as its name suggests, is designed to assist the player in adapting to the game mechanics and answering their questions. It covers topics such as:

- How to move
- How-to pick-up items and access the inventory to use and equip them.
- What do the colored indicators in the top corners of the screen mean.
- How to attack enemies

Conclusion

In this chapter, we presented the third sprint of our project, which focuses mainly on the interfaces for the main menu and help menu, as well as the functionalities for starting and quitting the game. We began by looking at the backlog for this sprint, the use case diagram, and its refinements, followed by the implementation.

General Conclusion

CGI STUDIO assigned me my final year project, which was to design, develop, and create a videogame called "Wild World." I'm very satisfied and proud of my work, as I achieved my goals and appreciated the opportunity to work on this project.

This report outlines all the steps involved in creating a 3D mobile videogame at CGI STUDIO. We used the Scrum methodology to help us organize the work.

First, we specified the requirements and general design, and we described the technologies we used. We developed our game in three sprints.

In the first sprint, I prioritized features that interact with the player character. The second sprint focused on using artificial intelligence for non-player characters. The third sprint connected the different parts of the game.

This project was a great opportunity to deepen my professional and personal knowledge and to learn innovative technologies like Unity.

While my solution meets the company's current needs, it could be improved to help CGI STUDIO achieve its long-term goals. I plan to add more features to make the game more immersive and enjoyable.

Webography

- [1] <https://www.minecraft.net/en-us>
- [2] <https://www.ubisoft.com/en-gb/game/far-cry/far-cry-6>
- [3] <https://www.livementor.com/blog/methodologie-agile>
- [4] <https://www.cloudsmart.lu/page/methode-scrum>
- [5] <https://unity.com/>
- [6] <https://www.adobe.com/africa/products/photoshop.html>
- [7] <https://visualstudio.microsoft.com/>
- [8] <https://www.blender.org/>
- [9] <https://app.diagrams.net/>
- [10] <https://www.mixamo.com/>
- [11] <https://docs.microsoft.com/en-us/dotnet/csharp/>
- [12] <https://www.uml.org/>
- [13] <https://assetstore.unity.com/>
- [14] <https://assetstore.unity.com/packages/3d/props/low-poly-ultimate-pack-54733>
- [15] <https://assetstore.unity.com/packages/essentials/starter-assets-character-controllers-urp-267961>

VIDEO GAME DEVELOPMENT

Rami Chaaben

الخلاصة: تم كتابة هذا التقرير كجزء من مشروع نهاية الدراسة للحصول على شهادة في علوم الكمبيوتر والوسائط المتعددة. الهدف الرئيسي هو تطوير لعبة فيديو من منظور الشخص الأول بدون اتصال بالإنترنت باستخدام تصميم منخفض التعدد. يعتمد تحقيق هذا التطبيق بشكل أساسي على محرك الألعاب Unity ولغة البرمجة C#.

Résumé: Ce rapport a été rédigé dans le cadre du projet de fin d'études afin d'obtenir une licence en informatique et multimédia. L'objectif principal est de développer un jeu vidéo en vue subjective hors ligne utilisant un design low poly. La réalisation de cette application repose principalement sur le moteur de jeu Unity et le langage C#.

Abstract: This report was written as part of the end-of-study project in order to obtain license in computer science and multimedia. the main objective is to develop an offline first-person videogame using low poly design. The realization of this application is mainly based on Unity game engine and C#.

المفاتيح: Unity ، C# ، الذكاء الاصطناعي (AI)، الرسوم المتحركة، لعبة تصويب من منظور الشخص الأول (FPS)، لعبة بدون اتصال بالإنترنت، تصميم منخفض التعدد (Low Poly)، تطوير الألعاب

Mots clés: Unity, C#, IA, Animation, Jeu De Tir à la Première Personne (FPS), Jeu Hors Ligne, Design Low Poly, Développement de Jeux

Key-words: Unity, C#, AI, Animation, First-Person Shooter (FPS), Offline Game, Low Poly Design, Game Development