МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри_____Є. О. Давиденко

«____»_____2023 р.

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

Ігровий застосунок на основі рушія Unreal Engine

Спеціальність «Інженерія програмного забезпечення»

121 – КРБ.1 – 408.21910822

Студент _____***Д. І. Ткач***

«___» червня 2023 р.

Керівник ст. викладач _____ ***С. Ю. Боровльова***

«___» червня 2023 р.

Консультант канд. техн. наук, доцент _____ ***А. О. Алексєєва***

«___» червня 2023 р.

Миколаїв – 2023

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри інженерії програмного забезпечення, канд.техн.наук, доцент,

_____Є.О. Давиденко

«____»_____2023 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи бакалавра

Видано студенту групи 408 факультету комп'ютерних наук

Ткач Дмитро Ігорович.

(прізвище, ім'я, по батькові студента)

1. Тема кваліфікаційної роботи

«Ігровий застосунок на основі рушія Unreal Engine» .

Затверджена наказом по ЧНУ від «17» березня 2023 р. № 60

2. Строк представлення кваліфікаційної роботи «____» _____2023 р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні

Вхідні дані до роботи − функціональні та нефункціональні вимоги до ігрового застосунку. Результат – функціонуючий ігровий застосунок .

4. Перелік питань, що підлягають розробці

- аналіз сучасних автоматизованих ресурсів для створення ігрового застосунку. Визначити переваги та недоліки даної платформи;
- специфікація вимог до ігрового застосунку;

– проєктування та моделювання ігрового застосунку;

– розробка функціональних модулів ПЗ та проєктування зручного та інтуїтивно зрозумілого інтерфейсу та опцій, який дозволить споживачам легко розібратися в управлінні та проходження ігрового застосунку.;

– Тестування та апробація ПЗ

5. Перелік графічних матеріалів:

Презентація.

6. Завдання до спеціальної частини

Дослідження питань охорони праці, які безпосередньо пов'язані з діяльністю розробника програмного забезпечення.

7. Консультанти:

Консультант	Кафедра (організація)	Частина роботи
Алексєєва А.О., канд. техн. наук, доцент (б. в. з.)	Кафедра екології Медичного інституту ЧНУ ім. Петра Могили	Спеціальна частина з охорони праці

Керівник роботи ст. викладач Боровльова Світлана Юріївна
(посада, прізвище, ім'я, по батькові)

Завдання прийнято до виконання *(підпис)*

Ткач Дмитро Ігорович
(прізвище, ім'я, по батькові студента)

Дата видачі завдання «_____» _____
2023 р *(підпис)*

КАЛЕНДАРНИЙ ПЛАН

виконання кваліфікаційної роботи

Тема: «Ігровий застосунок на основі рушія Unreal Engine»

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Розробка та затвердження завдання на виконання КРБ	04.10.2022	10.10.2022	Виконано
2.	Огляд літератури за темою роботи	11.10.2022	17.10.2022	Виконано
3.	Складання календарного плану КРБ	18.10.2022	24.10.2022	Виконано
4.	Аналіз предметної області	17.01.2022	30.01.2022	Виконано
5.	Розробка проєктних рішень	31.01.2023	23.02.2023	Виконано
6.	Моделювання та конструювання ПЗ	11.04.2022	17.04.2023	Виконано
7.	Кодування, тестування та апробація розробленого ПЗ, аналіз результатів тестування, розробка керівництва користувача	18.04.2023	08.05.2023	Виконано
8.	Розробка спеціальної частини з охорони праці	09.05.2023	15.05.2023	Виконано
9.	Відгук керівника КРБ	16.05.2023	17.05.2023	Виконано
10.	Оформлення КРБ та презентації	18.05.2023	22.05.2023	Виконано
11.	Попередній захист	23.05.2023	25.05.2023	Виконано
12.	Рецензування	26.05.2023	16.06.2023	Виконано
13.	Завершення оформлення КРБ та презентації	20.06.2023	16.06.2023	Виконано
14.	Захист кваліфікаційної роботи	23.06.2023	27.06.2023	

Розробив студент Ткач Дмитро Ігорович _____
(прізвище, ім'я, по батькові студента) (*підпис*)

«____» _____2023р.

Керівник роботи ст. викладач Боровльова Світлана Юріївна _____
(посада, прізвище, ім'я, по батькові) (*підпис*)

«____» _____2023р.

АНОТАЦІЯ

до кваліфікаційної роботи бакалавра

«Ігровий застосунок на основі рушія Unreal Engine»

Студент 408 гр.: Ткач Дмитро Ігорович

Керівник: ст. викладач кафедри інженерії програмного забезпечення факультету комп'ютерних наук Боровльова Світлана Юріївна

Зважаючи на все більшу популярність ігор у світі, розробка програмного забезпечення на платформі Unreal Engine для створення ігрових застосунків є актуальною темою. Unreal Engine є потужним інструментом для створення ігор різних жанрів, таких як екшн, рольові ігри, стратегії та інші.

За допомогою Unreal Engine можна розробити ігри з високоякісною графікою та реалістичною фізикою, що робить гру більш привабливою для користувачів. Більш того, Unreal Engine надає можливість розробникам створювати гри для різних платформ, включаючи ПК, консолі, мобільні пристрої та інші.

Об’єкт кваліфікаційної роботи – процеси, що пов’язані із організацією та створенням ігрового застосунку.

Предмет кваліфікаційної роботи – інструментальні засоби та інформаційні технології розробки ігрового застосунку.

Метою кваліфікаційної роботи є популяризація відеогри серед непрофесійних гравців, шляхом створення проєкту в Unreal Engine з простим але інтенсивним геймплеєм та атмосферним сеттінгом, що задовольнить потреби таких користувачів та буде позитивно впливати на їх психологічний стан.

Для досягнення цієї мети необхідно вирішити наступні завдання:

1. Провести аналіз сучасних автоматизованих ресурсів для створення ігрового застосунку. Визначити переваги та недоліки даної платформи.

2. Сформулювати вимоги до ігрового застосунку.

3. Спроєктувати та змоделювати ігровий застосунок.

4. Розробити функціональні модулі ПЗ та спроєктувати зручний та інтуїтивно зрозумілий інтерфейс, який дозволить споживачам легко розібратися в управлінні та проходженні ігрового застосунку.

У першому розділі КРБ проведено аналіз сучасних ігрових застосунків та ресурсів, що розроблені на базі рушія Unreal Engine. Оцінено сильні та слабкі сторони кожного застосунку, проведено порівняльний аналіз.

У другому розділі визначено вимоги до програмного забезпечення ігрового застосунку на основі результатів аналізу сучасних ігрових застосунків та ресурсів, що розроблені на базі рушія Unreal Engine.

Третій розділ включає проєктування та моделювання програмного забезпечення для ігрового застосунку на базі рушія Unreal Engine. Це було виконано, включаючи створення архітектурної діаграми та діаграми потоків даних, діаграми варіантів використання та інших відповідних моделей.

Четвертий розділ охоплює розробку функціональних програмних модулів та створення зручного та інтуїтивно зрозумілого інтерфейсу, щоб дозволити гравцям легко керувати грою.

Результатом КРБ є функціональний ігровий застосунок, розроблений на базі рушія Unreal Engine в процесі комплексного аналізу, проєктування та розробки.

КРБ викладена на 71 сторінках, вона містить 4 розділи, 25 ілюстрацій, 9 таблиць, 10 джерел в переліку посилань

Ключові слова: ігровий застосунок, Unreal Engine, графічний рушій, розробка гри, віртуальна реальність.

ABSTRACT

to the bachelor's qualification work

"Game application based on the Unreal Engine"

Student of 408 group: Tkach Dmytro Ihorovych

Supervisor: Senior Lecturer of the Department of Software Engineering,

Faculty of Computer Science Borovleva Svitlana Yurievna

Given the increasing popularity of games in the world, software development on the Unreal Engine platform for creating game applications is a hot topic. Unreal Engine is a powerful tool for creating games of various genres, such as action, role-playing, strategy, and others.

With the help of Unreal Engine, you can develop games with high-quality graphics and realistic physics, which makes the game more attractive to users. Moreover, Unreal Engine allows developers to create games for various platforms, including PC, consoles, mobile devices, and others.

The object of qualification work is the processes associated with the organization and creation of a game application.

The subject of qualification work is tools and information technologies for developing a game application.

The purpose of the qualification work is to popularize the video game among non-professional players by creating a project in Unreal Engine with simple but intense gameplay and atmospheric setting that will meet the needs of such users and have a positive impact on their psychological state.

To achieve this goal, the following tasks need to be solved:

1. Analyze modern automated resources for creating a gaming application. Identify the advantages and disadvantages of this platform.

2. Specification of requirements for the game application.

3. Design and modeling of the game application.

4. Development of functional software modules and design of a convenient and intuitive interface and options that will allow consumers to easily understand the management and passage of the gaming application.

The first section of the CRB analyzes modern gaming applications and resources developed on the basis of the Unreal Engine. The strengths and weaknesses of each application are assessed, and a comparative analysis is conducted.

The second section defines the requirements for the software of the game application based on the results of the analysis of modern game applications and resources developed on the basis of the Unreal Engine.

The third section includes the design and modeling of the software for the game application based on the Unreal Engine. This was accomplished by creating an architecture and data flow diagram, use case diagram, and other relevant models.

The fourth section covers the development of functional program modules and the creation of a user-friendly and intuitive interface to allow players to easily control the game.

The result of the CRB is a functional gaming application developed on the basis of the Unreal Engine in the process of comprehensive analysis, design and development.

The CRB is set out on 71 pages, it contains 4 sections, 25 illustrations, 9 tables, 10 sources in the list of references

Keywords: game application, Unreal Engine, graphics engine, game development, virtual reality.

ЗМІСТ

ВСТУП

Unreal Engine - це популярний ігровий рушій, розроблений Epic Games, який надає набір потужних інструментів для створення та розробки ігор на широкому спектрі платформ, включаючи ПК, консолі та мобільні пристрої. Він широко визнаний як один з найпотужніших і найгнучкіших ігрових рушіїв на сьогоднішній день, що пропонує розробникам можливість створювати високоякісні, захоплюючі ігри з приголомшливою графікою і реалістичною фізикою [1].

Однією з ключових особливостей Unreal Engine є його система візуальних сценаріїв Blueprint, яка дозволяє розробникам створювати складну ігрову логіку та поведінку без необхідності традиційного програмування. Це робить його доступним інструментом для розробників усіх рівнів кваліфікації, від початківців до досвідчених професіоналів.

Ще однією важливою особливістю Unreal Engine є підтримка розробки віртуальної реальності (VR) та доповненої реальності (AR), що зробило його популярним вибором для розробників, які прагнуть створювати ігри з ефектом занурення в ці нові технології. Рушій також пропонує підтримку передових фізичних симуляцій, анімації та візуальних ефектів, що дає розробникам можливість створювати високодеталізоване та реалістичне ігрове середовище [2].

Незважаючи на численні переваги, розробка ігрових додатків у середовищі Unreal Engine пов'язана з певними труднощами. Однією з найбільших проблем є складність рушія, яка може бути непосильною для розробників, які не знайомі з його численними функціями та можливостями. Крім того, високі системні вимоги, необхідні для запуску рушія, можуть стати бар'єром для невеликих розробників, які не мають доступу до висококласного обладнання [3].

Незважаючи на ці проблеми, Unreal Engine використовується для створення деяких з найпопулярніших і найуспішніших ігор на ринку,

включаючи Fortnite, Gears of War і Unreal Tournament. Його універсальність і потужність зробили його основним інструментом для розробників, які прагнуть створювати інноваційні та захопливі ігри [4].

Загалом, Unreal Engine справив значний вплив на ігрову індустрію, надавши розробникам потужний та гнучкий інструментарій для створення інноваційного та захопливого ігрового досвіду. Вивчивши його особливості, можливості та проблеми, ви зможете глибше зрозуміти роль, яку відіграють ігрові програми в середовищі Unreal Engine у формуванні майбутнього ігор [5].

Об’єкт кваліфікаційної роботи – процеси, що пов’язані із організацією та створенням ігрового застосунку.

Предмет кваліфікаційної роботи – інструментальні засоби та інформаційні технології розробки ігрового застосунку.

Метою кваліфікаційної роботи є популяризація відеогри серед непрофесійних гравців, шляхом створення проєкту в Unreal Engine з простим але інтенсивним геймплеєм та атмосферним сеттінгом, що задовольнить потреби таких користувачів та буде позитивно впливати на їх психологічний стан [6].

Для досягнення цієї мети необхідно вирішити наступні завдання:

1. Провести аналіз сучасних автоматизованих ресурсів для створення ігрового застосунку. Визначити переваги та недоліки даної платформи.

2. Специфікація вимог до ігрового застосунку.

3. Проєктування та моделювання ігрового застосунку.

4. Розробка функціональних модулів ПЗ та проєктування зручного та інтуїтивно зрозумілого інтерфейсу та опцій, який дозволить споживачам легко розібратися в управлінні та проходження ігрового застосунку.

5. Тестування та апробація ПЗ.

1 АНАЛІЗ ПРЕДМЕТНОЇ СФЕРИ РОЗРОБКИ КОМП'ЮТЕРНИХ ІГОР

1.1 Опис предметної сфери розробки комп'ютерних ігор

Предметна сфера розробки комп'ютерних ігор охоплює безліч аспектів, пов'язаних зі створенням, поширенням і використанням ігор. Це галузь, яка об'єднує розробників, видавців, інвесторів, маркетологів, дизайнерів, музикантів і багатьох інших професіоналів, які працюють над створенням ігрового контенту [2].

Важливим аспектом розробки комп'ютерних ігор є використання передових технологій, включно з графічними рушіями, звуковими технологіями, штучним інтелектом, хмарними технологіями та багато іншого. Розробники використовують ці технології для створення ігрових світів, які дають змогу гравцям зануритися в унікальний досвід гри.

Крім того, розробка комп'ютерних ігор має величезне значення для культурної сфери, оскільки ігри стали невід'ємною частиною сучасної культури та розваг. Вони є формою мистецтва і розваги, яка надає унікальну можливість для передачі ідеологій, цінностей і культурних норм [7].

Комп'ютерні ігри також мають соціальне значення, оскільки вони є потужним інструментом соціальної інтеракції та комунікації. Багато ігор дають змогу гравцям об'єднуватися в команди та взаємодіяти один з одним, що може зміцнити соціальні зв'язки та поліпшити комунікацію.

Нарешті, розробка комп'ютерних ігор має величезний економічний потенціал. Комп'ютерні ігри є багатомільярдною галуззю, яка включає в себе безліч різних компаній та ігрових студій. Ця галузь генерує безліч робочих місць і прибутку, залучаючи інвесторів і спонсорів [5].

Загалом, предметна сфера розроблення комп'ютерних ігор - це величезна та динамічна галузь, яка має важливе значення для культурного, соціального та економічного життя. Вона об'єднує технології, культурні аспекти та соціальні питання.

Важливим аспектом розробки комп'ютерних ігор є також їхній маркетинг і просування. В індустрії комп'ютерних ігор існує величезна конкуренція, і розробники повинні вміти виділити свій продукт на тлі інших ігор і привернути увагу гравців. Для цього використовуються різні стратегії маркетингу, включно з рекламою в соціальних мережах, партнерськими програмами, тизерами, трейлерами та багатьма іншими [8].

Також важливим аспектом розробки комп'ютерних ігор є їхня технічна підтримка та оновлення. Ігри мають бути оновлюваними та підтримуватися протягом тривалого часу після випуску, щоб задовольняти запити гравців і вирішувати проблеми, що виникають. Розробники також повинні вміти забезпечувати безпеку ігрових світів і даних користувачів [4].

Загалом, предметна сфера розроблення комп'ютерних ігор є динамічною та багатогранною галуззю, що поєднує в собі технології, культурні та соціальні аспекти. Вона має важливе значення для сучасної культури та розваг, а також являє собою величезний економічний потенціал.

Одним із ключових чинників успіху в розробці комп'ютерних ігор є вміння розуміти та передбачати вподобання та вимоги гравців. Розробники повинні враховувати різні культурні та соціальні контексти, а також тенденції в індустрії та зміни в поведінці споживачів. Ігри мають бути цікавими, такими, що затягують, і надавати гравцям нові враження та можливості [2].

Для того щоб створювати інноваційні та успішні комп'ютерні ігри, розробники повинні постійно стежити за новими технологіями та тенденціями в індустрії. Наприклад, останнім часом велика увага приділяється розробці віртуальної та доповненої реальності, і ігри, які використовують ці технології, стають дедалі популярнішими. Також з'являються нові можливості для використання штучного інтелекту і машинного навчання в розробці ігор.

Таким чином, предметна сфера розробки комп'ютерних ігор є важливою та перспективною галуззю, яка поєднує в собі технології, культурні та соціальні аспекти. Розробники ігор повинні вміти передбачати вимоги гравців, слідкувати за новими технологіями та тенденціями, а також враховувати свою

відповідальність перед суспільством. Комп'ютерні ігри відіграють важливу роль у житті мільйонів людей, і їхній вплив на культуру та суспільство триватиме в майбутньому [9].

1.2 Етапи розробки комп'ютерних ігор

Розробка комп'ютерних ігор є складним процесом, який включає в себе безліч етапів. Кожен етап важливий для створення якісної гри, яка буде популярна серед гравців.

Перший етап - концепція. На цьому етапі розробники визначають ідею гри, її основну механіку, стиль і жанр. Вони також визначають цільову аудиторію і створюють документ, який містить основні ідеї гри та її особливості. Цей документ називається концепцією гри і є відправною точкою для всього процесу розробки [2].

Другий етап - прототипування. На цьому етапі розробники створюють прототип гри. Прототип - це базова версія гри, яка дає змогу розробникам випробувати основні механіки гри та визначити, наскільки гра цікава та захоплива. Це допомагає розробникам переконатися в правильності вибору концепції гри і зробити коригування, якщо це необхідно.

Третій етап - дизайн. На цьому етапі розробники створюють дизайн гри, включно з графікою, звуком, інтерфейсом та іншими елементами гри. Дизайн - це важливий аспект гри, який визначає її візуальний стиль і атмосферу. Він також включає в себе розробку рівнів, місій та інших ігрових елементів [3].

Четвертий етап - програмування. На цьому етапі розробники починають створювати код гри. Вони використовують різні мови програмування та інструменти для створення ігрових механік, управління персонажами та інших елементів гри. Цей етап є одним із найважливіших і найтриваліших у процесі розробки.

П'ятий етап - тестування. На цьому етапі розробники тестують гру на наявність помилок, багів та інших проблем. Вони використовують різні інструменти для тестування, такі як тестування функціоналу, ігрове

тестування, тестування сумісності та інші. Цей етап допомагає розробникам виправити всі помилки та покращити якість гри [7].

Шостий етап - випуск. На цьому етапі гра готова до випуску. Розробники готують гру до випуску, включно з її пакуванням і розміщенням на різних платформах. Вони також рекламують гру, щоб привернути увагу гравців. Коли гру випущено, розробники продовжують її підтримувати, випускаючи патчі та оновлення для поліпшення ігрового процесу.

Важливо зазначити, що кожен етап розробки гри тісно пов'язаний з іншими етапами. Наприклад, прототипування дає змогу визначити правильність вибору концепції гри та зробити необхідні коригування в дизайні та програмуванні. Тестування дає змогу виявити помилки та проблеми в грі, які розробники можуть виправити на етапі програмування [3].

Крім того, у процесі розробки ігор часто виникають несподівані проблеми, які можуть торкнутися всіх етапів розробки. Наприклад, зміни в технологіях або платформах, на яких випускається гра, можуть вимагати додаткових зусиль з боку розробників. Також може виникнути необхідність змінити дизайн гри або її механіку у зв'язку зі змінами в конкурентному середовищі або відгуками користувачів [4].

У підсумку, кожен етап розробки гри є важливим і необхідним для створення якісної гри. Концепція гри дає змогу визначити ідею гри та її особливості, прототипування допомагає випробувати механіки гри та ввести необхідні корективи, дизайн створює візуальний стиль та атмосферу гри, програмування дає змогу створити ігрові механіки та керувати персонажами, тестування допомагає виявити та виправити помилки й проблеми в грі, а випуск - запустити гру на ринок і продовжити її підтримку.

Загалом, розробка комп'ютерних ігор - це складний процес, який вимагає від розробників творчого підходу та технічних знань. Кожен етап розробки гри є важливим і необхідним для створення якісної та успішної гри, яка буде популярною серед гравців [8].

Крім того, розробка комп'ютерних ігор не обмежується лише створенням однієї гри. Розробники можуть створювати серії ігор, розширювати можливості наявних ігор і створювати додатковий контент для них. Також існують ігрові рушії та інструменти, які допомагають прискорити процес розробки ігор і знизити витрати на їх створення.

Нові технології та платформи, як віртуальна реальність або штучний інтелект, надають нові можливості для створення унікальних ігор, але також потребують додаткових знань і навичок. Швидкий технологічний розвиток також означає, що гравці стають дедалі більш вимогливими до якості та рівня інновацій в іграх, що призводить до збільшення витрат на розробку ігор [2].

Ще одним викликом є конкурентне середовище. Кількість ігор швидко зростає, і розробникам потрібно зробити все можливе, щоб виділити свою гру серед величезної кількості інших. Крім того, ігрові платформи також змагаються одна з одною за залучення гравців, що призводить до ускладнення процесу випуску ігор на різних платформах.

Кожен етап розробки гри є важливим і необхідним для створення якісної та захопливої гри. Швидкий технологічний розвиток і конкурентне середовище становлять виклики для розробників, але також надають можливості для створення нових та унікальних ігор [5].

Крім того, важливо враховувати потреби та очікування гравців під час розробки ігор. Ігрова індустрія постійно змінюється та розвивається, і гравці стають дедалі вимогливішими. Тому розробники повинні не тільки враховувати нові технології та тенденції, а й активно досліджувати потреби та вподобання гравців, щоб створювати ігри, які будуть захопливими та цікавими для аудиторії.

Загалом, розробка комп'ютерних ігор - це процес, який вимагає безлічі зусиль і витрат, але який може призвести до створення унікальних, захопливих ігор, що можуть потішити та розважити мільйони гравців по всьому світу. Важливо пам'ятати, що кожен етап розроблення гри важливий і необхідний

для створення якісної гри, і що врахування потреб та очікувань гравців є ключовим фактором успіху в ігровій індустрії [9].

1.3 Класифікація ігрових застосунків

У сучасному світі комп'ютерні ігри є однією з найпопулярніших розваг. Ринок ігрових застосунків розвивається з неймовірною швидкістю, і щороку з'являються нові ігри та жанри. Для того щоб чіткіше визначитися в цьому розмаїтті, існує класифікація ігрових застосунків. Класифікація ігрових застосунків може ґрунтуватися на різних критеріях, таких як жанр, тип пристрою, цільова аудиторія та багато інших. На таблиці 1.1 наведено опис жанрів ігрових застосунків.

Таблиця 1.1 – Класифікація ігрових застосунків

Жанр	Опис
Стратегії	Передбачає планування та керування різними ресурсами з метою розвитку, будівництва та керування власними територіями та арміями, а також стратегічне планування бойових дій з метою перемоги над противником.
Головоломки	Передбачає вирішення складних логічних завдань та головоломок, які вимагають від гравця зосередженості, терпіння та кмітливості. Гравці повинні розв'язувати різноманітні головоломки та завдання, щоб продовжити гру та досягти кінцевої мети.
Шутери	Передбачає геймплей, зосереджений на використанні різноманітної зброї та стрільбі на ворогів або інших цілей. Гравці можуть грати як з першої, так і з третьої особи, а головною метою є подолання різних рівнів та місій, знищення ворогів та досягнення кінцевої мети гри.

Кінець таблиці 1.1

RPG	Жанр RPG (рольових ігор) передбачає геймплей, де гравці беруть на себе роль героя і розвивають його, здійснюючи вибір та виконуючи завдання в містичному чи фантастичному світі.
Спортивні ігри	Жанр спортивних комп'ютерних ігор передбачає відтворення різноманітних видів спорту на комп'ютері, де гравці можуть брати участь у віртуальних змаганнях та турнірах, змагатися з іншими гравцями або з комп'ютером.

Кожен жанр характеризується своїми особливостями і механіками гри, а також своєю цільовою аудиторією. Наприклад, стратегії зазвичай адресовані більш дорослій аудиторії, яка шукає більш складні ігри, в той час як ігри-головоломки можуть бути цікаві і дітям, і дорослим [3].

Інший важливий критерій класифікації ігрових застосунків - це тип пристрою, на якому гра запускається. На таблиці 1.2 наведено опис пристроїв.

Таблиця 1.2 – Опис пристроїв

Тип пристрою	Особливості
ПК	Високоякісна графіка, більш детальне налаштування контролерів, можливість модифікацій.
Консолі	Простота використання, ексклюзивність, розвинутий мультиплеєр.
Мобільні пристрої	зазвичай мають простіший геймплей і коротшу тривалість, оскільки вони призначені для швидкого ігрового досвіду в дорозі.

Кожен тип пристрою має свої особливості, і розробники ігор враховують їх під час створення ігор.

Цільова аудиторія також є важливим критерієм класифікації ігрових застосунків. Розробники ігор зазвичай орієнтуються на певну групу користувачів під час створення ігор. Наприклад, ігри для дітей мають бути простішими та менш насильницькими, а ігри для дорослих можуть містити складніший сюжет і більш насильницькі елементи [1].

Ще одним важливим критерієм класифікації ігрових застосунків є орієнтація на певний ринок. Наприклад, ігри для PC можуть бути орієнтовані на західний ринок, а ігри для мобільних пристроїв - на азіатський ринок.

На таблиці 1.3 наведено класифікацію ігрових застосунків за їхнім форматом.

Таблиця 1.3 – Класифікація ігрових застосунків за їхнім форматом

Формат гри	Особливості
Одиночні ігри	Призначені для одного гравця і можуть бути як лінійними, так і нелінійними.
Багатокористувацькі ігри	Дають змогу кільком гравцям грати разом на одному пристрої.
Онлайн ігри	Дають змогу грати з іншими гравцями через Інтернет.

Ще один критерій класифікації ігрових застосунків – це ступінь складності гри. Серед ігор можна виділити ігри для початківців, для досвідчених гравців і для професійних гравців. Ігри для початківців зазвичай мають простіший рівень складності та навчають гравців основ ігрового процесу, тоді як ігри для досвідчених гравців можуть бути складнішими та містити більш просунуті механіки гри [3].

Нарешті, важливим критерієм класифікації ігрових застосунків є їхня спрямованість. Існує безліч напрямів ігрових застосунків, таких як симулятори, перегони, спортивні ігри, ігри на виживання, ігри на тему фентезі та багато інших. Кожен напрямок має свої особливості та дає змогу гравцям зануритися в різні світи й атмосфери [7].

Таким чином, класифікація ігрових застосунків має безліч критеріїв і дає змогу чіткіше визначитися в розмаїтті ігрових жанрів і напрямів. Кожен критерій має свої особливості та допомагає розробникам ігор точніше орієнтуватися на свою цільову аудиторію. У сучасному світі ігрові застосунки стають дедалі популярнішими та різноманітнішими, і класифікація ігрових застосунків допомагає гравцям вибрати найбільш підходящу для їхніх потреб гру, а розробникам - визначитися з цільовою аудиторією та найефективніше просувати свій продукт [9].

Важливо зазначити, що класифікація ігрових застосунків є динамічною та постійно розвивається. З розвитком технологій і зміною потреб гравців, з'являються нові жанри та напрямки ігрових застосунків. Наприклад, із появою віртуальної та доповненої реальності стали популярними ігри, що базуються на використанні цих технологій, такі як Pokémon Go і Beat Saber.

На рисунку 1.1 наведено приклад доповненої реальності.

Рисунок 1.1 – Гра Pokémon Go

У висновку можна сказати, що класифікація ігрових застосунків є важливим інструментом для гравців і розробників. Вона дає змогу чіткіше визначитися з цільовою аудиторією та орієнтуватися на її потреби й інтереси, а також ефективніше просувати ігровий продукт. Важливо пам'ятати, що класифікація ігрових застосунків є динамічною, постійно розвивається, і разом

із технологічними змінами та зміною потреб гравців з'являються нові жанри та напрямки ігрових за стосунків [5].

1.4 Специфікація вимог до програмного забезпечення ігрового застосунку

Функціональні вимоги

Ігровий процес

Ігровий застосунок повинен містити основну квестову лінію з декількома цілями та підквестами.

Гравець повинен мати можливість керувати рухом персонажа за допомогою клавіатури або сенсорного екрану.

Ігровий застосунок повинен включати бойові механіки, які дозволяють гравцеві вступати в бої з ворогами.

Гравець повинен мати можливість взаємодіяти з навколишнім середовищем та об'єктами в межах ігрового світу.

Ігровий застосунок повинен містити елементи вирішення головоломок, які вимагають логічного мислення та навичок вирішення проблем.

Гравець повинен мати можливість купувати та керувати предметами, спорядженням та здібностями для персонажа.

Ігровий застосунок повинен включати неігрових персонажів (NPC), з якими гравець може взаємодіяти, включаючи діалогові можливості.

У грі має бути система рівнів, яка дозволяє гравцеві покращувати здібності та навички персонажа.

Користувацький інтерфейс

Ігровий застосунок повинен мати візуально привабливий та інтуїтивно зрозумілий користувацький інтерфейс (UI).

Інтерфейс повинен відображати відповідну інформацію, таку як стан здоров'я, інвентар та цілі квесту.

Інтерфейс повинен містити опції для збереження та завантаження прогресу гри.

Ігровий застосунок повинен надавати можливості для налаштування параметрів аудіо, відео та ігрового процесу.

Графіка та звук

Ігровий застосунок повинен мати високоякісну 3D-графіку з деталізованим оточенням та моделями персонажів.

Ігровий застосунок повинен включати візуальні ефекти для покращення ігрового процесу.

Ігровий застосунок повинен мати різноманітні звукові ефекти та фонову музику, які доповнюють ігровий процес та покращують занурення.

Ігровий застосунок повинен надавати можливість самостійно регулювати гучність звукових елементів.

Платформи та сумісність

Ігровий застосунок розробляється для платформ Windows, macOS, iOS та Android.

Ігровий застосунок повинен бути сумісним з широким спектром апаратних конфігурацій для забезпечення доступності.

Продуктивність та оптимізація

Ігровий застосунок повинен бути оптимізована для забезпечення безперебійної роботи та мінімального часу завантаження.

Ігровий застосунок повинен мати розумний розмір файлу, щоб полегшити встановлення та оновлення.

Ігровий застосунок повинен ефективно використовувати апаратні ресурси для мінімізації системних вимог.

Локалізація

Ігровий застосунок повинен підтримувати декілька мов, щоб задовольнити потреби глобальної аудиторії.

Ігровий застосунок повинен надавати можливість вибору різних мов у користувацькому інтерфейсі.

Нефункціональні вимоги

Ігровий застосунок повинен забезпечувати захопливий та цікавий досвід для гравців.

Ігровий застосунок повинен мати чуйний та плавний ігровий процес.

Ігровий застосунок повинен бути стабільною і не містити серйозних помилок та збоїв.

Ігровий застосунок повинен мати зручний та інтуїтивно зрозумілий інтерфейс для легкої навігації та взаємодії.

Ігровий застосунок повинен мати переконливий і добре написаний сюжет, щоб захопити гравців.

Ігровий застосунок повинен забезпечувати регулярні оновлення та підтримку для усунення помилок та впровадження нових функцій.

Ігровий застосунок повинен відповідати відповідним галузевим стандартам та законодавчим вимогам.

Обмеження

Розробка ігрового застосунку має відбуватися в рамках виділеного бюджету та графіку.

Висновки до розділу 1

За результатами проведеного аналізу предметної сфери розробки комп'ютерних ігор було досліджено та проаналізовано основні етапи процесу розробки гри, класифікацію ігрових застосунків та вплив технологічного розвитку на цю галузь. Було встановлено, що розробка комп'ютерних ігор є складним та багатоетапним процесом, що включає в себе такі етапи, як проєктування гри, розробку інтерфейсу та графіки, програмування та тестування. Крім того, було визначено, що ігрові застосунки можна класифікувати за різними ознаками, такими як жанр, тип гри, платформа тощо.

2 ІНСТРУМЕНТАЛЬНІ ЗАСОБИ РОЗРОБКИ КОМП'ЮТЕРНОЇ ГРИ

2.1 Рушій для розробки ігор Unreal Engine 5

Unreal Engine 5 (UE5) - це остання версія ігрового рушія, розробленого компанією Epic Games. UE5 був анонсований у травні 2020 року й офіційно випущений у травні 2021 року. Цей рушій призначений для створення якісних та інтерактивних відеоігор, зокрема для використання у віртуальній і доповненій реальності [4].

UE5 являє собою потужний інструмент для розробки ігор. Він надає широкий набір інструментів і функцій, включно з потужною системою візуалізації, фізичною симуляцією і системою роботи зі світлом. Однією з особливостей UE5 є його новий рушій для рендерингу Nanite. Він дає змогу відображати графіку високого рівня деталізації, використовуючи технологію мікрополігонів. Ця технологія дає змогу розробникам створювати більші та детальніші сцени без втрати продуктивності [6].

Ще однією важливою функцією UE5 є система віртуальної камери, яка дає змогу розробникам керувати камерою в реальному часі, створюючи нові ракурси та кути огляду. Це дає змогу домогтися більш високого ступеня інтерактивності та підвищити залученість гравця [5].

UE5 також надає інструменти для роботи з аудіо, включно з можливістю створення тривимірних звукових ефектів і міксування звуку в реальному часі. Це дає змогу створювати реалістичніше звукове середовище і підвищити імерсивність гри.

Однією з головних особливостей UE5 є його здатність працювати з великими обсягами даних. Наприклад, система Lumen дає змогу створювати динамічні та реалістичні світлові ефекти у великих сценах без необхідності створення попередньо обчислених освітлень. Це спрощує процес створення ігрових сцен і підвищує ефективність роботи [1].

UE5 також надає потужні інструменти для роботи зі штучним інтелектом, включно із системою навігації, що дає змогу персонажам у грі

переміщатися по сцені та взаємодіяти з об'єктами. Крім того, UE5 також підтримує інструменти для створення анімації, які дають змогу розробникам створювати більш реалістичні рухи персонажів і об'єктів.

UE5 також надає підтримку для різних платформ, включно з ПК, консолями, мобільними пристроями, а також віртуальною та доповненою реальністю. Це дає змогу розробникам створювати ігри, які можуть працювати на різних пристроях і платформах.

Для роботи з UE5 розробники можуть використовувати кілька мов програмування, включно з C++, Python і Blueprint. Blueprint - це візуальна мова програмування, яка дає змогу створювати ігрову логіку без необхідності писати код. Це спрощує процес розробки та дає змогу розробникам швидше створювати прототипи та тестувати різні ідеї [7].

UE5 також надає інструменти для роботи в команді, включно з системою контролю версій і можливістю спільної роботи над проєктом. Це дає змогу розробникам легко спільно працювати над проєктом, обмінюватися ідеями та покращувати процес розробки.

Крім того, UE5 має активне співтовариство розробників і велику документацію, що дає змогу швидко знаходити відповіді на запитання і вирішувати проблеми [6].

Іншим важливим нововведенням UE5 є технологія Lumen. Ця технологія відповідає за реалістичне освітлення в грі, використовуючи глобальну ілюмінацію в реальному часі. Lumen дає змогу створювати рівні з динамічним освітленням, яке реагує на рух об'єктів і джерела світла. Це створює більш реалістичну атмосферу в грі та покращує візуальний досвід для гравців.

UE5 також надає потужні інструменти для створення фізичних взаємодій у грі. Розробники можуть створювати різні матеріали та поверхні, які поводяться по-різному під час взаємодії з об'єктами. Наприклад, скляна пляшка може розбитися під час падіння на тверду поверхню, а м'яч може відскочити від стіни під час удару. Це створює більш реалістичний світ у грі та підвищує рівень взаємодії гравця з навколишнім середовищем [8].

UE5 також надає можливість створювати різні ефекти та анімацію в грі. Розробники можуть створювати різні ефекти, такі як вибухи, дим, вогонь і дощ. Крім того, UE5 надає інструменти для створення анімації, які дають змогу створювати більш реалістичні рухи персонажів та об'єктів.

UE5 також надає підтримку для роботи з різними платформами, включно з ПК, консолями та мобільними пристроями. Розробники можуть створювати ігри для різних платформ, використовуючи один і той самий рушій, що спрощує процес розробки та знижує витрати на розробку [5].

Крім того, UE5 надає інструменти для створення багатокористувацьких ігор. Розробники можуть створювати мережеві ігри з підтримкою багатокористувацького режиму гри та взаємодії гравців між собою. UE5 надає підтримку для створення серверів і хостингу ігор, що робить створення багатокористувацьких ігор ще більш доступним і простим.

Unreal Engine 5 - це потужний інструмент для розробки комп'ютерних ігор, який надає розробникам величезний набір інструментів і функцій для створення високоякісних ігор. Нововведення, як-от Nanite, Lumen і багатокористувацький режим гри, роблять UE5 одним із найінноваційніших рушіїв для розроблення ігор на сьогодні. UE5 продовжує розвиватися і поліпшуватися, і ми можемо очікувати ще більшої кількості нових функцій та інструментів у майбутньому [4].

2.2 Система програмування BluePrint

Blueprint - це система візуального програмування, яка входить до складу ігрового рушія Unreal Engine. Вона дає змогу розробникам створювати ігрову логіку та функціональність за допомогою графічного інтерфейсу, замість того, щоб писати код мовою програмування [10].

Blueprint надає інтуїтивно зрозумілий інтерфейс, який дає змогу створювати складні системи та взаємодії в грі, використовуючи готові блоки та вузли, які з'єднуються один з одним. Кожен вузол виконує певну функцію, наприклад, переміщення персонажа або перевірку умови.

На рис. 2.1 наведено приклад Blueprint.

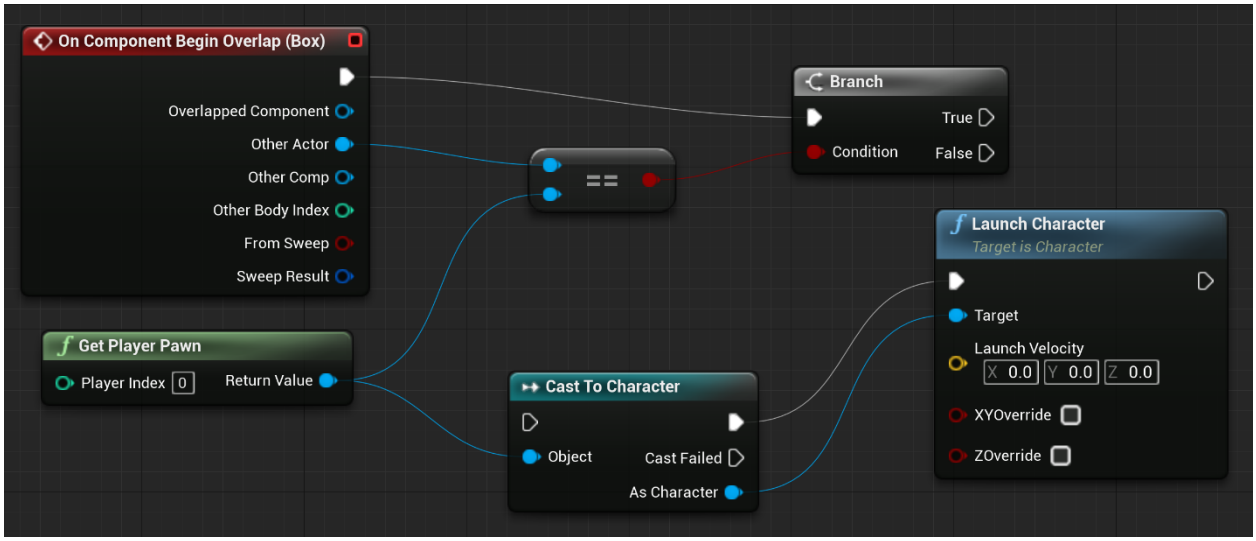

Рисунок 2.1 – Схема Blueprint

Однією з основних переваг Blueprint є можливість швидкого прототипування ігрової логіки. Розробник може швидко створити просту систему і протестувати її в грі, не витрачаючи багато часу на написання коду. Це дає змогу скоротити час розробки та швидше отримати зворотний зв'язок від тестувальників і гравців [4].

Blueprint також має високий ступінь гнучкості та налаштування. Розробник може створювати свої власні вузли та блоки, які відповідають особливостям його гри. Це дає змогу створювати унікальні та складні системи і взаємодії, які неможливо було б реалізувати за допомогою стандартного кодування.

Крім того, Blueprint підтримує використання об'єктно-орієнтованого підходу, що дає змогу розробникам створювати ієрархію класів і успадкування властивостей і методів від батьківських класів. Це дає змогу створювати більш структурований і легко підтримуваний код [9].

Однак, як і будь-який інструмент, Blueprint має свої обмеження. Складні системи можуть стати важкодоступними для розуміння і налагодження, якщо вони не організовані належним чином. Крім того, деякі завдання можуть бути

виконані тільки за допомогою написання коду мовою програмування, що вимагає знання синтаксису і правил програмування.

Blueprints являють собою візуальну систему програмування, яка використовує блоки з кодом, звані вузлами. Користувачі можуть створювати нові вузли і з'єднувати їх один з одним, створюючи таким чином схему логіки гри [3].

У Unreal Engine Blueprint-вузли можуть бути використані для створення поведінки персонажів, визначення цілей і завдань, а також для налаштування ігрових елементів, як-от зброя, рівні, ефекти та інші об'єкти.

Крім того, Blueprint дає змогу користувачам створювати власні компоненти, які можна повторно використовувати в різних проєктах. Це значно спрощує процес розробки та дає змогу зберігати й переносити логіку між різними проєктами.

Blueprint має безліч переваг перед іншими мовами програмування, такими як C++ або Python. Завдяки візуальному інтерфейсу Blueprint дає змогу швидко створювати та змінювати ігрову логіку без необхідності перекомпіляції коду [2].

Також важливо зазначити, що Blueprint в Unreal Engine надає можливість створювати комплексні системи логіки, які можуть взаємодіяти між собою. Наприклад, створюючи логіку для поведінки персонажа, можна використовувати систему камери, яка стежитиме за персонажем, змінюючи свою позицію і кут огляду залежно від дій персонажа.

Крім того, Blueprint підтримує спільну роботу кількох членів команди розробників. Це дає змогу швидко й ефективно обмінюватися ідеями та створювати комплексні системи логіки, що задовольнятимуть вимоги проєкту [10].

Однак, слід зазначити, що Blueprint не може замінити повністю кодування мовою C++. У деяких випадках для створення складної та оптимізованої логіки все

Крім візуального програмування, Blueprint в Unreal Engine також підтримує подієво-орієнтоване програмування. Це означає, що код може бути написаний для реагування на певні події в грі, такі як натискання клавіші або зіткнення об'єктів. Коли подія відбувається, Blueprint викликає пов'язаний із нею вузол, який може виконувати певні дії.

Також Blueprint підтримує успадкування, що дає змогу створювати базові класи зі спільними функціями та властивостями, а потім похідні класи можуть успадковувати ці властивості та доповнювати їх власними. Це спрощує розробку і підтримку коду, оскільки можна уникнути дублювання коду і легко внести зміни до загальних функцій і властивостей, які відображатимуться у всіх класах, що їх успадковують [4].

Ще однією корисною функцією Blueprint є можливість створення користувацьких вузлів, які можна використовувати в різних місцях візуальної схеми. Це дає змогу створювати власні функції та процедури, які можуть бути повторно використані в різних частинах проєкту.

Нарешті, Blueprint забезпечує візуальне відображення взаємодії різних компонентів і об'єктів у грі, що робить налагодження і тестування проєкту більш інтуїтивним і зручним [6].

Загалом, Blueprint в Unreal Engine надає програмістам і дизайнерам можливість створювати складну логіку і функціональність гри без необхідності написання великої кількості коду. Це прискорює процес розроблення і забезпечує більшу гнучкість під час створення та зміни ігрових механік і функцій.

2.3 Поєднання VisualStudio та Unreal Engine в одне інтегроване середовище

Середовище розробки Visual Studio та ігровий рушій Unreal Engine є ключовими інструментами у створенні комп'ютерних ігор. Об'єднання цих двох середовищ в одне інтегроване середовище може значно спростити і прискорити процес розробки ігор [7].

Visual Studio забезпечує можливість розробки застосунків на різних мовах програмування, включаючи C++, C# та інші. Unreal Engine, у свою чергу, є потужним інструментом для створення ігор, який використовує мову програмування C++. Однією з головних переваг інтеграції Visual Studio та Unreal Engine є можливість використовувати всі переваги Visual Studio під час роботи з кодом Unreal Engine.

Поєднання функціональних можливостей Visual Studio та Unreal Engine дає змогу розробникам створювати ігри швидше та ефективніше. Наприклад, Visual Studio дає змогу створювати шаблони коду і генерувати код автоматично, що спрощує створення нових функцій і можливостей гри. Крім того, Visual Studio дає змогу працювати з Unreal Engine API і створювати власні користувацькі класи та функції [9].

Однією з ключових особливостей інтеграції Visual Studio і Unreal Engine є можливість налагодження коду прямо в середовищі Unreal Editor. Це дає змогу швидко знаходити та виправляти помилки, пов'язані із взаємодією коду та ігрових об'єктів. Крім того, Visual Studio дає змогу використовувати потужний відладчик для пошуку помилок у коді, що також істотно спрощує процес розробки [8].

Загалом, Visual Studio являє собою потужне середовище розробки, яке дає змогу розробникам створювати високоякісні застосунки за допомогою різних мов програмування і технологій. Її інтуїтивно зрозумілий інтерфейс і функціональні можливості допомагають підвищити продуктивність і ефективність розробки, що робить її незамінною для багатьох розробників.

Поєднання широких можливостей, інтеграції з іншими інструментами та зручності роботи з Visual Studio роблять її однією з найкращих середовищ розробки на ринку.

2.4 Мова програмування C++

Мову програмування C++ розробив 1983 року Бйорн Страуструп як розширення мови C. Відтоді вона стала однією з найпопулярніших мов програмування, яку використовують для створення широкого спектра застосунків, включно з іграми, операційними системами, драйверами пристроїв та багатьма іншими.

Основні концепції мови програмування C++:

1) Об'єктно-орієнтоване програмування (ООП): мова C++ дає змогу створювати об'єкти, які є екземплярами класів. Класи визначають стан і поведінку об'єктів, а також зв'язки між ними. ООП дає змогу створювати більш модульний, гнучкий і масштабований код [3].

2) Множинне успадкування: на відміну від багатьох інших мов програмування, C++ дає змогу успадковувати властивості та методи від кількох базових класів. Це дає змогу створювати складніші та гнучкіші ієрархії класів.

3) Покажчики та посилання: у мові C++ є можливість використовувати покажчики та посилання на об'єкти. Це дає змогу ефективніше керувати пам'яттю, а також передавати об'єкти між функціями.

4) Шаблони: мова C++ підтримує шаблони, які дають змогу створювати універсальний код, який може бути використаний для різних типів даних. Це спрощує процес розробки та знижує ймовірність помилок [9].

Однією з основних особливостей C++ є те, що вона є мовою зі статичною типізацією. Це означає, що тип кожної змінної в програмі має бути оголошений заздалегідь, і компілятор C++ перевірятиме, чи відповідає тип змінної використанню в програмі. Статична типізація робить C++

безпечнішим і запобігає багатьом помилкам, які можуть виникнути під час роботи з динамічно типізованими мовами [3].

C++ також є мовою з високою продуктивністю. Це пов'язано з тим, що C++ надає програмісту повний контроль над пам'яттю та процесором, що дає змогу створювати швидкі та ефективні програми. C++ також підтримує багатопоточність, що дає змогу створювати багатопотокові програми, які можуть використовувати кілька ядер процесора для прискорення виконання.

Однією з головних причин популярності C++ є його широке використання в різних галузях, включно з розробкою ігор. Багато відомих ігор, як-от World of Warcraft, Minecraft і Grand Theft Auto, були написані на C++. Це пов'язано з тим, що C++ дає змогу створювати високопродуктивний код, який може використовуватися для створення складних ігрових за стосунків [10].

C++ також є мовою з великою стандартною бібліотекою. Стандартна бібліотека C++ містить безліч функцій і класів, які спрощують написання програм. Наприклад, стандартна бібліотека C++ містить контейнери, як-от вектори та списки, а також безліч алгоритмів, які можуть бути використані для обробки даних [7].

Іншою причиною популярності C++ є те, що він підтримує об'єктно-орієнтоване програмування. В об'єктно-орієнтованому програмуванні програму розглядають як сукупність об'єктів, кожен з яких має свої власні дані та методи. C++ надає розробникам можливість створювати класи та об'єкти, що дає їм змогу більш ефективно організовувати свій код і підвищувати його модульність і переносимість.

C++ також є мовою з високою продуктивністю, завдяки використанню статичної типізації та компіляції в машинний код. Статична типізація дає змогу компілятору виявляти помилки на етапі компіляції, що може призвести до більш швидкої та безпомилкової роботи програми. Крім того, компіляція в машинний код дає можливість оптимізувати код для конкретної апаратної платформи, що також може підвищити продуктивність [5].

На закінчення, мова програмування C++ має безліч переваг, які роблять її популярною серед розробників. Вона є мовою з високою продуктивністю, підтримує об'єктно-орієнтоване програмування і має велику екосистему бібліотек та інструментів. Вона також є базовою мовою для розроблення багатьох застосунків, включно з комп'ютерними іграми, системним і вбудованим програмним забезпеченням, науковими обчисленнями та багато іншого.

Висновки до розділу 2

Unreal Engine 5, Visual Studio і мова програмування C++ — це основні інструменти розробки комп'ютерних ігор. Unreal Engine 5 є потужним рушієм для створення ігрових світів і надає безліч інструментів для роботи з графікою, анімацією, фізикою і звуком. Visual Studio являє собою інтегроване середовище розробки для створення застосунків на мові програмування C++, яка широко використовується в ігровій індустрії.

Однією з основних переваг взаємодії Unreal Engine 5, Visual Studio і C++ є можливість створювати високоякісні ігри, які можна запускати на різних платформах, включно з ПК, консолями та мобільними пристроями. За допомогою C++ можна написати оптимізований код, який працює швидко та ефективно, що дає змогу створювати ігри з високою продуктивністю.

3 МОДЕЛЮВАННЯ ТА ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ КОМП'ЮТЕРНОЇ ГРИ

3.1 Опис ідеї та створення плану розробки

Ідея: "Біо-Битва: Крах Генетичної Загрози" - це захоплюючий тактичний шутер, який поєднує в собі елементи науково-фантастичного світу та стратегічного геймплею. Гравець потрапляє у майбутнє, де наука досягла нових висот у розробці квантових технологій [9].

У цьому шутері гравець братиме на себе роль учасника секретного військового загону, який володіє передовими квантовими пристроями та зброєю.

Основа гри передбачає появу персонажа у середовищі земного типу, яке має помірні розміри та технологічну атмосферу майбутнього. Гравець виступає в ролі озброєного кібернетичного солдата, який рухається по локації і захищається від мобів [10].

Було вирішено зробити гру у перегляді від третьої особи, оскільки цей підхід дозволяє гравцям краще сприймати навколишнє середовище, бачити свого персонажа та насолоджуватися естетикою гри. Такий режим також забезпечує більший контроль над рухом, дозволяє маневрувати в просторі та уникати перешкод. Моби переслідують персонажа і наносять йому певну кількість шкоди, яка призводить до завершення гри після чого гравця відправляють у головне меню,звідки гравець має можливість почати нову гру. Щоб захиститися, персонаж користується різними видами зброї з самого початку у гравця на озброєнні буде тільки початкова зброя - пістолет. Зброя, яка буде швидше або завдасть більшої шкоди, можна буде знайти на локації, вона буде абсолютно різною, починаючи від холодної та закінчуючи важкою, зброя завдає шкоди мобам при попаданні в зазначені моделі моба. Якщо вдалося достатню кількість разів вразити мобів, вони загинуть і в залежності від моба випаде певна кількість досвіду, який вплине на рівень персонажа;.

План розробки гри: "Біо-Битва: Крах Генетичної Загрози".

Концепція і дизайн.

Визначення основних механік гри, тактичних елементів та унікальних особливостей.

Геймплей та механіки.

Розробка основних механік гри, включаючи стрільбу, рух, покриття, тактичні розрахунки та систему рівнів [10].

Локація.

Створення локації зі специфічними тактичними ситуаціями та ворогами.

Розробка атмосферного дизайну локації, включаючи архітектуру, освітлення та деталізацію оточення.

Графіка та аудіо.

Створення високоякісних 3D моделей персонажів, зброї, ворогів та оточення.

Розробка текстур, анімацій та спеціальних ефектів для покращення візуального досвіду [8].

Створення атмосферної звукових ефектів.

Штучний інтелект.

Розробка штучного інтелекту для ворожих NPC, здатного до тактичного розуміння та реагування на дії гравця.

Програмування ворогів з різними стратегіями та поведінкою.

Тестування та налагодження.

Проведення внутрішнього тестування гри для виявлення та виправлення помилок, балансування геймплею та покращення ігрового досвіду [5].

Гравець зможе насолодитися наступними особливостями:

1. Екшен та стрільба: Гра пропонує інтенсивні битви та екшен-сцени, де є можливість використовувати різноманітну зброю для подолання ворогів.

2. Тактична стратегія: Гравець повинен ретельно планувати свої дії та використовувати тактичні прийоми, щоб успішно протистояти ворогам. Також гравець зможе приховуватися за перешкодами та створювати стратегічні пастки для ворогів.

3. Еволюція персонажа: Гравець зможе отримати досягнення, за кожен певний рівень персонаж отримає медалі, які будуть указувати на його майстерність [1].

4. Дослідження локації: Гравець буде мати змогу досліджувати локацію, це дає змогу знайти більш кращу зброю або боєзапаси до цієї зброї.

5. Поведінка штучного інтелекту: Штучний інтелект може використовувати різні тактики. Прямувати вперед і використовувати прикриття, втікати від вогню супротивників, застосовувати командну тактику та співпрацювати з іншими штучними інтелектами

6. Двигун гри: Unreal Engine 4 - графіка високої якості, цей двигун має потужні засоби рендерингу, що дозволяють створювати вражаючу візуальну якість. Також він підтримує фотореалістичний рендеринг, динамічне освітлення, воду, тіні та багато інших ефектів [2].

3.2 Розробка ігрових механік

Розробка ігрових механік для гри “Біо-Битва: Крах Генетичної Загрози”, тактичного шутера з елементами науково-фантастичної тематики: У цьому тактичному шутері гравець отримає глибокий геймплей і захоплюючий досвід завдяки різноманітним інноваційним механікам, змога проявити себе як відважного воїна який прекрасно володіє всіма видами зброї і гарною стрільбою або як гравця який підходить до ситуації з розумом та креативністю щоб показати свою унікальну тактичну перевагу. Основні елементи геймплею та механіки забезпечують стратегічний підхід до битви, розвивають навички управління персонажем і викликають почуття насиченості і атмосферності [5].

Перш за все, було створено мапу, яка включає в себе науково-фантастичні елементи. Гравець отримає можливість досліджувати різні точки карти. Це забезпечить можливості для стратегічного руху, пошук додаткової зброї, яка знаходиться на локації, та пошук переваг у битвах.

Другий елемент геймплею та механіки - швидкість та маневреність персонажа. Гравець матиме гнучкість руху, зокрема можливість стрибати та

швидко переміщатися. Це зроблено для ефективного ухилення від ворожих атак, знаходження укриття та уникання пасток. Швидкість дозволить гравцям комфортне переміщення по локації і зменшить час пошуку інших ігрових речей, а завдяки маневрам і взагалі ухилятись від ворожих атак [4].

Третій елемент геймплею та механіки гри - унікальна зброя, вона є важливим елементом геймплею. Гравцю буде доступний арсенал, який надає перевагу у битві. Арсенал має такий перелік зброї як, автомат, пістолет, ніж, пістолет-кулемет, снайперська гвинтівка, гранатомет, ракетниця, патрони та снаряди, які не лише пошкоджують ворогів, а й відштовхують їх. Це дозволяє гравцю вибрати стратегічний підхід до битви, використовуючи певні типи зброї відповідно до ситуації. А механіка стрільби надасть комфортне використання зброї у грі, адже усі кулі попадають точно в ціль і завдають ворогу певною шкоди, в залежності від зброї [6].

Четвертий елемент геймплею та механіки тісно пов'язаний з попередньою темою - система прицілювання. Розроблена таким чином, щоб гравець мав контроль над зброєю. Гравець зможе прицілюватися на велику дистанцію, стріляти з прицілюванням і ефективно використовувати зброю для дального бою, а для більш близького контактного бою підійде ніж, або пістолет-кулемет. Це дозволяє встановлювати прецеденти і створювати власні тактики в битвах, змога відчути себе снайпером або навпаки майстром рукопашного бою.

П’ятий елемент геймплею та механіки гри - система видачі медалі за певний рівень персонажа, це може надати гравцеві додаткову мотивацію для досягнення конкретних цілей і виконання викликів. Медалі можуть бути видані як нагороди за досягнення певного рівня в грі. Ця система може надати гравцеві візуальний показник його прогресу та досягнень у грі. Користувач може досягти найрідкіснішої медалі у грі та відображати її в ігровому інтерфейсі, що створює відчуття задоволення і досягнень [5],

Шостий елемент геймплею та механіки - стратегія і миттєві рішення. Цей елемент стосується способу, яким гравець планує та виконує свої дії у грі,

враховуючи обставини та завдання, що постають перед ним. Стратегія відноситься до довгострокового планування та прийняття рішень у грі. Гравець може розробити стратегію, що включає в себе вибір оптимального шляху, використання ресурсів, встановлення пріоритетів та визначення цілей. Він має можливість аналізувати ситуацію, передбачати наслідки своїх дій та здійснювати обґрунтовані рішення, спрямовані на досягнення бажаного результату. З іншого боку, миттєві рішення відносяться до реакційного та швидкого прийняття рішень у грі відповідно до зміни обставин. Гравець може знаходитися у ситуації, де необхідно швидко вирішити проблему, зреагувати на атаку ворога або здійснити непередбачуваний крок. Миттєві рішення вимагають від гравця швидкого мислення, рефлексів та вміння адаптуватися до зміни обставин. Поєднання стратегії та миттєвих рішень створює різноманітність і виклик у геймплеї. Гравець повинен збалансувати довгострокове планування та реакцію на події в реальному часі, що додає глибину та варіативність до геймплею та дозволяє гравцю відчути себе справжнім стратегом [7].

Останній сьомий елемент, але не менш важливий, штучний інтелект ворогів. У грі створено тип ворога, який має свою власну тактику та поведінку. Ворог може використовувати спритні маневри і об'єднаєтися в команду з іншим штучним інтелектом. Це створює виклик для гравців і змушує їх використовувати свої тактичні навички для подолання різноманітних противників.

Загалом, ці інноваційні механіки додають атмосферу та глибину геймплею тактичного шутера з елементами науково-фантастичної тематики. Гравець отримає можливість відчути себе часткою цього фантастичного світу, досліджувати різноманітні середовища, використовувати унікальну зброю та вести захоплюючи бої проти запрограмованих інтелектуальних ворогів [10].

3.3 Опис ігрового циклу

Опис ігрового циклу. Гравцю цього тактичного шутера надається можливість відчути захоплюючий досвід від гри <<Біо-Битва: Крах Генетичної Загроз>>. Глибокий геймплей, різноманітні інноваційні механіки та багато іншого [5].

Гра «Біо-Битва: Крах Генетичної Загроз» пропонує гравцям захоплюючий ігровий цикл, що включає в себе наступні етапи:

1) Вибір режиму: Гравець має можливість обрати режим гри, залежно від режиму гри буде залежати і сенс.

2) Головна місія на гру: Гравець стикається з генетично зміненими створіннями, які загрожують світовому порядку. Це може відбуватися у вигляді масштабної інфекції або натовпу мутованих монстрів, що нападають на міста або лабораторії. Перед гравцем з'явиться мета врятувати світ від повного занурення в цей хаос.

3) Експлорація території: Гравець досліджує локацію з метою знайти ворога з якого почалась війна в глобальному масштабі, ліквідувати генетичну загрозу.

4) Битва з ворогами: Гравець зіткнеться зі змутованими істотами, розробленими генетичними інженерами. Він використовує свої навички та зброю, щоб перемогти ворогів у бойових сценах, що можуть включати швидкі перестрілки, тактичні бої або майстерне використання зброї.

5) Збір додаткової зброї і система рівнів: Під час експлорації гравець збирає нові види зброї, і отримує досвід - рівень. Що дозволяє гравцю більш комфортно себе почувати і бачити свої результати візуально, для того щоб бути більш ефективним у боротьбі з генетичною загрозою.

6) Розвиток сюжету: Кількість переможених ворогів - розвиток сюжетів, основа сюжету полягає в тому, щоб перемогти всю генетичну загрозу і врятувати світ від жорсткої тиранії.

7) Фінал гри: Гравець долає всі виклики і всіх ворогів на своєму шляху і нарешті призводять до розв'язання конфлікту, але загроза не вщухла, це був лише маленький крок до звільнення всього світу.

Таким чином, ігровий цикл «Біо-Битва: Крах Генетичної Загроз» пропонує глибокий геймплей, включаючи боротьбу з ворогами, експлорацію, систему рівнів персонажа, саме це забезпечить захоплюючий досвід для гравців [8].

3.4 Проєктування інтерфейсів

Для підтримки ігрового досвіду і ефективної взаємодії з геймплеєм, ігровий інтерфейс має відповідати потребам гравця. Для цієї гри був використаний мінімалістичний стиль мішаний з футуристичним стилем ігрового інтерфейсу. Ігровий інтерфейс спрощений і мінімалістичний. У грі представлені чисті лінії, прості форми, нейтральні кольори і мінімум графічних деталей, але до нього додано футуристичний стиль який, натхненний науково-фантастичними світами і технологіями майбутнього. Зазвичай він включає гладкі лінії, світлові ефекти, неонові відтінки та інтерактивні елементи. В жанрі шутер інтерфейс грає велику роль у забезпеченні комфортного та ефективного геймплею. Інтерфейс включає в себе всі елементи, які гравець бачить на екрані під час гри, такі як інформаційні панелі, ознаки життя і боєзапасу, візуальні ефекти та багато іншого. В даному тактичному шутері інтерфейс був розроблений з урахуванням ключових елементів геймплею [3].

Перш за все, на екрані буде відображено головне меню гри, у ньому гравець зможе перейти у вибір режиму, побачити відображення своїх досяг у вигляді медалі чи рівня. Після натиснення кнопки, яка розроблено для початку гри, гравцю можна вибрати свій режим, від режиму гри залежить її сенс і натиснути кнопку для початку гри [7].

Перед очима гравця з'явиться багато корисної інформації, на яку треба звернути увагу, інформація про стан персонажа, а саме життя та боєзапас. Ця

інформація допомагає гравцеві відстежувати свої можливості і приймати відповідні рішення під час битви.

Для кращого показника реалізму у грі відсутня мапа та путівник, мапу гравець повинен дослідити, проаналізувати та зіставити собі її образ, але розташування різних об'єктів у середовищі будуть виділятися на фоні самої локації, сукупність цих аспектів і ускладнень гри дає змогу гравцеві стратегічно планувати свої дії, розраховувати час та використовувати оточуюче середовище для переваги [10].

Інтерфейс включає в себе швидкий доступ до різних функцій, таких як зміна зброї або користування цією зброєю, поповнення боєзапасу і набуття нового рівня у грі. Це дозволяє гравцеві зручно і швидко виконувати дії відповідно до ситуації і розрахувати свої сили.

Окрім того, інтерфейс має зрозумілу інформаційну структуру та інтуїтивно зрозумілі елементи керування. Чітке відображення показників, іконок та піктограм допомагає гравцеві швидко зорієнтуватися в складній ситуації та приймати рішення на основі отриманої інформації, тобто інформація відображається та організовується таким чином, щоб користувачу було легко зорієнтуватися і зрозуміти, що відбувається. Чітка ієрархія, логічна послідовність та належне структурування ігрового інтерфейсу забезпечують зрозумілість інформації і комфорт користувача [9].

У грі присутні аудіоіндикатори та звукові ефекти є важливою частиною інтерфейсу. Аудіоіндикатори можуть використовуються для показу різних станів та подій у грі через звукові сигнали. Вони мають на меті покращити геймплей та надати гравцю більшу іммерсивність та сприятливіші умови для гри. Використання аудіоіндикаторів та звукових ефектів допомагає покращити взаємодію гравця з грою, забезпечують звуковий вихід та створюють більш виразну і атмосферну гру [4].

Усі ці елементи інтерфейсу співпрацюють, між собою і створюють атмосферу, комфорт і розуміння того що відбувається. Гра забезпечує гравцю зручну навігацію, швидкий доступ до важливої інформації та контроль над

персонажем і його діями. Ігровий інтерфейс допомагає гравцю глибше зануритися в геймплей та насолодитися захоплюючим досвідом тактичного шутера з елементами науково-фантастичної тематики [6].

У грі ми намагалися уникнути проблем з ігровим інтерфейсом, які вплинуть на комфорт і зрозумілість для гравця, шляхом створення інтуїтивно зрозумілого та легкого в управлінні інтерфейсу. У грі великій увазі приділяли вимогам щодо чіткості, доступності та зручності інтерфейсу персонажа, а також забезпеченню зрозумілих інструкцій і пояснень для гравця. Крім того, пройшла праця над оптимізацією швидкості та відгуку ігрового інтерфейсу, щоб забезпечити плавну та безперебійну геймплей-взаємодію. Метою було створити гру, в якій гравець міг би насолоджуватися без зайвих перешкод і зосередитися на самому процесі гри [1].

3.5 Стратегії розробки інтерфейсів ігрових застосунків

Розробка інтерфейсів ігрових застосунків вимагає уваги до деталей, адаптивності до потреб гравця та співпраці між розробниками, дизайнерами та іншими членами команди. Враховуючи текст, що був наданий вище, ось підхід до розробки інтерфейсів ігрових застосунків:

Розуміння контексту геймплею: Важливо ретельно вивчити текст та розуміти основні елементи геймплею, механіки та настрій гри. Це допоможе визначити, яку інформацію необхідно відобразити, які функції потрібно забезпечити та які елементи інтерфейсу будуть найбільш важливими для гравця [5].

Проєктування інформаційної структури: Один із ключових аспектів розробки інтерфейсу - це структурування та представлення інформації. Необхідно ретельно визначити, які показники, іконки, текстові елементи та інші деталі відображатимуться на екрані гри. Це допомагає гравцеві зрозуміти стан персонажа, оточуючий світ та доступні дії.

Інтуїтивно зрозумілі елементи керування: Інтерфейс має бути легким у використанні та інтуїтивно зрозумілим для гравця. Елементи керування, такі

як кнопки, повзунки або жести, повинні бути розташовані в логічних місцях та мати відповідні піктограми або підказки. Використання стандартних конвенцій та орієнтація на розповсюджені ігрові шаблони може полегшити навігацію гравців [2].

Естетика та настрій: Дизайн інтерфейсу має відповідати настрою та атмосфері гри. З врахуванням науково-фантастичної тематики, інтерфейс може мати відповідні космічні або футуристичні елементи. Використання підходящих кольорів, шрифтів та графічних ефектів допомагає створити консистентну та привабливу візуальну презентацію.

3.6 Діаграма варіантів використання

Діаграма прецедентів - в UML, діаграма, на якій зображено відношення між акторами та прецедентами в системі. Діаграма прецедентів показує різні варіанти використання та різні типи користувачів системи і часто супроводжується іншими типами діаграм.

На рисунку 3.1 наведено діаграма варіантів використання.

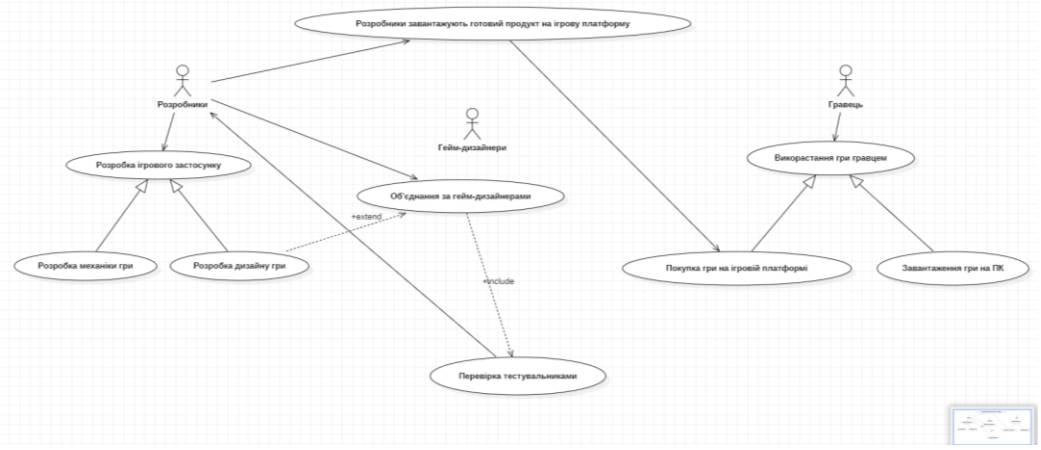

Рисунок 3.1 – Діаграма варіантів використання для розробки ігрового застосунку

Діаграма має в собі 3 actors, такі як розробники, гейм-дизайнери та гравці. Розробник має 3 use cases, один из них головний, два інших дочірні. З дочірнього use case(розробка дизайну гри), йде розширення до кейсу

об’єднання з гейм дизайнерами, після цього через include включення використовується кейс(перевірка тестувальниками на коректність). Після цього actor розробники, виконують кейс(завантаження гри на платформу), де actor гравці, можуть встановити собі гру на ПК, через два дочірніх кейса [7].

Опис дійових осіб

Розробники – починає розробку ігрового застосунку.

Гейм-дизайнери – розробляє графіку ігрового застосунку.

Тестувальники – проводять тестування ігрового застосунку.

Менеджери – роблять та просувають рекламу для ігрового застосунку.

Гравці – використовують вже повністю готовий продукт.

Опис варіантів використання

Варіант використання "Розробка ігрового додатку" є важливою частиною процесу розробки ігор, оскільки він гарантує, що ігровий додаток буде добре продуманим, цікавим і відповідатиме всім необхідним вимогам і специфікаціям. Використовуючи середовище Unreal Engine, розробники можуть створювати високоякісні ігри, які пропонують захоплюючий і приємний ігровий процес для гравців [8].

Цей варіант використання передбачає розробку ігрового додатку з використанням середовища Unreal Engine. Він зосереджений на створенні якісного, цікавого та захоплюючого ігрового процесу для гравців.

Діаграма має в собі 3 actors, такі як розробники, гейм-дизайнери та гравці. Розробник має 3 use cases, один из них головний, два інших дочірні. З дочірнього use case(розробка дизайну гри), йде розширення до кейсу об’єднання з гейм дизайнерами, після цього через include включення використовується кейс(перевірка тестувальниками на коректність). Після цього actor розробники, виконують кейс(завантаження гри на платформу), де actor гравці, можуть встановити собі гру на ПК, через два дочірніх кейса [10].

Опис дійових осіб

Розробники – починає розробку ігрового застосунку.

Гейм-дизайнери – розробляє графіку ігрового застосунку.

Тестувальники – проводять тестування ігрового застосунку.

Менеджери – роблять та просувають рекламу для ігрового застосунку.

Гравці – використовують вже повністю готовий продукт.

Опис варіантів використання

Варіант використання "Розробка ігрового додатку" є важливою частиною процесу розробки ігор, оскільки він гарантує, що ігровий додаток буде добре продуманим, цікавим і відповідатиме всім необхідним вимогам і специфікаціям. Використовуючи середовище Unreal Engine, розробники можуть створювати високоякісні ігри, які пропонують захоплюючий і приємний ігровий процес для гравців [2].

Цей варіант використання передбачає розробку ігрового додатку з використанням середовища Unreal Engine. Він зосереджений на створенні якісного, цікавого та захоплюючого ігрового процесу для гравців [4].

3.6.1 Варіант використання “Розробка ігрового застосунку”

Передумови:

1) Визначено концепцію гри та вимоги до неї.

2) Середовище Unreal Engine налаштоване і готове до використання.

3) Необхідні ресурси, такі як графіка та аудіо, доступні для використання у грі.

4) Створено та схвалено стейкхолдерами документи з дизайну гри.

Таблиця 3.1 – Головний розділ сценарію виконання варіанта використання “Розробка ігрового застосунку”.

Варіант використання	**Розробка ігрового застосунку**
Актори	Розробник гри

Кінець таблиці 3.1

Короткий опис	Описується детальну розробку ігрового застосунку з використанням UE.
Мета	Розробити ігровий застосунок.
Тип	Базовий.

Таблиця 3.2 – Типовий хід подій сценарію виконання варіанта використання “Розробка ігрового застосунку”.

Дії актора	**Відгук системи**
1. Розробник гри придумує ідею для гри, яка сподобається широкій аудиторії та матиме потенціал для отримання прибутку.	3. Запуск: Гра запускається та добре працює механіка гри.
2. Розробники починають писати код для гри.	

Таблиця 3.3 – Винятки сценарію виконання варіанта використання “Розробка ігрового застосунку”.

Дії актори	**Відгук системи**
Виняток №1. Розробник допустили помилку у розробці гри	
	1.Гра відображає помилку.
2. Розробники заново перевіряють код та виправляють помилку.	

3.6.2 Варіант використання ”Розробка дизайну”

Передумови:

1) Визначено концепцію гри та вимоги до неї.

2) Середовище Unreal Engine налаштоване і готове до використання.

3) Необхідні ресурси, такі як графіка та аудіо, доступні для використання у грі.

Таблиця 3.4 – Головний розділ сценарію виконання варіанта використання ”Розробка дизайну”.

Варіант використання	**Розробка дизайну**
Актори	Ігровий дизайнер/розробник.
Короткий опис	Почати проєктування та розробку ігрового додатку з використанням середовища Unreal Engine.
Варіант використання	Розробка дизайну.
Мета	Розробити графіку гри за ідеєю розробника.
Тип	Підлеглий.

Таблиця 3.5 – Типовий хід подій сценарію виконання варіанта використання ”Розробка дизайну”.

Дії актора	**Відгук системи**
1.Розробка концепції гри: Геймдизайнер/розробник разом зі стейкхолдерами визначає концепцію гри, включаючи жанр, механіку геймплею та візуальний дизайн.	4.Прийшло повідомлення, про завантаження графічних ресурсів.
2.Створення дизайн-документів: Ігровий дизайнер/розробник створює дизайн-документи, такі як розкадровки, дизайн персонажів та макети рівнів, щоб окреслити загальний дизайн гри.	

Кінець таблиці 3.5

3. Завантаження графічних ресурсів у гру.	

Таблиця 3.6 – Винятки сценарію виконання варіанта використання ”Розробка дизайну”.

Дії актора	**Відгук системи**
Виняток №2. Дизайнери не правильно завантажили графічні ресурси у гру.	
2. Перезавантаження всіх ресурсів.	1. Повідомлення про помилку.

3.7 Діаграма станів

Діаграма станів - діаграма, що визначає зміну станів об'єкту у часі, одна з діаграм моделювання поведінки в UML. Подає об'єкт як автомат з теорії автоматів зі стандартизованими умовними позначеннями.

На рисунку 3.2 наведено діаграму стану головного меню.

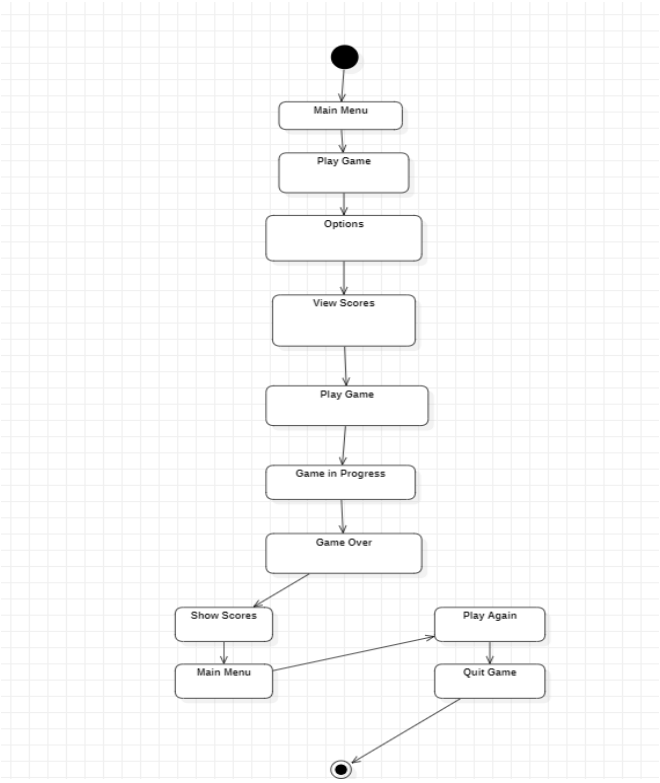

Рисунок 3.2 – Діаграма стану головного меню

На діаграмі описується головне меню, діаграма виконана послідовно, має старт та фінал. Має 11 Simple State які послідовно з’єднанні друг за другом.

На рисунку 3.3 наведено діаграму standing.

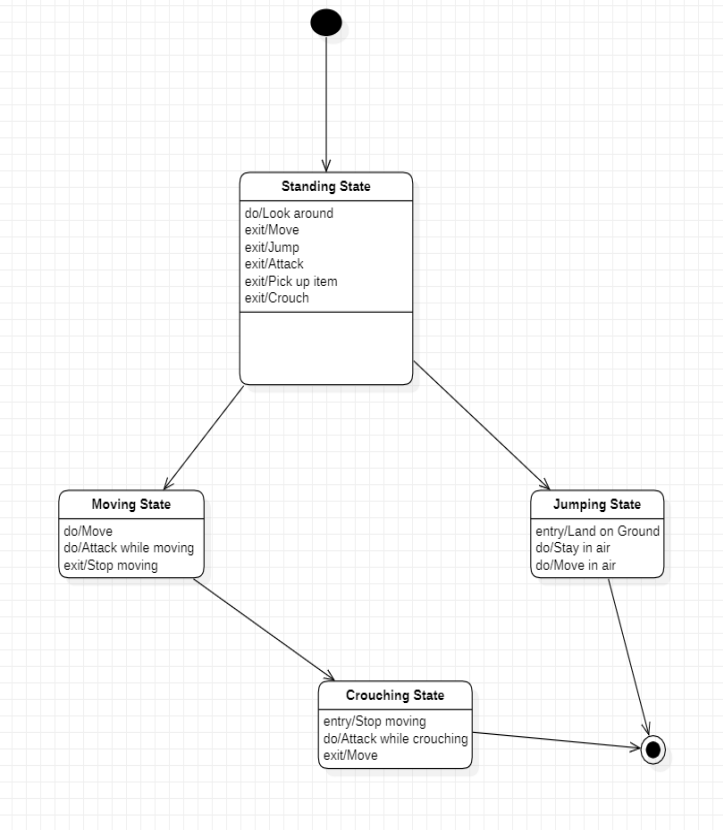

Рисунок 3.3 – Діаграма стану Standing

На діаграмі описується Standing State, діаграма має 4 State, у standing є 6 активностей, один do та 5 exit, standing переходе до 2 інших які називаються Moving та Jumping. Moving має один do та два exit, Jumping має один entry та два do. З State Moving переходе до Crouching State, який має 3 активності(entry, do, exit) та финал йде як від Crouching State так, і Jumping State.

На рисунку 3.4 наведено діаграма стану patrol.

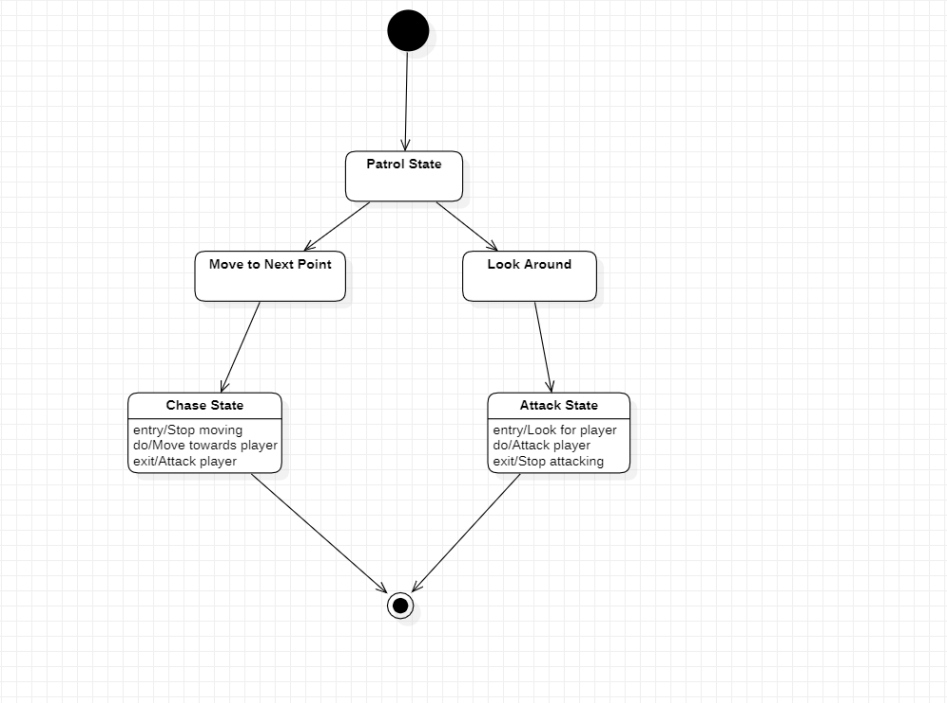

Рисунок 3.4 – Діаграма стану Patrol

На діаграмі описується Patrol State, який переходить до Move to next point та Look around. З Move to next point переходить до Chase State який має 3 активності (entry, do, exit). З Look around переходе до Attack State який має теж 3 активності (entry,do,exit). Та фінал йде як від Chase State, так і Attack State.

Висновки до розділу 3

В третьому розділі було проведено детальний опис ігрового циклу в контексті розробки ігрових застосунків. Дослідження включало аналіз основних етапів, елементів та характеристик ігрового циклу, а також їх взаємодії з гравцем. На основі здобутих знань і вивчення сучасних ігрових трендів були виявлені ключові аспекти, які сприяють успіху імплементації ігрового циклу в за стосунку.

Було ознайомлено з результатами дослідження, які показали, що ефективний ігровий цикл включає такі етапи, як введення гравця в гру, геймплей, прогресію та завершення. Кожен етап має свої унікальні вимоги та впливає на задоволення гравця та його зацікавленість у грі. Крім того,

виявлено, що ігровий цикл може бути додатково збагачений елементами, такими як нарратив, розвиток персонажа, виклики та досягнення.

У процесі дослідження були розглянуті різні підходи до реалізації ігрового циклу, включаючи лінійний, нелінійний та гібридний підходи. Визначено, що оптимальний підхід до розробки ігрового циклу залежить від конкретної гри, її жанру, цілей та аудиторії. Важливо забезпечити збалансованість, різноманітність та виклик в рамках ігрового циклу, щоб зберегти інтерес гравців на тривалому проміжку гри.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ КОМП'ЮТЕРНОЇ ГРИ

4.1 Діаграма класів

Діаграма класів - статичне представлення структури моделі в UML. Відображає статичні (декларативні) елементи, такі як: класи, типи даних, їх зміст та відношення.

На рисунку 4.1 наведено діаграма класів

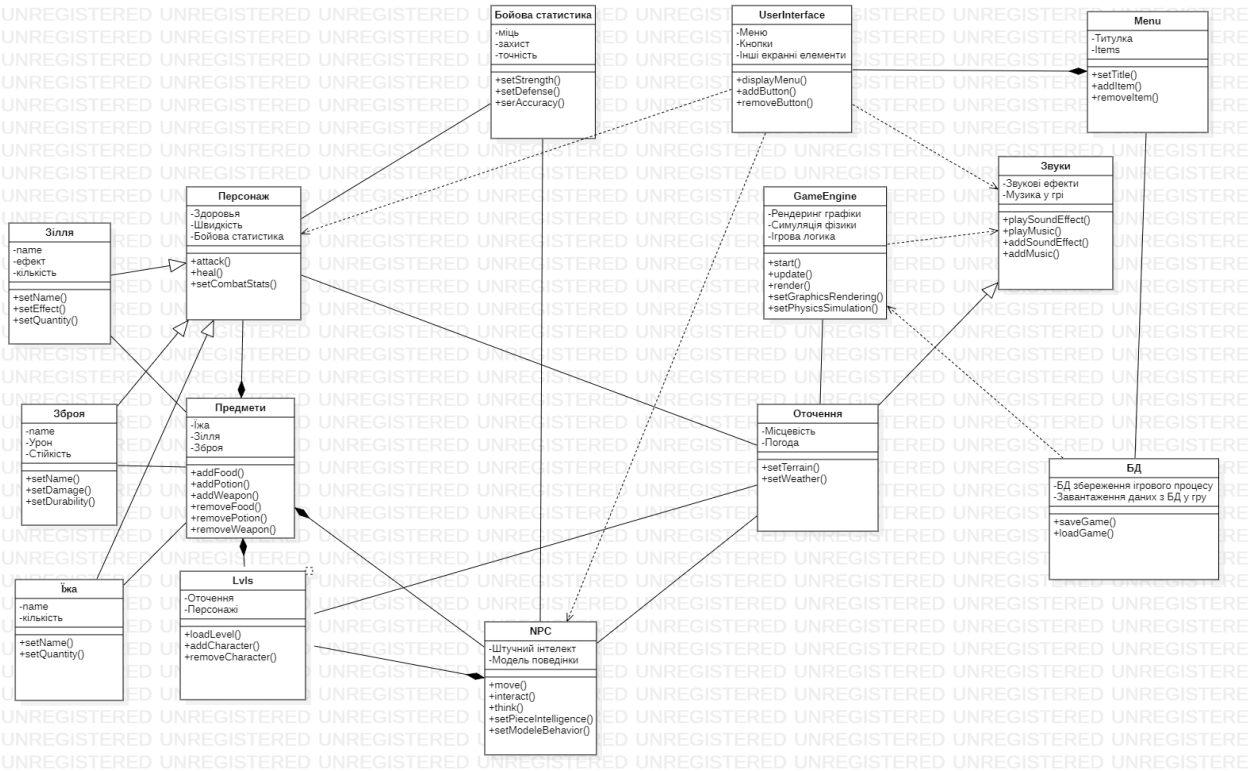

Рисунок 4.1 – Діаграма класів

Клас Персонаж має відношення композиції з класом Предмети, оскільки персонаж гравця може мати у своєму інвентарі різні предмети, такі як їжа, зілля та зброя.

Клас GameEngine має відношення залежності з класом Звуки, оскільки рушій може відтворювати звукові ефекти або музику під час ігрового процесу [8].

Клас UI має відношення композиції з класом Меню, оскільки інтерфейс може містити різні кнопки, на які гравець може натискати.

4.2 Опис функціоналу гри

Для створення головного меню у Unreal Engine 4 було додано “WB_MainMenu”, створений за допомогою Unreal Editor, де є можливість створити користувацький інтерфейс для головного меню за допомогою Blueprint-інтерфейсу.

На рисунку 4.2 наведено процес розробки головного меню.

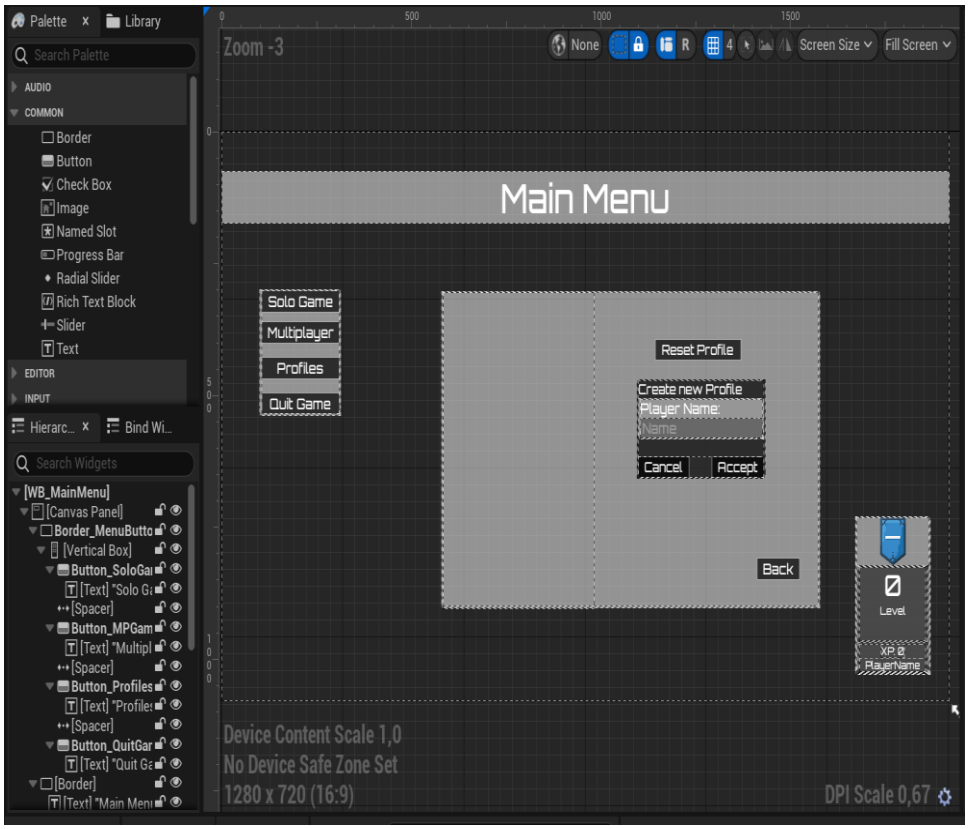

Рисунок 4.2 – Скріншот вікна розробки головного меню

Завдяки цьому класу було додано можливості для створення і розташування елементів інтерфейсу, кнопки, текст, зображення та інші візуальні елементи.

На рисунку 4.3 наведено екран завантаження, як це повинно виглядати у грі.

Рисунок 4.3 – Скріншот готового головного меню

Режим вибору гри було створено по аналогії створення головного меню, замість “WB_MainMenu” створено “WB_GameMode” за допомогою Unreal Editor [10].

На рисунку 4.4 наведено екран завантаження, як це повинно виглядати у грі.

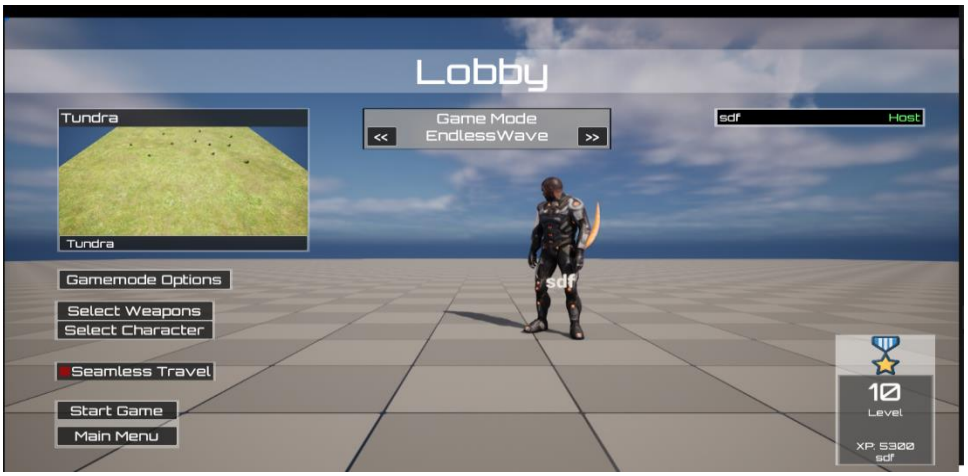

Рисунок 4.4 – Скріншот вибору режиму гри

Для створення екрану завантаження у Unreal engine 4 було додано "WB_LoadScreen", він надає можливість встановити екран завантаження без анімації в Unreal Engine 4. Завдяки цьому не знадобиться робити складні анімаційні ефекти і завантаження гри буде виконуватися швидше [5].

На рисунку 4.5 показано розробку екрану завантаження.

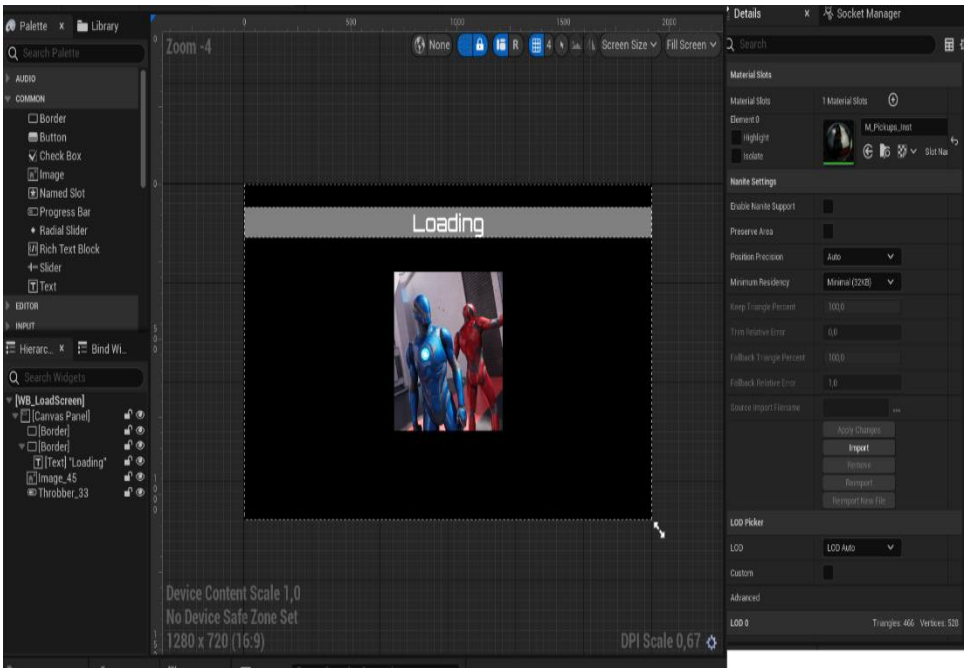

Рисунок 4.5 – Скріншот розробки екрану завантаження

Режим "WB_LoadScreen" дозволяє встановити статичне зображення або логотип як екран завантаження. Це дозволяє використовувати свої власні зображення, щоб відтворити брендований екран завантаження або передати спеціальну повідомлення аудиторії гри [6].

Unreal Engine 4 забезпечує "WB_LoadScreen" оптимізовану швидкість завантаження, що є важливим аспектом для задоволення гравців. За допомогою цих нових можливостей можуть привернути увагу гравців оптимізацією екранами завантаження.

Для створення ігрової місцевості у Unreal engine 4 було додано функцію Landscape Tool та інструмент Level Editor. Landscape Tool дає можливість швидко та зручно створювати рельєф та поверхневі деталі для місцевості. Level Editor дозволяє розміщувати, масштабувати та розташовувати різноманітні об'єкти у грі [5].

На рисунку 4.6 наведено фрагмент розробки ігрової місцевості.

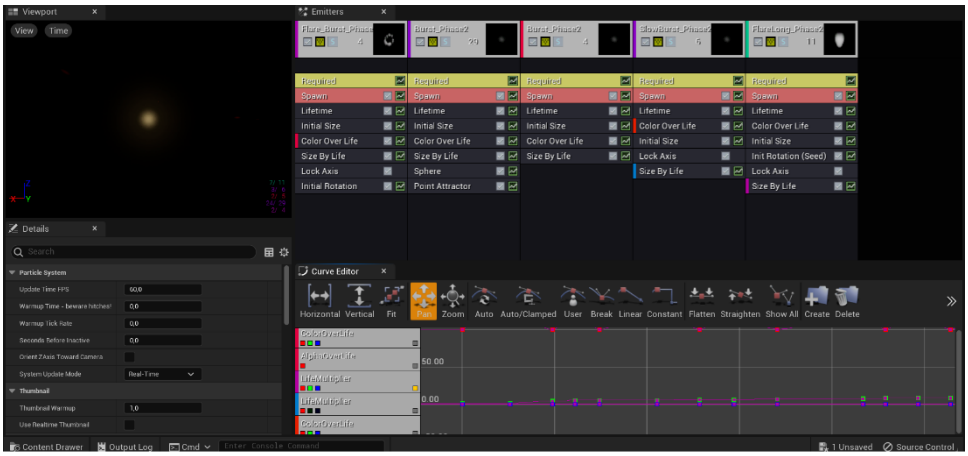

Рисунок 4.6 – Скріншот фрагменту розробки ігрової місцевості

Крім цього, Unreal Engine 4 надає широкі можливості для налаштування освітлення та створення атмосферних ефектів. За допомогою системи динамічного освітлення можна створювати реалістичне освітлення з різними джерелами світла [10].

На рисунку 4.7 наведено ігрову місцевість, як це повинно виглядати у грі.

Рисунок 4.7 – Скріншот ігрової місцевості

Для створення ігрової зброї у Unreal engine 4 було використано готові моделі які пропонував Unreal Engine, після імпорту зброї додано компонент mesh, а саме Static Mesh Component - статичний компонент мешу, та

призначити імпортований меш зброї до об’єкту. Це дозволить відображати модель зброї і подальше використання у грі [5].

На рисунку 4.8 наведено ігрову зброю, як це повинно виглядати у грі.

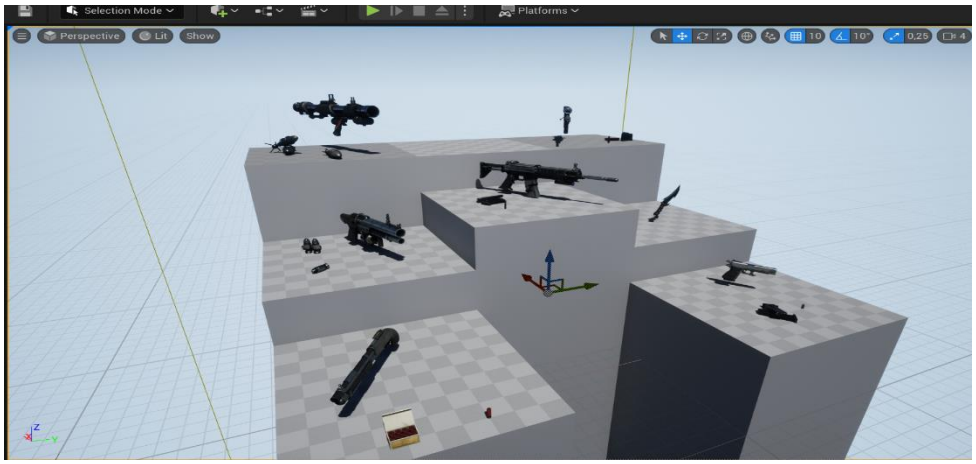

Рисунок 4.8 – Скріншот ігрової зброї

Такий же підхід використовувався і для створення боєзапасу. Використані моделі запропоновані від Unreal Engine, після імпорту моделі боєзапасу додано компонент mesh, та призначений до об'єкта боєзапасу [3].

На рисунку 4.9 наведено екран завантаження, як це повинно виглядати у грі.

Рисунок 4.9 – Скріншот розробки боєзапасу

Поповнення боєзапасу зброї на Unreal Engine відбувається через програмування Blueprint, створення атрибуту боєзапасу, який відповідатиме за кількість патронів або боєприпасів у гравця. Ініціалізація боєзапасу, тобто початкова кількість патронів.

4.3 Механіка пересування персонажа

Моделювання персонажа в Unreal Engine 4 включає кілька кроків, які допомагають створити реалістичну та високоякісну модель персонажа. Основні етапи моделювання персонажа в Unreal Engine 4 включають наступне:

Створення моделі: Перший крок - це створення 3D-моделі персонажа в зовнішньому 3D-редакторі, такому як Blender або 3ds Max. Модель повинна бути створена з урахуванням реалістичних анатомічних пропорцій та деталізації [7].

Розгортка UV-координат: Після створення моделі потрібно виконати процес розгортки UV-координат. Це дозволяє розмістити текстури на поверхні моделі без спотворень. Розгортка UV-координат забезпечує правильне відображення текстур на персонажі [2].

Текстурування: Після розгортки UV-координат можна перейти до створення текстур для персонажа. Це включає створення кольорових текстур (албедо), текстур зіткнень, нормалей, матеріальних карт і т. д. Текстури додають реалізм та деталізацію до моделі.

Риггінг - це процес створення скелетної системи для персонажа, яка дозволяє йому анімуватися. Скелет включає кістки і суглоби, які дозволяють керувати рухом персонажа.

Анімація: Після створення скелетної системи можна створити анімації для персонажа. Анімації можуть бути створені шляхом запису рухів актора, використовуючи захоплення руху або створення анімаційних ключів.

Імпорт в Unreal Engine 4: Останній крок - це імпорт моделі персонажа, текстур та анімацій в Unreal Engine 4. За допомогою інструментів Unreal Engine 4, таких як Persona або Animation Blueprint, розробники можуть налаштувати анімації та взаємодію персонажа з грою [1].

Ці кроки дозволять створити реалістичного персонажа з анімаціями, який буде використаний у грі, створеній з використанням Unreal Engine 4.

На рисунку 4.10 наведено редагування імпортованого персонажа в Unreal Engine 4.

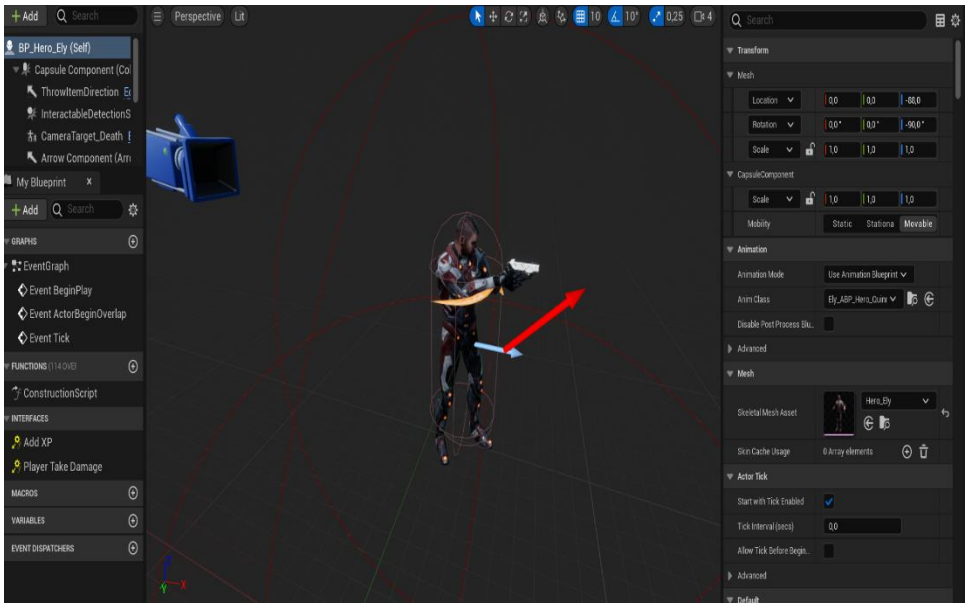

Рисунок 4.10 - Скріншот імпортованого персонажа в Unreal Engine 4

Опис команд з програми Unreal Engine 4.

IA_Move: Ця команда використовується для керування рухом штучного інтелекту (AI). Вона визначає шляхи руху, напрямок та швидкість руху AI-персонажа.

IA_Look: Ця команда встановлює напрямок погляду AI-персонажа. Вона контролює, куди спрямований погляд персонажа, і може бути використана для наведення AI на цілі, виявлення противників або огляду оточуючої обстановки.

IA_ADS: Ця команда включає або виключає режим "прицілювання з прицілом" AI-персонажа. При включеному режимі AI отримує більшу точність при стрільбі.

IA_Crouch: Ця команда змушує AI-персонажа присідати. Вона може використовуватися для приховування AI, зменшення його видимості або отримання переваги в бойових ситуаціях.

IA_Dash: Ця команда виконує швидкий рух AI-персонажа у певному напрямку. Вона може бути використана для ухилення від ворожих атак або швидкого переміщення AI по полі бою.

IA_FireWeapon: Ця команда викликає вогонь зі зброї AI-персонажа. Вона активує вогневу діяльність AI та визначає напрямок і ціль стрільби.

IA_Interact: Ця команда взаємодіє з об'єктами або елементами середовища AI-персонажа. Вона може включати взаємодію з важелями, виконання дій або активацію механізмів.

IA_Jump: Ця команда забезпечує AI-персонажу можливість стрибати. Вона дозволяє AI уникати перешкод або переміщуватися до вище розташованих площин.

IA_Melee: Ця команда викликає ближній бій AI-персонажа. Вона активує атаку AI в непрямому контакті з противником.

IA_PauseGame: Ця команда призупиняє гру. Вона може бути використана для створення паузи в геймплеї або відображення меню налаштувань.

IA_Reload: Ця команда виконує перезарядку зброї AI-персонажа. Вона забезпечує відновлення боєприпасів та підготовку AI до наступного вистрілу.

IA_SwitchShould: Ця команда змінює наплічник AI-персонажа. Вона дозволяє переключатися між правим та лівим наплічником, що впливає на комфортність та точність стрільби.

IA_SwitchWeapon: Ця команда змінює зброю AI-персонажа. Вона дозволяє переключатися між різними типами зброї та амуніції в арсеналі AI.

IA_ThrowGrenade: Ця команда викликає кидок гранати AI-персонажем. Вона активує викидання гранати у вказаному напрямку або на певну відстань.

IA_ToggleInventory: Ця команда відкриває або закриває інвентар AI-персонажа. Вона дозволяє гравцю взаємодіяти з інвентарем, обмінюватися предметами або виконувати інші дії, пов'язані з інвентарем.

IA_ToggleScoreBoard: Ця команда відображає або приховує таблицю результатів гри. Вона дозволяє гравцю переглядати статистику, оцінювати успішність та здійснювати інші дії, пов'язані зі збором інформації про гру.

На рисунку 4.11 наведено розробку команд для пересування.

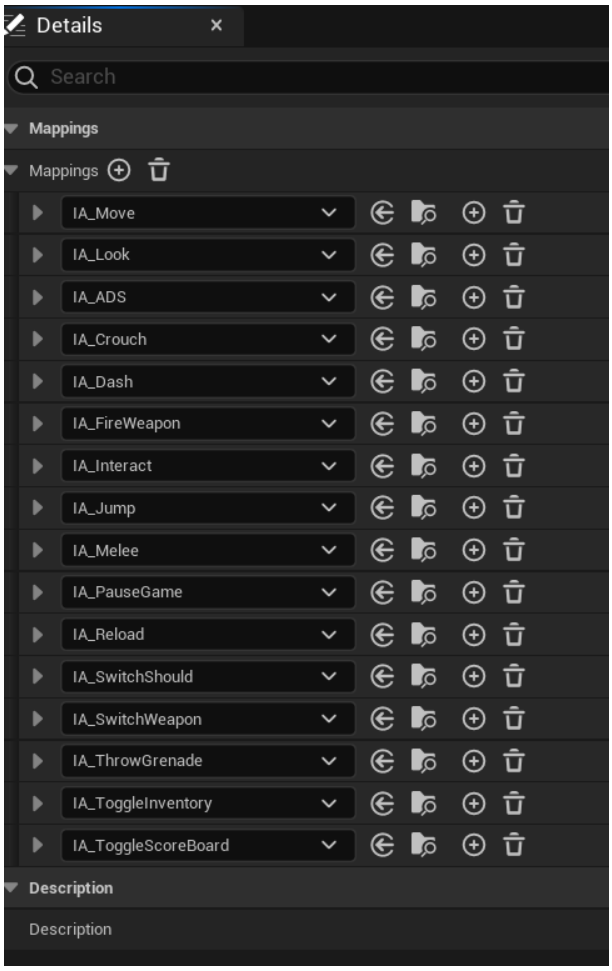

Рисунок 4.11 - Скріншот розробки команд

Ці команди забезпечують гнучкість та контроль над поведінкою AI-персонажа у грі. Використовуючи команди вдалося створити різноманітні реалістичні та більш ускладненні механіки пересування персонажа [10].

На рисунку 4.12 наведено процес розробки пересування персонажа у присядці.

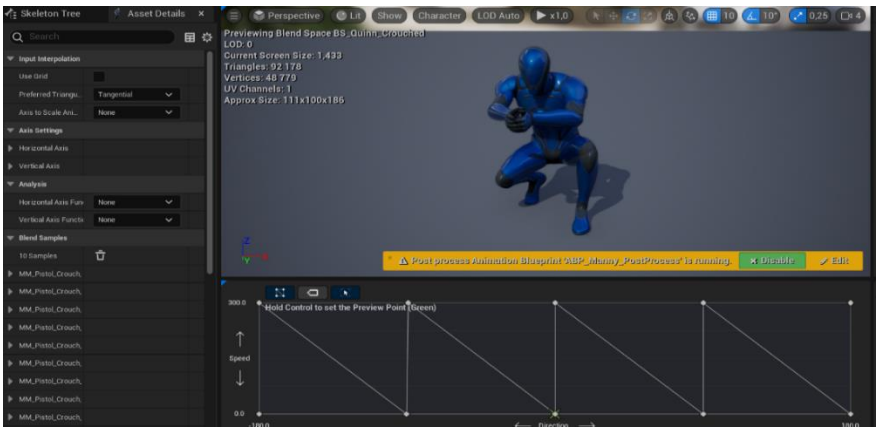

Рисунок 4.12 - Скріншот розробки персонажа у присіду

4.4 Алгоритм пошуку шляху

Для створення пошуку локації гравця у грі на Unreal Engine 4 було використано скрипти Blueprint, а саме Structure. Завдяки цьому гра сама знаходить місце розташування персонажа за координатами вектору і автоматично переміщує його в цю позицію на екрані. Цей механізм полегшив розробку гри, тому що не потрібно власноруч програмувати рух персонажа на кожному кадрі. Замість цього, можна просто вказати нові координати вектору розташування персонажа, і гра сама забезпечить його переміщення

Крім того, такий підхід забезпечує більш плавний рух персонажа, оскільки гра може використовувати різні алгоритми інтерполяції, щоб згладити перехід між позиціями. У грі застосовано лінійну інтерполяцію для прямолінійного руху і квадратичну інтерполяцію для зміни швидкості [9].

Крім переміщення персонажа, цей механізм також застосований для руху камери у грі. В цьому випадку, розробник може задати координати вектору розташування камери і гра автоматично забезпечить їх переміщення.

На рисунку 4.13 наведено фрагмент розробки пошуку локації гравця.

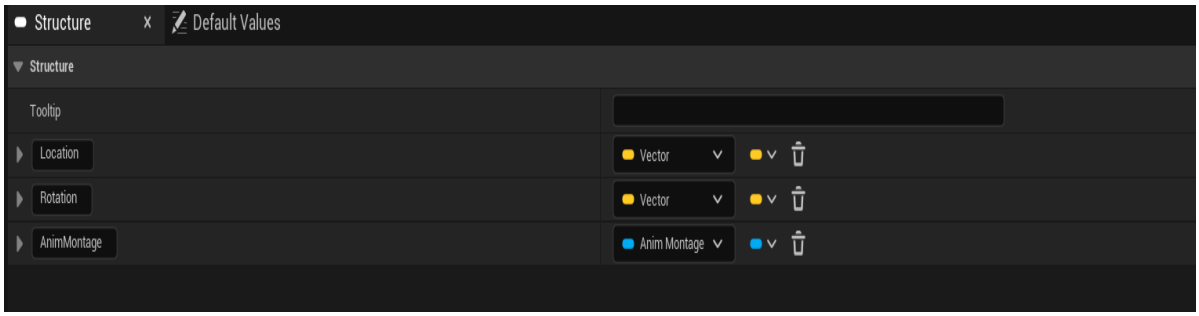

Рисунок 4.13 - Скріншот фрагменту розробки пошуку локації гравця

Завдяки пошуку локації гравця штучний інтелект знає інформацію про місцезнаходження гравця. Завдяки використанню технологій пошуку місцезнаходження гравця, штучний інтелект набуває інформації про векторні координати, що дозволяє йому виявляти положення гравця в грі. Ця механіка є ключовим елементом для оптимізації взаємодії з гравцем і покращення геймплею [8].

На рисунку 4.14 наведено фрагмент розробки напряму ворожого моба.

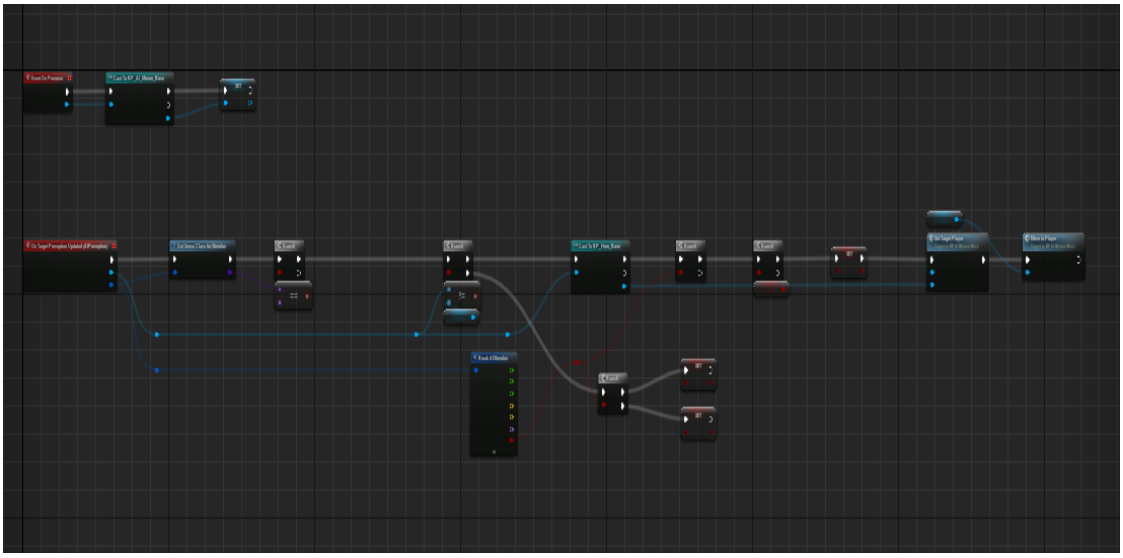

Рисунок 4.14 - Скріншот фрагменту розробки напряму ворожого моба

4.5 Створення мобів

У Unreal Engine 4 можна створювати мобів за допомогою системи Blueprint, яка дозволяє визначати логіку поведінки та налаштувати анімацію у грі.

У грі було створено анімацію моба для атаки. У Blueprint Editor додано анімаційні вузли для атаки лівою та правою рукою. Завдяки AnimNotify було створено AnimNotify_Sword_R_Start вона використана для сповіщення анімації про початок атаки правою рукою з мечем, функцію Set дає змогу налаштувати моба, а саме прив'язка до персонажа та змога завдати шкоди. Принцип з лівою рукою той самий AnimNotify_Sword_L_Start, додається функція Set , яка дозволяю завдати шкоди персонажу і переслідувати його. Наприкінці працює AnimNotify_Sword_R_End, яка завершує атаку правої руки з мечем, для лівої такий самий підхід. Для послідовних атак з чергуванням рук було додано AnimNotify_StartRagdoll, коли вона активується під час відтворення анімації, об'єкт переходить в режим "Ragdoll", і фізична симуляція бере на себе управління об'єктом. Ragdoll в свою чергу відтворює фізичну

поведінку об'єкта, дозволяючи йому взаємодіяти з фізичним середовищем на основі реальних фізичних сил [7].

На рисунку 4.15 наведено процес розробки атаки моба.

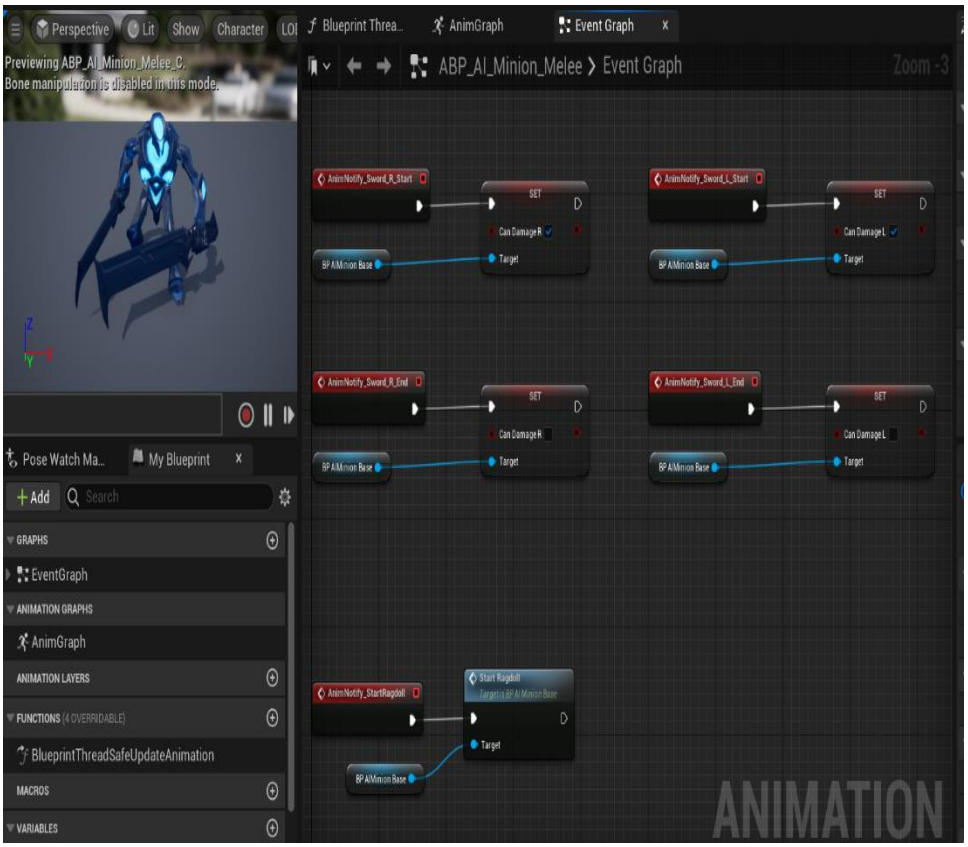

Рисунок 4.15 - Скріншот розробки атаки моба

Для створення анімації були задіяні Blueprint, завдяки ньому і створюються управління об'єктами і інші механіки, Thread Safe відноситься до дизайну або реалізації, яка гарантує безпечне виконання коду декількома потоками одночасно, не викликаючи пошкодження даних або інших проблем, Update animation для розробки анімація відіграє важливу роль у оживленні персонажів, об'єктів та оточення, завдяки цьому і відбувається оновлення анімації, яка передбачає зміну їх властивостей або станів з часом для створення руху або візуальних ефектів. Set це посилання вузол у системі, завдяки якому встановляється значення Speed, яке позначає швидкість відтворення анімації. Функція “TryGetPawnOwner.GetVelocity” використовується для отримання вектору швидкості об'єкта для моба. Vector length XY являє собою величину і напрямок який розглядає тільки компоненти X і Y [10].

На рисунку 4.16 наведено процес розробки анімації моба.

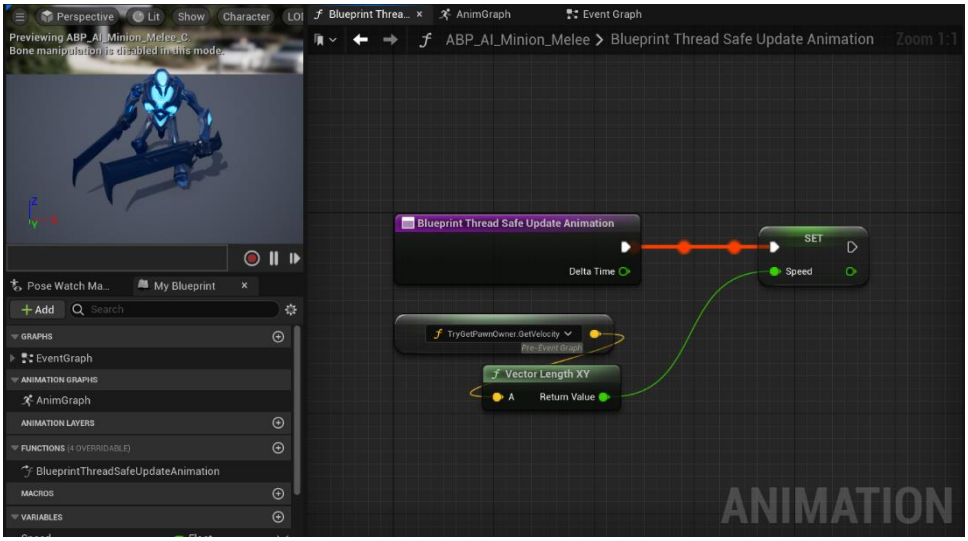

Рисунок 4.16 - Скріншот розробки анімації моба

4.6 Написання механік взаємодії

Health: Ця команда використовується для відстеження та управління рівнем здоров'я персонажа. Вона може бути використана для відображення життєвої панелі, виконання дій при досягненні певного рівня здоров'я та обробки смертельних ударів.

Strength: Ця команда використовується для відстеження та управління рівнем сили персонажа. Вона може впливати на можливості атаки, підйому важких предметів та інші фізичні дії [10].

Stamina: Ця команда використовується для відстеження та управління рівнем витривалості персонажа. Вона може впливати на тривалість та інтенсивність рухів, стрибків та інших фізичних дій, які вимагають енергії.

PistolEquipMontage: Ця команда відтворює анімаційну послідовність для екіпірування пістолета. Вона може включати анімацію виймання пістолета з кобури та підготовку до використання [1].

RifleEquipMontage: Ця команда відтворює анімаційну послідовність для екіпірування гвинтівки. Вона може включати анімацію виймання гвинтівки зі стоячого або лежачого положення та підготовку до використання.

Death_Front, Death_Back, Death_Left, Death_Right: Ці команди відтворюють анімаційні послідовності для смертельних ударів, спрямованих з різних напрямків. Вони можуть включати анімацію падіння та відтворення аудіо-ефектів.

Hit_Front, Hit_Back, Hit_Left, Hit_Right: Ці команди відтворюють анімаційні послідовності для ударів, спрямованих з різних напрямків. Вони можуть включати анімацію реакції на удар та відтворення аудіо-ефектів [3].

CrouchEntry, CrouchExit: Ці команди відтворюють анімаційні послідовності для переходу у присідальне положення та виходу з нього. Вони можуть включати анімацію згинання ніг, зміни висоти персонажа та зміни швидкості руху.

На рисунку 4.17 наведено процес розробки команди взаємодії

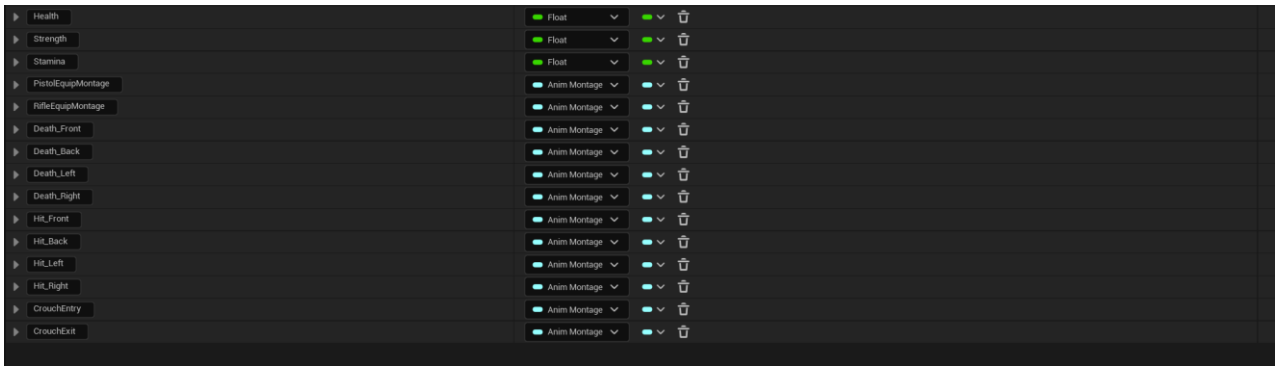

Рисунок 4.17 – Скріншот вікна розробки команд взаємодії

ModelsAndSkins використовується для керування моделями та скинами персонажів у грі. Вона дозволяє змінювати зовнішній вигляд персонажів шляхом використання різних моделей та текстур. Гравці можуть обирати з різних стилів, виглядів та варіантів одягу для своїх персонажів за допомогою цієї команди. Вона дозволяє налаштовувати зовнішній вигляд персонажів таким чином, що кожен гравець може мати унікальний вигляд свого персонажа [2].

UnlockLevelCharacter це система підняття рівня за вбивство ворогів та отримання медалей базується на прогресі гравця у грі. Кожне вбивство ворога приносить гравцеві досвід і підвищує його рівень.

Гравець може отримувати медалі за підняття рівня. Ці медалі можуть бути нагородою за досягнення певних цілей [6].

На рисунку 4.18 наведено процес розробки системи рівнів.

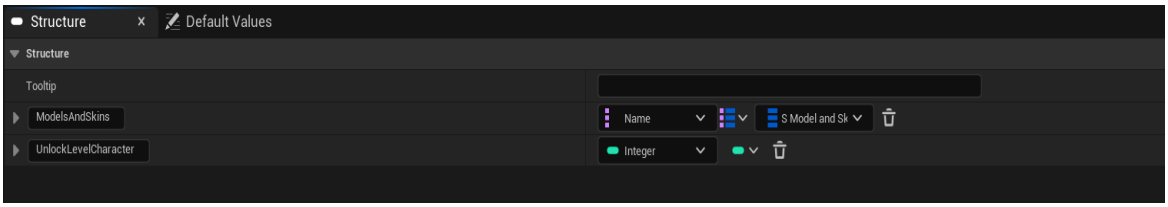

Рисунок 4.18 – Скріншот вікна розробки системи рівнів

Створення стрільби на Unreal Engine 4 базуються на логіці стрільби.

Спочатку створюється вхід для дії стрільби, натискання кнопки "Ліва кнопка миші". Додається Event Tick до Blueprint, потім об'єднують Event Tick з входом "Pressed" - "Ліва кнопка миші".

Далі створюється вихід дії стрільби.

Додаємо в Blueprint компонент "Куля", який буде використовуватися для стрільби. Потім з'єднуємо вихід "Pressed" з кнопкою "Ліва кнопка миші" і входом "Fire" стрільба [5].

Додаємо вхід стрільби.

Event Tick з'єднуємо з Blueprint і додаємо Eevent Tick до входу "Fire"

Створюємо вихід стрільби

Визначаємо шлях польоту снаряду або кулі, встановивши його вектор швидкості. Оброблюємо попадання або зіткнення кулі з ворогам. Додаємо ефекти, звуки, показ вогню з дула і пошкодження [8].

На рисунку 4.19 наведено процес розробки механіки стрільби.

Рисунок 4.19 – Скріншот вікна розробки стрільби

4.7 Інструкція користувача гри

Інструкція користувача комп'ютерної гри "Тактичний екшен науково-фантастика"

Ласкаво просимо до світу науково-фантастичного тактичного екшену! Ця інструкція допоможе ознайомитися з основними функціями та контролем гри, та надасть можливість насолоджуватися епічними битвами та захоплюючими пригодами [9].

Керування персонажем.

Використовуйте клавіші W, A, S, D для переміщення персонажа вперед, вліво, назад та вправо.

Використовуйте мишу для орієнтації та повороту персонажа.

Лівою кнопкою миші здійснюйте атаку на ворогів або виконуйте інтерактивні дії [3].

Використовуйте клавішу пробілу для стрибка або перебування в покритті.

Бойова система.

Прицільтеся на ворогів, тримаючи натиснутою праву кнопку миші.

Використовуйте різну зброю для атаки. Знайдіть оптимальну стратегію та використовуйте тактику, щоб перемогти супротивників.

Звертайте увагу на своє здоров'я. Використовуйте аптечки та інші елементи для відновлення сил.

Взаємодія з оточенням.

Досліджуйте світ гри, взаємодіючі з різними об'єктами та персонажами.

Збирайте ресурси, які можуть бути корисні у вашому проходження гри.

Менеджмент персонажа:

Накопичуйте свій досвід і збирайте нагороду у виді медалі [4].

Заміняйте зброю та змінюйте стиль свого персонажа для комфортної та естетичної гри.

Завдання.

Здолайте усіх ворогів на своєму шляху заради миру на своїй рідній землі! Кожен ворог, з яким ви зіткнетеся, представляє загрозу для безпеки та гармонії вашого світу. Ваша рідна земля зазнає нападів іноземних сил, які намагаються підкорити її [6].

Збереження гри.

Використовуйте автоматичне або ручне збереження гри, щоб зберегти свій прогрес та продовжити грати в подальшому.

Висновки до розділу 4

У четвертому розділі було детально розглянуто програмну реалізацію комп'ютерної гри, аналізуючи різні аспекти та елементи, що входять в процес розробки. Проєктування гри розпочалося зі створення діаграм, які візуалізували структуру та взаємодію компонентів гри.

Опис функціоналу гри включав в себе розробку основних можливостей та особливостей, які гра пропонує гравцеві. Було визначено, які дії та реакції персонажа будуть доступні, які цілі гравця мають бути досягнуті та які виклики та завдання стоятимуть перед ним.

Механіка пересування персонажа виявилася однією з ключових частин програмної реалізації. Були вивчені та використані алгоритми та методи для забезпечення плавності та реалістичності руху персонажа в грі. Це включало в себе обробку вхідних сигналів, розрахунок координат та керування анімацією персонажа.

Окрему увагу було приділено алгоритму пошуку шляху, який дозволяє персонажу ефективно виконувати навігацію у світі гри. Цей алгоритм допомагає виявляти місцезнаходження персонажа і його камери та надавати можливість мобам знаходити гравця.

Також було проведено роботу зі створенням мобів та механік взаємодії. Моби - це рухомі об'єкти в грі, які можуть взаємодіяти з персонажем.

ВИСНОВКИ

Розглянувши предметну сферу розроблення комп'ютерних ігор, було встановлено, що використання Unreal Engine є одним з найпопулярніших та ефективних підходів. Це об'єктивно зумовлено тим, що Unreal Engine надає розробникам широкий набір інструментів для створення високоякісних та інтерактивних ігрових проєктів.

У поєднанні з середовищем розробки Visual Studio та мовою програмування C++, Unreal Engine дозволяє легко створювати та налагоджувати код, що полегшує процес розробки гри. Такий підхід дозволяє розробникам працювати на високому рівні, забезпечуючи швидке та якісне створення проєкту.

Першим етапом було проведення аналізу сучасних автоматизованих ресурсів для створення ігрового застосунку з метою визначення переваг та недоліків платформи. Другим етапом було складання специфікації вимог до ігрового застосунку.

Третім етапом було проєктування та моделювання ігрового застосунку, де було розроблено архітектуру застосунку та визначено деталі реалізації. Четвертий етап включав розробку функціональних модулів ПЗ та проєктування зручного та інтуїтивно зрозумілого інтерфейсу та опцій для користувачів.

У цілому, виконання всіх етапів плану дозволило успішно створити ігровий застосунок, який задовольняє вимоги користувачів та дозволяє їм отримувати задоволення від гри.

Загалом, використання Unreal Engine в поєднанні з Visual Studio та мовою програмування C++ — є одним з найбільш оптимальних шляхів розробки комп'ютерних ігор.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Chris Hargrove. Unreal Engine 4 Game Development in 24 Hours: textbook. Indianapolis: Sams Publishing, 2016. 416p.
2. Satheesh PV. Unreal Engine 4.x By Example: textbook. Birmingham: Packt Publishing, 2016. 490p.
3. Alan Thorn. Mastering Unreal Technology, Volume I: Introduction to Level Design with Unreal Engine 3: textbook. Boston: Cengage Learning, 2009. 384p.
4. Tom Looman. Unreal Engine 4.x Scripting with C++ Cookbook: textbook. Birmingham: Packt Publishing, 2016. 332p.
5. Ryan Shah. Unreal Engine 4 Game Development Essentials: textbook. Birmingham: Packt Publishing, 2016. 198p.
6. John P. Doran, William Sherif. Unreal Engine 4.X By Example: textbook. Birmingham: Packt Publishing, 2016. 458p.
7. B. Burfoot, R. Plant. Unreal Development Kit Game Programming with UnrealScript: Beginner's Guide: textbook. Birmingham: Packt Publishing, 2011. 372p.
8. Wes McDermott. Mastering Unreal Engine 4.X: textbook. Birmingham: Packt Publishing, 2016. 292p.
9. Richard J. Moore. Learning Unreal Engine Game Development: textbook. Birmingham: Packt Publishing, 2015. 304p.
10. Chris Totten. Game Character Creation with Blender and Unity: textbook. Indianapolis: John Wiley & Sons, 2012. 384p.