

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук та технологій
(повне найменування факультету)

Кафедра комп'ютерних систем та мереж
(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

магістра

(ступінь вищої освіти)

на тему КОМП'ЮТЕРНА ГРА В ЖАНРІ 3D-ПЛАТФОРМЕРА
(назва теми)

Виконав: студент 2 курсу, групи КНТ-513м

Спеціальності 123 Комп'ютерна інженерія
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Комп'ютерні системи та мережі

КНИРИК Є.О.
(ПРІЗВИЩЕ та ініціали)

Керівник ТЯГУНОВА М.Ю.
(ПРІЗВИЩЕ та ініціали)

Рецензент ЄФИМЕНКО М.В.
(ПРІЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет Комп'ютерних наук і технологій
Кафедра «Комп'ютерні системи та мережі»
Ступінь вищої освіти магістерський
Спеціальність 123 Комп'ютерна інженерія
(код і найменування)
Освітня програма (спеціалізація) «Комп'ютерні системи та мережі»
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри Кудерметов Р.К.

«_____» _____ 2024 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

КНИРИК Єгор Олександрович

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Комп'ютерна гра в жанрі 3D-платформера

керівник проєкту (роботи) к.т.н., доцент ТЯГУНОВА Марія Юріївна
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від « 18 » жовтня 2024 року № 149

2. Строк подання студентом проєкту (роботи) 10 грудня 2024 року

3. Вихідні дані до проєкту (роботи) Гра The-Ball, що розроблена на ігровому
двигуні Unity 2021.3.15f1, ігровий двигун Unreal Engine 5.4.4, комп'ютер для
тестування з параметрами: процесор Ryzen 5 5500, 16 ГБ оперативної пам'яті,
відеокарта GTX 1050 TI, SSD диск 256 ГБ, операційна система Windows 11 Pro
23H2.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1) Аналіз предметної області;

2) Розробка концепції та основних складових гри;

3) Реалізація гри на базі Unreal Engine 5.4.4;

4) Дослідження впливу ігрового двигуна на апаратні ресурси.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів) _____

Презентація

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4	ТЯГУНОВА М.Ю., к.т.н., доцент		
нормоконтроль	ЩЕРБАК Н.В., ст. викл.		

7. Дата видачі завдання « 28 » жовтня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Робота з джерелами інформації	15.09.2024	
2	Аналіз предметної області	20.09.2024	
3	Розробка концепції гри	27.09.2024	
4	Розробка комп'ютерної гри	27.10.2024	
5	Дослідження впливу ігрового двигуна на апаратні ресурси	05.11.2024	
6	Оформлення отриманих результатів у ПЗ	20.11.2024	
7	Оформлення графічного матеріалу	25.11.2024	
8	Оформлення допоміжного матеріалу	05.12.2024	

Студент(ка)

(підпис)

Єгор КНИРИК

(Ім'я ПРІЗВИЩЕ)

Керівник проєкту (роботи)

(підпис)

Марія ТЯГУНОВА

(Ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

ПЗ: 85 с., 109 рис., 2 табл., 26 джерел.

BLUEPRINT, GODOT ENGINE, REDOT ENGINE, UNITY, UNREAL ENGINE, 3D-ПЛАТФОРМЕР, ІГРОВИЙ ДВИГУН, КОМП'ЮТЕРНА ГРА

Об'єкт дослідження - процес розробки та порівняння продуктивності комп'ютерної гри, створеної на різних ігрових рушіях.

Предмет дослідження – показники продуктивності комп'ютерної гри, розробленої на різних ігрових рушіях, зокрема: кількість кадрів на секунду, навантаження на відеокарту та центральний процесор, а також вплив графічних налаштувань на ці показники.

Мета дослідження – дослідити вплив ігрового двигуна, використаного при розробці гри у жанрі 3D-платформера, на апаратні ресурси комп'ютера.

Пояснювальна записка складається з чотирьох розділів.

У першому розділі проаналізовано предметну область та різні ігрові рушії, обрано програмні засоби для розробки.

У другому розділі описуються загальна концепція та основні складові гри, спроектовано інтерфейс майбутньої гри.

У третьому розділі описується реалізація гри на базі Unreal Engine 5.4.4, а саме: загальні відомості про Unreal Engine 5.4.4 та Blueprint, реалізація інтерфейсу, реалізація ігрових об'єктів та персонажа, реалізація ігрових локацій, збірка проекту, реалізація інсталятора.

У четвертому розділі наведено отримані результати дослідження та їх аналіз, який показав, що менше навантаження на ресурси комп'ютера відбувається грою, яка розроблена на базі Unity 2021.3.15f1.

ABSTRACT

EN: 85 p., 109 images, 2 tables, 26 sources.

BLUEPRINT, GODOT ENGINE, REDOT ENGINE, UNITY, UNREAL ENGINE, 3D-PLATFORMER, GAME ENGINE, COMPUTER GAME

Object of research - the process of developing and comparing the performance of a computer game created on different game engines.

The subject of the study is the performance indicators of a computer game developed on different game engines, in particular: the number of frames per second, the load on the video card and the central processor, as well as the impact of graphic settings on these indicators.

The purpose of the study is to investigate the impact of the game engine used in the development of a 3D platformer game on computer hardware resources.

The explanatory note consists of four sections.

The first section analyses the subject area and various game engines, and selects software tools for development.

The second section describes the general concept and main components of the game, and designs the interface of the future game.

The third section describes the implementation of the game based on Unreal Engine 5.4.4, namely: general information about Unreal Engine 5.4.4 and Blueprint, interface implementation, implementation of game objects and character, implementation of game locations, project assembly, and installer implementation.

The fourth section presents the results of the study and their analysis, which showed that the game developed on the basis of Unity 2021.3.15f1 is less loaded on computer resources.

ЗМІСТ

Скорочення та умовні позначки	8
Вступ.....	9
1 Аналіз предметної області.....	11
1.1 Загальні концепції комп'ютерної гри.....	11
1.2 Класифікація комп'ютерних ігор	13
1.3 Етапи розробки гри	18
1.3.1 Підготовка до виробництва.....	18
1.3.2 Виробництво	19
1.3.3 Випуск	19
1.3.4 Підтримка.....	19
1.4 Аналіз ігрових двигунів.....	20
1.4.1 Ігровий двигун Unity.....	20
1.4.2 Ігровий двигун Unreal Engine.....	20
1.4.3 Ігровий двигун Godot Engine	21
1.4.4 Ігровий двигун Redot Engine	22
1.4.5 Порівняння ігрових рушіїв.....	23
1.5 Вибір мови програмування	23
2 Розробка концепції та основних складових гри.....	25
2.1 Розробка концепції гри	25
2.2 Ігрові механіки.....	25
2.3 Створення прототипу інтерфейсу.....	27
3 Реалізація гри на базі Unreal Engine 5.4.4	30
3.1 Загальні відомості про Unreal Engine 5.4.4 та Blueprint	30
3.2 Реалізація інтерфейсу	33
3.3 Реалізація ігрових об'єктів та персонажа	45
3.4 Реалізація ігрових локацій.....	57

3.5 Збірка проєкту.....	63
3.6 Реалізація інсталятора.....	64
4 Дослідження впливу ігрового двигуна на апаратні ресурси	65
4.1 Загальні концепції	65
4.2 Результати моніторингу ресурсів ПК.....	67
4.3 Аналіз отриманих результатів	70
4.3.1 Вплив на апаратні ресурси при низьких налаштуваннях графіки	70
4.3.2 Вплив на апаратні ресурси при середніх налаштуваннях графіки	74
4.3.3 Вплив на апаратні ресурси при максимальних налаштуваннях графіки ...	78
4.3.4 Аналіз розміру файлів гри на накопичувачі.....	81
Висновки	82
Перелік джерел посилання	83

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧКИ

2D – Двовимірний простір (2-Dimensional)

3D – Тривимірний простір (3-Dimensional)

ОС – Операційна система (OS)

ПЗ – Програмне забезпечення

ПК – Персональний комп'ютер (PC)

AR – Доповнена реальність (Augmented reality)

ММО – Масова багатокористувацька онлайн гра (massively multiplayer online game)

MMORPG – Масова багатокористувацька онлайн рольова гра (massively multiplayer online role-playing game)

UE – Unreal Engine

UI – Інтерфейс користувача (User interface)

VR – Віртуальна реальність (Virtual reality)

ВСТУП

В останні роки розробка комп'ютерних ігор стрімко розвинулася, перетворившись на глобальну індустрію з мільярдними доходами, яка постійно розвивається завдяки технологічним інноваціям.

З моменту появи перших ігор у 1950-х роках, ця сфера пройшла значний шлях - від простих текстових програм до складних, реалістичних віртуальних світів.

Сучасні ігри слугують не лише для розваг, а й активно використовуються для навчання, оскільки сприяють розвитку когнітивних навичок, командної роботи та критичного мислення. Вони також стали ефективним засобом для зняття стресу та проведення дозвілля, що робить їх частиною повсякденного життя мільйонів людей у світі. Комп'ютерні ігри не лише впливають на сучасну культуру, а й відкривають нові можливості для розвитку освіти та соціальної взаємодії.

Сучасні 3D-платформери, такі як Super Mario Odyssey, A Hat in Time та Psychonauts 2, демонструють високу популярність цього жанру. Гравці цінують свободу дослідження, інноваційні механіки та елементи пригод, які такі ігри пропонують. Жанр 3D-платформерів надає широкий простір для творчості завдяки можливості інтеграції унікальних механік, відкритих світів та складних рівнів. Це дозволяє розробникам створювати ігри з високим рівнем взаємодії та естетики.

3D-платформери часто розраховані на широку аудиторію, включаючи сім'ї та дітей. Вони мають яскраву графіку, простий сюжет і доступний геймплей, що робить їх популярними серед гравців різного віку. Сучасні ігрові рушії, забезпечують високоякісну графіку та зручні інструменти для створення 3D-ігор. Це дозволяє навіть невеликим командам створювати проєкти з високим рівнем деталізації. Таким чином, обрана тема є актуальною.

Мета магістерської роботи – дослідити вплив ігрового двигуна, використаного при розробці гри у жанрі 3D-платформера, на апаратні ресурси комп'ютера.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

- проаналізувати предметну область;
- проаналізувати та порівняти ігрові двигуни;
- розробити концепцію гри;
- розробити інтерфейс гри;
- розробити ігрові об'єкти та персонажа;
- розробити ігрові локації;
- провести дослідження впливу ігрового двигуна на апаратні ресурси;
- проаналізувати отримані результати.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Загальні концепції комп'ютерної гри

Комп'ютерна гра (відеогра) - програмне забезпечення, призначене для розваг, відпочинку і навчання, у якому користувач взаємодіє з віртуальним середовищем через різноманітні інтерфейси (клавіатуру, маніпулятор миші, відеокамеру, VR гарнітуру). Розглянемо нижче відомі «історичні» ігри для розуміння розвитку цієї сфери.

OXO 1952 р. - варіація гри в «Хрестики-Нолики», в якій гравець міг змагатися з комп'ютером [1, ст. 23]. Це одна з перших відомих ігор, яка вважається значним етапом у розвитку відеоігор.

Tennis for Two 1958 р. - одна з перших відеоігор, в якості екрану використовувався осцилограф [1, ст. 23]. Гравці керували ракетками, які могли рухатися вгору та вниз, щоб відбивати м'яч через сітку.

Spacewar! 1961 р. - гра була розроблена для комп'ютера PDP-1 [1, ст. 23]. У грі двоє гравців керують космічними кораблями і змагаються в бойових сутичках, намагаючись знищити один одного, при цьому уникати гравітації центральної зірки.

Atari PONG 1972 р. – гравці керують вертикальними «ракетками», які рухаються вгору і вниз по екрану, намагаючись відбивати м'яч (піксель) до ворога. Мета гри - не дати м'ячу вийти за межі екрану і набрати очки.

Pac Man 1979 р. - гравець управляє жовтим круглим персонажем на ім'я Pac-Man, який рухається лабіринтом і з'їдає пункти (крапки). На його шляху є чотири привиди, які намагаються його захопити. Мета гри - зібрати всі крапки на рівні, уникаючи при цьому привидів.

Counter Strike 1999 р. - командний шутер від першої особи. Гравці розподіляються на дві протиборчі команди: терористи та контр-терористи.

Основне завдання - виконати специфічну мету карти, або знищити іншу команду.

У 2000-2010 рр. відбувся розвиток консолей та онлайн-ігор. На початку 2000-х вийшли відомі консолі, такі як PlayStation 2, Xbox і GameCube, які підняли 3D-графіку на новий рівень. PlayStation 2, випущена в 2000 році, стала найпопулярнішою консоллю всіх часів.

World of Warcraft (WoW) – це масова багатокористувацька онлайн рольова гра (MMORPG), випущена 23 листопада 2004 року компанією Blizzard Entertainment.

Вона стала однією з найвпливовіших і найпопулярніших MMORPG усіх часів, встановивши стандарт жанру та створивши величезну базу відданих гравців. Також набули популярності багатокористувацькі ігри, такі як Counter-Strike, Lineage II, які створили міжнародні спільноти.

Хоча ще на початковому рівні, мобільний геймінг почав розвиватися завдяки простим іграм на телефонах Nokia, що підготувало ґрунт для майбутнього розвитку мобільних ігор.

Мобільний геймінг, кіберспорт та віртуальна реальність «2010 – 2020».

Завдяки виходу iPhone і смартфонів на Android мобільний геймінг став масовим і доступним. Ігри на кшталт Angry Birds, Candy Crush Saga і Clash of Clans стали мільйонними бестселерами.

Платформи Steam і пізніше Epic Games Store, дали інді-розробникам можливість виходити на ринок без великих видавців, що призвело до успіхів таких ігор, як Minecraft (2009), Undertale (2015) і Hollow Knight (2017).

Професійні змагання з кіберспорту стали великим бізнесом. Ігри, такі як, Dota 2 і Counter-Strike: Global Offensive, здобули багатомільйонні аудиторії та значні призові фонди.

Технологія VR увійшла в масовий ринок завдяки пристроям Oculus Rift, HTC Vive і PlayStation VR. Хоча ці пристрої ще не стали масовими, вони показали потенціал ігор у віртуальній реальності.

Ігри почали переходити на модель постійних оновлень і внутрішньо-ігрових покупок. Fortnite і Destiny 2 стали прикладами «вічно онлайн» досвіду, орієнтованого на залучення гравців до тривалого користування.

Хмарний геймінг та штучний інтелект «2020 – 2024».

Послуги хмарного геймінгу Google Stadia, Xbox Cloud Gaming і Nvidia GeForce Now, дали можливість грати на різних пристроях без потреби в потужному обладнанні.

Інструменти на основі штучного інтелекту почали використовуватись для вдосконалення діалогів неігрових персонажів (NPC) та автоматизованого створення контенту (текстур, звуків, зображень і т. д.).

Мобільні ігри продовжували зростати і становлять велику частку доходів індустрії.

1.2 Класифікація комп'ютерних ігор

Існує низка параметрів, за якими можна класифікувати комп'ютерні ігри:

- за ігровим жанром;
- за системою монетизації;
- за числом людей, які можуть брати участь у грі;
- за платформою;
- за віковими категоріями.

Ігровий жанр - група ігор, об'єднаних спільними рисами ігрового процесу.

Існує дуже багато ігрових жанрів та під жанрів. Нижче представлені деякі жанри.

Платформер (Platformer) - жанр відеоігор, основною механікою яких є переміщення персонажа по різноманітних платформах, долаючи перешкоди, стрибаючи, ухиляючись і уникаючи пасток.

Жахи (Horror) - жанр відеоігор, метою яких є створення напруженої атмосфери та занурення гравця в сюжет, що лякає або тримає в напрузі.

Файтинг (Fighting) - жанр відеоігор, де гравець бере участь у поєдинках один на один або в командних боях, використовуючи комбінації ударів та спецприйоми.

Виживання (Survival) - жанр відеоігор, метою яких є виживання в небезпечному середовищі, збирання ресурсів, створення укриття та оборона від ворогів або стихій.

Інтерактивне кіно (Interactive Movie) - жанр відеоігор з високим ступенем інтерактивності, в яких гравець робить вибір у ключових моментах сюжету, що визначає розвиток подій.

Ритм-ігри (Rhythm) - жанр відеоігор, де потрібно натискати кнопки або виконувати рухи у такт музиці, слідуючи ритму.

Клікер або інкрементальна гра (clicker game, tap game, idle game) - жанр відеоігор, де гравець, клікає на екран для отримання ресурсів. Поступово гра автоматизується, і гравець може отримувати винагороди без активного управління.

Пісочниці (Sandbox) - жанр відеоігор, де гравець отримує максимальну свободу дій і можливість взаємодіяти з відкритим світом за власними правилами, досліджувати його, будувати об'єкти або змінювати оточення.

Автошахи (Auto Chess / Auto Battlers) - жанр відеоігор, де гравець обирає персонажів, що автоматично б'ються на дошці, схожій на шахову, і займається стратегією та комбінуванням героїв, які підвищують свої навички під час гри.

Навчальні (Educational Games) - жанр відеоігор, які спеціально розроблені для навчання, що охоплюють різні теми та навички, включаючи мову, математику, історію тощо.

Гача (Gacha Game) - жанр відеоігор, у якому ключовий елемент геймплею полягає в отриманні випадкових персонажів, предметів чи іншого ігрового контенту через механіку, схожу на лотерею.

Відеоігри за системою монетизації поділяються на:

- платні (Paid) - людина платить лише один раз за відеогру і отримує весь контент;
- безкоштовні з внутрішньо-ігровими покупками або рекламою (Free-to-play) – відеогра доступна безкоштовно, але основну частку прибутку становлять внутрішні покупки або внутрішньо-ігрова реклама;
- підписка (Subscription-based) – для доступу до гри необхідно щомісяця або року купувати підписку.

За числом людей, які одночасно можуть грати, ігри поділяються на:

- одиночні (Single-player) - призначені для одного гравця;
- багатокористувацькі (Multiplayer) - передбачають участь кількох гравців одночасно (кооперативні або змагальні);
- масові багатокористувацькі онлайн ігри (ММО) - величезні віртуальні світи, де одночасно можуть грати тисячі гравців.

За платформою відеоігри поділяються:

- ПК-ігри - для персональних комп'ютерів;
- VR-ігри – для окуляр віртуальної реальності (Oculus Quest, HTC Vive);
- браузерні ігри – призначені для браузерів;
- консольні ігри - для ігрових консолей (PlayStation, Xbox, Nintendo);
- мобільні ігри - для смартфонів і планшетів;
- ігрові автомати.

Комп'ютерні ігри класифікуються за віковими категоріями за допомогою спеціальних рейтингових систем:

- Pan European Game Information (PEGI);
- Entertainment Software Rating Board (ESRB);
- Unterhaltungssoftware Selbstkontrolle (USK);

- Computer Entertainment Rating Organization (CERO).

Pan European Game Information (PEGI) - європейська система вікових рейтингів, яка використовується для оцінки відеоігор та їхньої відповідності для певних вікових груп [2]. PEGI охоплює більшість країн Європи. Рейтинги включають позначки віку (3, 7, 12, 16, 18) [3]. Рейтинг PEGI зображено на рисунку 1.1.



Рисунок 1.1 – Рейтинг PEGI [4]

Entertainment Software Rating Board (ESRB) - американська система, яка оцінює відеоігри в залежності від їхнього вмісту для різних вікових категорій [5]. Рейтинг ESRB наведено на рисунку 1.2.



Рисунок 1.2 – Рейтинг ESRB [6]

Вікові категорії, що використовуються в системі ESRB:

- E (Everyone) - для всіх вікових категорій. Може містити легке насильство, мінімальні сцени зображення наслідків насильства [6];
- E10+ (Everyone 10 and up) - для дітей віком від 10 років і більше [6];
- T (Teen) - для підлітків від 13 років. Містить насильство середньої інтенсивності, нецензурну лексику [6];
- M (Mature) - для гравців від 17 років. Містить інтенсивне насильство, кров, сексуальний зміст [6];
- AO (Adults Only) - тільки для дорослих (18+). Включає контент із високим рівнем насильства або сексуальними сценами [6].

Unterhaltungssoftware Selbstkontrolle (USK) є німецькою системою класифікації відеоігор, яка регулює вміст ігор відповідно до місцевих законів і культурних норм [7]. Рейтинг USK представлено на рисунку 1.3.



Рисунок 1.3 – Рейтинг USK [8]

Вікові категорії, що використовуються в системі USK:

- USK ab 0 freigegeben – схвалено без вікових обмежень [8];
- USK ab 6 freigegeben – схвалено для дітей віком від 6 років [8];
- USK ab 12 freigegeben – схвалено для дітей віком від 12 років [8];
- USK ab 16 freigegeben – схвалено для дітей віком від 16 років [8];
- USK ab 18 – не схвалено для людей до 18 років [8].

Computer Entertainment Rating Organization (CERO) - це японська система рейтингу відеоігор, яка поділяє ігри на вікові категорії від A до Z [9]. Рейтинг CERO наведено на рисунку 1.4.

Computer Entertainment Rating Organization



Рисунок 1.4 – Рейтинг CERO [10]

Вікові категорії, що використовуються в системі CERO:

- CERO A - для будь-якого віку [9];
- CERO B - для дітей віком від 12 років [9];
- CERO C - для 15-річних і старше [9];
- CERO D - для 17-річних і старше [9];
- CERO Z - лише для 18-річних і старше [9].

1.3 Етапи розробки гри

Розробка комп'ютерних ігор – це добре організований процес, що складається з деяких етапів, які обов'язково проходять при створенні ігор [1, ст. 41]. Розглянемо етапи розробки типової комп'ютерної гри.

1.3.1 Підготовка до виробництва

На цьому етапі команда в формує бачення майбутнього продукту, основну ідею гри, створює дизайн персонажів, обирає необхідні інструменти для реалізації проекту, розробляє перший прототип гри та складає план робіт, який має бути затверджений керівництвом або компанією-видавцем [1, ст. 41].

Зазвичай для розробки ігор шукають конкретного видавця, який готовий інвестувати значні кошти в проект [1, ст. 41]. Коли всі організаційні питання вирішено, розпочинається основний етап виробництва гри [1, ст. 41].

1.3.2 Виробництво

Виробництво гри - це важливий етап, коли розробники починають реалізовувати задуманий план [1, ст. 41]. Однак початкові концепції можуть змінюватися, і часто ці коригування відбуваються на регулярній основі, іноді навіть щодня [1, ст. 41]. Протягом цього етапу, особливо для комерційних проектів, організовуються періодичні огляди прогресу, де команда повинна представити проект, що досягнув певного рівня розвитку [1, ст. 41]. Це можуть бути такі моменти, як готова демоверсія гри або перший рівень [1, ст. 41]. Такі проміжні досягнення часто використовуються для реклами ігор: демоверсії викладаються на спеціалізованих сайтах, і геймери можуть перевірити їх роботу на своєму обладнанні [1, ст. 41].

1.3.3 Випуск

Після завершення розробки, тестування та налаштування гри настає час її релізу [1, ст. 42]. Видавці зазвичай активно стимулюють інтерес до цієї події, оскільки їхня головна мета - отримати максимальний прибуток [1, ст. 42]. Зазвичай, найуспішніші ігри значно перевищують сподівання видавців [1, ст. 42].

1.3.4 Підтримка

Ігри часто виходять з помилками, оскільки розробникам не вистачає часу на ретельне тестування та налагодження [1, ст. 42]. Однак є можливість виправляти ці помилки за допомогою патчів, які встановлюються на вже існуючі версії ігор [1, ст. 42]. Розробники часто випускають гри, що не зовсім готові, а потім регулярно виправляють їх через випуски оновлень [1, ст. 42].

1.4 Аналіз ігрових двигунів

Ігровий двигун або ігровий рушій (Game Engine) – це спеціалізоване програмне забезпечення, призначене переважно для розробки відеоігор, що дозволяє розробникам ефективно створювати гри. Ігрові рушії зазвичай включають супутні бібліотеки та допоміжні програми, які прискорюють процес створення гри, спрощуючи розробнику візуалізацію, фізику, написання сценаріїв, управління звуком та інші завдання. Існує багато сучасних ігрових двигунів [11].

1.4.1 Ігровий двигун Unity

Unity – ігровий рушій для розробки ігор та додатків з широким спектром можливостей і функцій [12]. Він дозволяє створювати проекти різної складності. Незалежно від того, чи ви новачок, чи досвідчений розробник, Unity надає зручні інструменти, які роблять процес розробки ігор простим та ефективним.

Переваги використання Unity:

- підтримка сучасних платформ;
- простий інтерфейс;
- активна спільнота та навчальні ресурси;
- інструменти для розробки VR і AR;
- візуальна мова програмування BOLT .

Недоліки використання Unity:

- проблеми з продуктивністю у великих проєктах;
- важко досягти високого рівня графіки.

1.4.2 Ігровий двигун Unreal Engine

Unreal Engine (UE) – широко поширений ігровий движок компанії Epic Games, який відомий своїм реалістичним рендерингом, можливостями для

створення великих відкритих світів, інструментами для кінематографічних ефектів [13].

Основні переваги Unreal Engine:

- високоякісна графіка;
- велика спільнота;
- багато навчального матеріалу;
- відкритий вихідний код ігрового рушія;
- візуальна мова програмування Blueprint;
- зручні інструменти для розробки;
- розвинута фізика.

Основні недоліки Unreal Engine:

- дуже високі системні вимоги, при використанні всього потенціалу ігрового двигуна;
- важкість навчання;
- складність у використанні.

1.4.3 Ігровий двигун Godot Engine

Godot Engine – один з безкоштовних ігрових рушіїв з відкритим вихідним кодом [14]. Завдяки простоті, гнучкості та універсальності, Godot Engine користується популярністю серед незалежних розробників, а також використовується для навчання та створення прототипів.

Ігровий рушій розповсюджується за ліцензією MIT, тому його можна безкоштовно використовувати навіть в комерційних проєктах [14].

Переваги ігрового рушія Godot Engine:

- безкоштовність та відкритий код;
- сильні 2D можливості;
- проста та гнучка архітектура на основі вузлів;
- підтримка GDScript та інших мов;
- активна спільнота і документація.

Недоліки ігрового рушія Godot Engine:

- менше навчальних ресурсів для специфічних завдань;
- потреба в налаштуванні та оптимізації;
- обмежена кількість сторонніх інструментів та ресурсів.

1.4.4 Ігровий двигун Redot Engine

Redot Engine – це відкритий і гнучкий ігровий рушій, що дає змогу розробляти як 2D, так і 3D ігри з мінімальними зусиллями [15]. Він пропонує інтуїтивно зрозумілий редактор і потужні інструменти для створення та налагодження ігор, підтримуючи безліч платформ

Redot Engine був відокремлений від Godot Engine у вересні 2024 року з метою поліпшити Godot Engine, реалізуючи його потенціал та сприяючи розвитку спільної кодової бази обох проектів через більш ефективну модель, орієнтовану на керування спільнотою [15].

Ігровий рушій є відносно новим і функціонально не дуже відрізняється від Godot Engine на цей час, тому його переваги і недоліки співпадають з Godot Engine окрім спільноти. Спільнота Redot Engine на даний не дуже велика. Логотип ігрового рушія наведено на рисунку 1.5.



Рисунок 1.5 – Логотип Redot Engine [16]

1.4.5 Порівняння ігрових рушіїв

Порівняння ігрових рушіїв наведено в таблиці 1.1.

Таблиця 1.1 - Порівняння ігрових рушіїв

Критерій	Unity	Unreal Engine	Godot Engine	Redot Engine
Складність навчання	Низька	Висока	Середня	Середня
Кросплатформеність	+	+	+	+
Рівень графіки	Середній	Високий	Середній	Середній
Спільнота	Велика	Велика	Середня	Мала
Мова програмування	C#, Bolt	C++, Blueprint	GDScript, C, C#, C++	GDScript, C, C#, C++

Порівнявши характеристики та проаналізувавши ігрові рушії, для розробки гри був обраний Unreal Engine.

1.5 Вибір мови програмування

Blueprint - це система візуального програмування, вбудована в Unreal Engine, яка дозволяє створювати ігрову логіку без написання коду. За допомогою Blueprint розробники можуть реалізувати складні функції і взаємодії шляхом з'єднання вузлів у графічному інтерфейсі [17].

Порівняння мов програмування Unreal Engine наведено в таблиці 1.2.

Таблиця 1.2 - Порівняння мов програмування

Критерій	C++	Blueprint
Швидкість виконання	Висока продуктивність	Повільна, особливо в математичних завданнях
Швидкість написання коду	Повільно	Швидко
Можливість модифікації ігрового рушія	Є	Немає
Складність вивчення	Висока	Низька
Обробка помилок	Більш функціональна обробка помилок	Базові інструменти налагодження,
Застосування	Використовується для складної логіки гри, функцій, що потребують високої продуктивності, оптимізації та управлінні ресурсами	Ідеально підходить для нескладних механік гри, анімації, елементів UI логіки, що не є критичною для продуктивності.

Порівнявши ключові характеристики та проаналізувавши функціональні можливості, було прийнято рішення обрати мову візуального програмування Blueprint.

2 РОЗРОБКА КОНЦЕПЦІЇ ТА ОСНОВНИХ СКЛАДОВИХ ГРИ

2.1 Розробка концепції гри

Жанр гри – 3D-платформер.

Цільова аудиторія гри – діти віком від 5 до 12 років.

Ідея гри – Гравець управляючи ігровим персонажем рухається по платформах від початку ігрової зони до її кінця долаючи різні перешкоди.

Персонаж – ігровий персонаж буде представляти собою футбольний м'яч, яким буде управлять гравець.

Місце подій гри – ігрові дії відбуваються високо в небі на платформах.

Ціль гри – пройти всі рівні.

Платформа для якої розроблюється гра – гра розробляється для ПК.

Підтримка операційних систем – гра розроблюється для ОС Windows 10, та Windows 11.

Складність гри – низька, але в залежності від віку та навичок взаємодії з маніпулятором миші та клавіатури може стати досить складною.

Тривалість проходження гри – повне проходження гри займає приблизно 30 хвилин, проте може бути довшим або швидшим все залежить від стилю проходження гри. Проходження одного рівня займає приблизно 3 хвилини.

2.2 Ігрові механіки

Основними ігровими об'єктами будуть виступати платформи з різними механіками, які наведені нижче.

Механіка падіння платформи. Коли ігровий персонаж доторкнеться до платформи через декілька секунд вона почне падати і потім видалиться з рівня без можливості відновлення.

Механіка руху платформи. Платформа рухається від однієї точки до іншої. Коли платформа досягне однієї з двох точок, то буде очікувати декілька секунд.

Механіка обертання платформ. Платформа обертається на 360 градусів після чого буде очікувати декілька секунд.

Інші механіки ігрових об'єктів наведені далі.

Ящик представлятиме об'єкт, який необхідно буде здвинути за допомогою ігрового персонажа.

Шипи представлятиме нерухомий об'єкт при зіткненні з яким ігровий персонаж знищується після чого ігрова локація перезавантажиться.

Механіка телепорту. Коли ігровий персонаж доторкнеться до телепорту, він переміститься в іншу частину ігрової локації.

Механіки ігрового персонажа наведені нижче.

Механіка руху. Ігровий персонаж може рухатися за допомогою фізики: вперед, назад, ліворуч, праворуч. На рух персонажа також впливає напрямок погляду камери.

Механіка управління камерою. Камера слідує за ігровим персонажем та дозволяє гравцю повертати її для огляду ігрової локації. Також камера має можливість наближення та віддалення, які обмежені деякими значеннями.

Механіка знищення ігрового персонажа. При зіткненні ігрового персонажа з об'єктом шипи, персонаж розсипається на кілька сфер.

Програш може настати в трьох випадках:

- внаслідок падіння персонажа за межі рівня;
- внаслідок доторкання ігрового персонажа до шипів;
- випадкове активування падаючої платформи внаслідок чого втрачається можливість рухатись ігровою локацією.

В перших двох випадках рівень перезавантажиться автоматично. В третьому випадку необхідно за допомогою меню паузи перезавантажити рівень.

2.3 Створення прототипу інтерфейсу

Створення прототипу - це процес розробки попередньої спрощеної моделі або зразка чогось для перевірки ідеї, тестування функцій, демонстрації концепції або дизайну перед майбутнім створенням.

Прототипи можна розділити на прототипи низької чіткості та високої чіткості.

Прототипи низької чіткості - схематичний ескіз або простий макет, який дає загальне уявлення про структуру та взаємодію, не заглиблюючись у деталі.

Прототипи високої чіткості - надає більш детальні моделі, включаючи графічний дизайн, анімацію та інтерактивні елементи, які наближаються до остаточної версії.

Прототипи інтерфейсу гри представлені на рисунках 2.1 – 2.7.

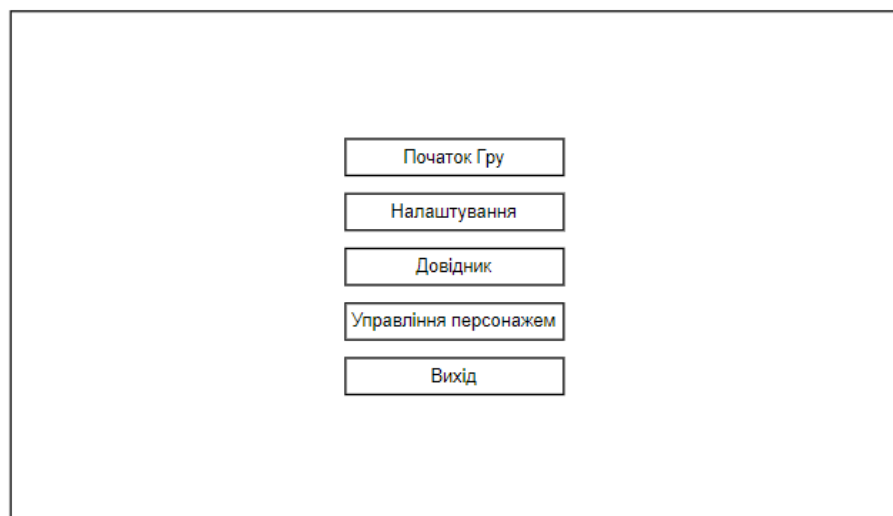


Рисунок 2.1 – Прототип головного меню

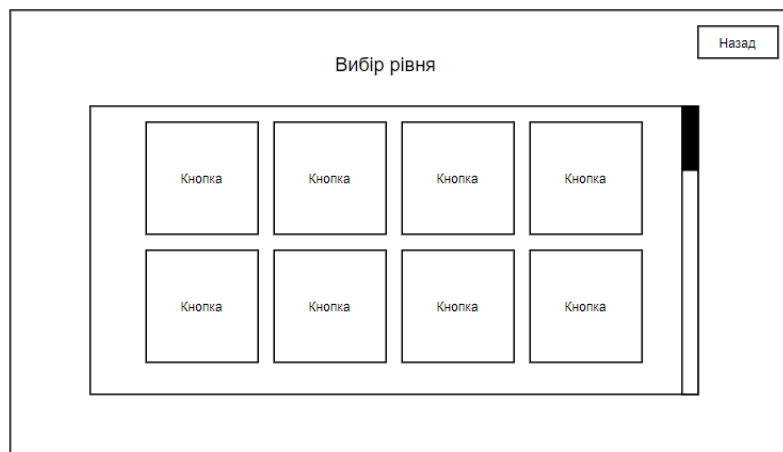


Рисунок 2.2 – Прототип меню вибору ігрових локацій

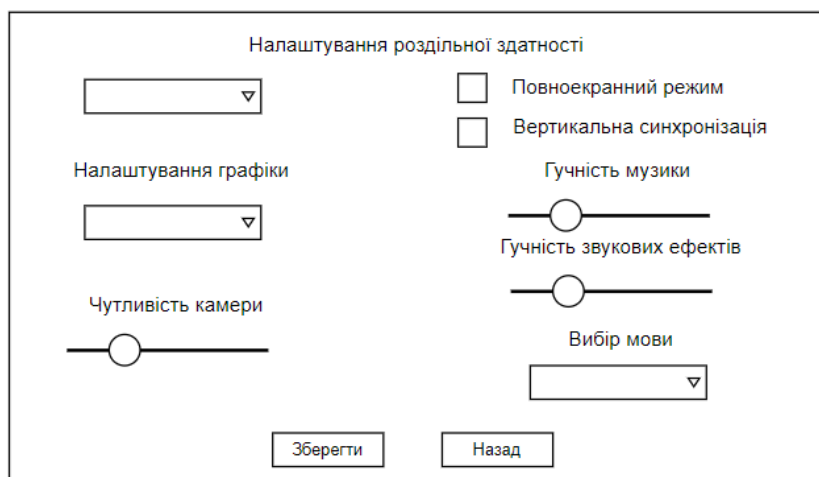


Рисунок 2.3 - Прототип меню налаштувань гри

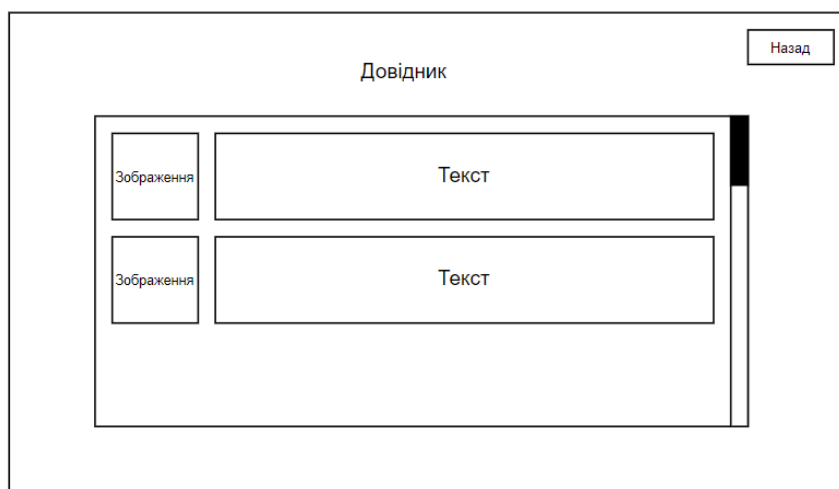


Рисунок 2.4 – Прототип вікна довідника



Рисунок 2.5 – Прототип інформаційного вікна управління персонажем

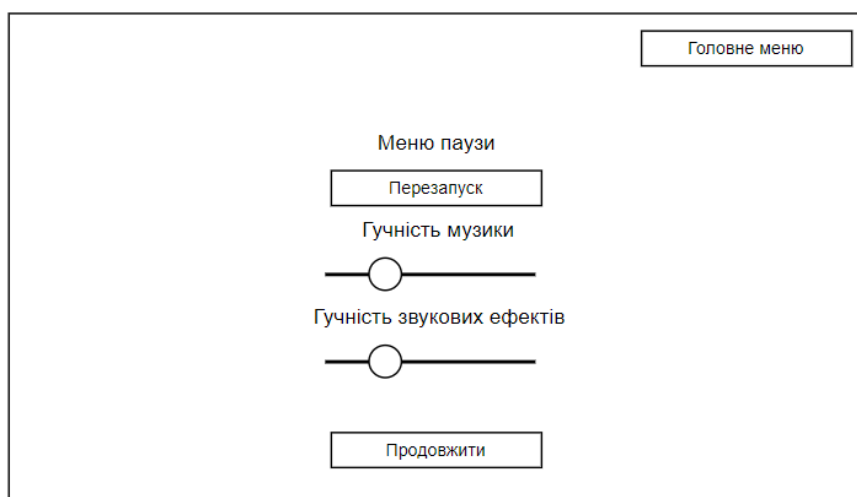


Рисунок 2.6 -Прототип меню паузи

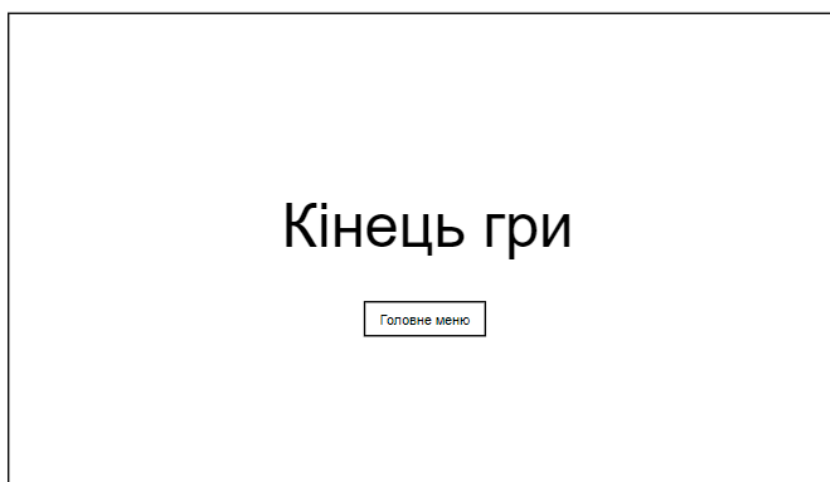


Рисунок 2.7 – Прототип вікна кінця гри

3 РЕАЛІЗАЦІЯ ГРИ НА БАЗІ UNREAL ENGINE 5.4.4

3.1 Загальні відомості про Unreal Engine 5.4.4 та Blueprint

Інтерфейс Unreal Engine 5.4.4 наведений на рисунку 3.1 розділяється на декілька зон:

- рядок меню використовується для доступу до команд та функцій редактора;
- головна панель інструментів містить найпоширеніші інструменти, ярлики для входу в режим відтворення, а також інструменти для збірки гри;
- вікно перегляду рівня відображає візуальне представлення світу гри;
- кнопка відкриття провідника ресурсів з якого можна отримати доступ до всіх ресурсів у проєкті;
- нижня панель інструментів містить ярлики для консолі, журналу виводу, статусу файлів, систем контролю версій;
- список об'єктів відображає всі об'єкти в поточному рівні проєкту;
- панель деталей відображає різні налаштування залежно від того, що виділено у вікні перегляду рівня.

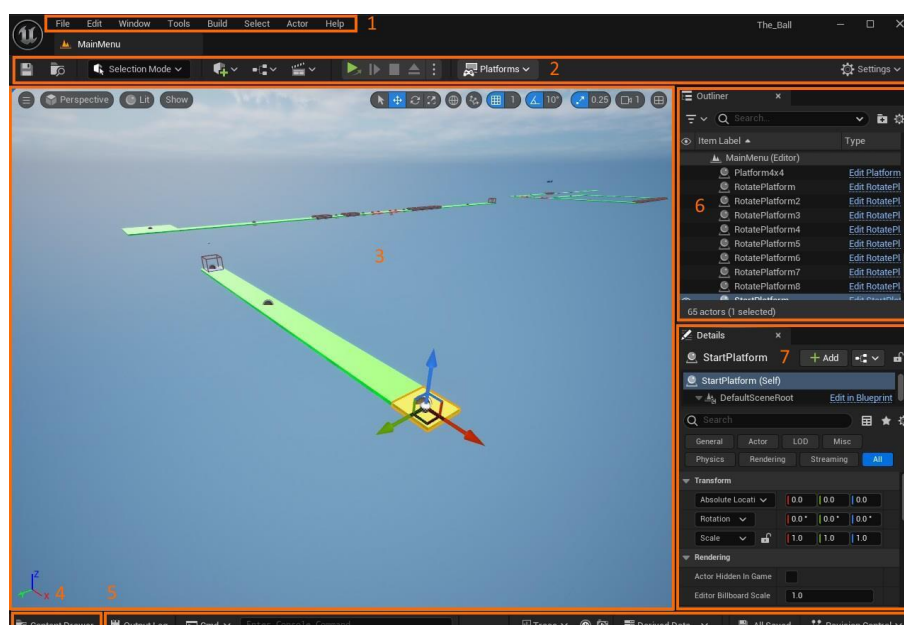


Рисунок 3.1 – Інтерфейс Unreal Engine 5.4.4

Далі наведені терміни пов'язані з розробкою гри на Unreal Engine 5.4.4.

Актор (Actor) – представляє собою будь-які об'єкти, які можуть бути розміщені в рівнях (сценах) гри.

Пішак (Pawn) або маріонетка - підклас актора, що представляє собою об'єкти, які можуть бути керовані гравцем. Pawn служить базою для створення персонажів і інших керованих об'єктів у грі.

Рівень (Level) є фундаментальним об'єктом, рівень можна розглядати як одну сцену, яка містить всі об'єкти ігрового середовища: освітлення, ландшафти, предмети і багато іншого.

Ігровий режим (Game Mode) - визначає різні ігрові умови в межах рівня.

Екземпляр гри (Game Instance) - об'єкт, який створюється під час запуску гри і не знищується, доки гра не буде закрита. Він є одним з базових об'єктів для зберігання даних.

Матеріал (Material) - визначає вигляд поверхні об'єкта в грі. Матеріали відповідають за кольори, прозорість, рельєфність, ефекти поверхні, як світло взаємодіє з матерією та інші візуальні характеристики об'єктів.

Граф подій (Event Graph) - основна область для створення логіки гри в Blueprint.

Blueprint - візуальна мова програмування, яка дозволяє використовувати вузли (nodes), з'єднані лініями для створення ігрової логіки.

Типи Blueprint:

- blueprint class. Використовуються для створення об'єктів (актори, персонажі, тощо);
- level blueprint. Містить логіку для конкретного рівня, наприклад, контроль камер, події чи відображення інтерфейсу;
- widget blueprint. Використовується для створення інтерфейсу користувача (UI), наприклад меню, HUD або інформаційних панелей та вікон.

Вузли (Nodes) - основні елементи Event Graph, які виконують різні функції. Кожен вузол може виконувати різні завдання, такі як зміна

властивостей об'єкта, виклик функцій, запуск подій. Розглянемо приклади вузлів.

Вузли подій (Event Nodes) - початкові точки для виконання логіки.

Приклад вузлів подій:

- event begin play - викликається один раз при запуску рівня або актора;
- event tick - викликається на кожному кадрі гри;
- event on destroyed - викликається, коли актор видаляється зі сцени.

Функціональ вузли (Function Nodes) - виконують конкретні дії або розрахунки, такі як зміна позиції об'єкта, застосування матеріалів тощо.

Приклад функціональних вузлів:

- get actor location - повертає позицію актора;
- set actor location - переміщує актора в задану позицію;
- destroy actor - видаляє актора зі сцени;
- spawn actor from class - створює актора на сцені з заданого класу;
- get all actors of class - отримує список всіх акторів конкретного класу;

Вузли змінних (Variable Nodes) - дозволяють читати і записувати значення змінних.

Приклад вузлів змінних:

- set variable - присвоює значення змінній;
- get variable - отримує поточне значення змінної.

Вузли управління потоком (Flow Control Nodes) - контролюють логіку виконання сценаріїв:

Приклад вузлів управління потоком:

- branch - умовний оператор аналог «if» в програмуванні;
- sequence – дозволяє створювати послідовність виконання логіки;
- delay – створює затримки в виконанні певної логіки;
- for loop – аналог стандартного циклу «for» в програмуванні;
- do once - виконує логіку лише один раз.

3.2 Реалізація інтерфейсу

Інтерфейс гри реалізовано за допомогою Widget Blueprint. Базові віджети для створення інтерфейсу гри наведені нижче.

Полотно (Canvas Panel) - основний контейнер для створення інтерфейсу користувача. Він забезпечує гнучке позиціонування і прив'язку віджетів, дозволяючи розміщувати дочірні віджети в будь-якому місці.

Перемикач віджетів (Widget Switcher) - базовий віджет, який дозволяє перемикатися між різними дочірніми віджетами.

Розмірна коробка (Size Box) - базовий віджет контейнер, який дозволяє контролювати розмір дочірніх віджетів.

Вертикальна коробка (Vertical Box) - базовий віджет контейнер, що дає змогу розташовувати дочірні віджети один під одним у вертикальній колонці.

Горизонтальна коробка (Horizontal Box) - базовий віджет контейнер, що дає змогу розташовувати дочірні віджети один поряд з іншим у горизонтальному ряду.

Поле прокрутки (Scroll Box) - базовий віджет, що дає змогу відображати велику кількість віджетів в обмеженій області з можливістю прокрутки.

Рамка (Border) - базовий віджет контейнер, який може мати лише один дочірній віджет. Рамка надає можливість оточити дочірній віджет фоновим зображенням.

Випадаючий список (Combo Box) - базовий віджет, що дає змогу користувачу вибрати один елемент у випадаючому списку.

Галочка (Check Box) - базовий інтерактивний віджет, який дозволяє користувачам встановлювати або знімати позначку.

Кнопка (Button) – базовий віджет з яким користувач може взаємодіяти шляхом натискання.

На рисунку 3.4 наведено реалізацію Game Instance, який відповідає за створення файлу збереження налаштувань звуку та чутливості миші при першому відкритті гри.

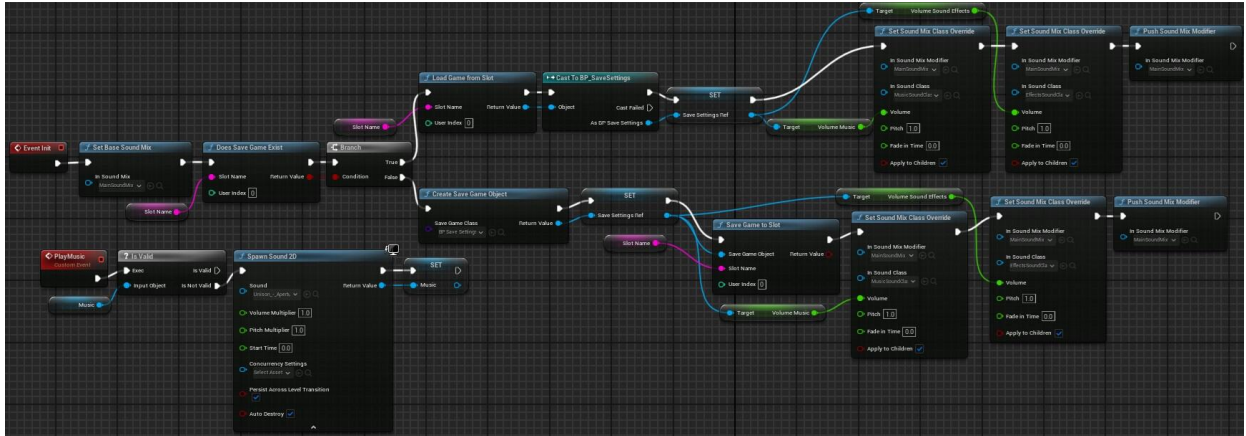


Рисунок 3.4 – Реалізація Game Instance

Реалізоване головне меню наведено на рисунку 3.5.

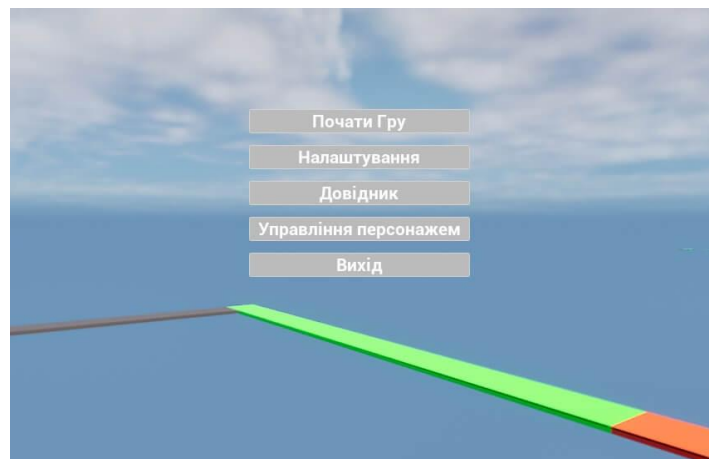


Рисунок 3.5 – Реалізоване головне меню

Реалізоване вікно довідника наведено на рисунку 3.6.

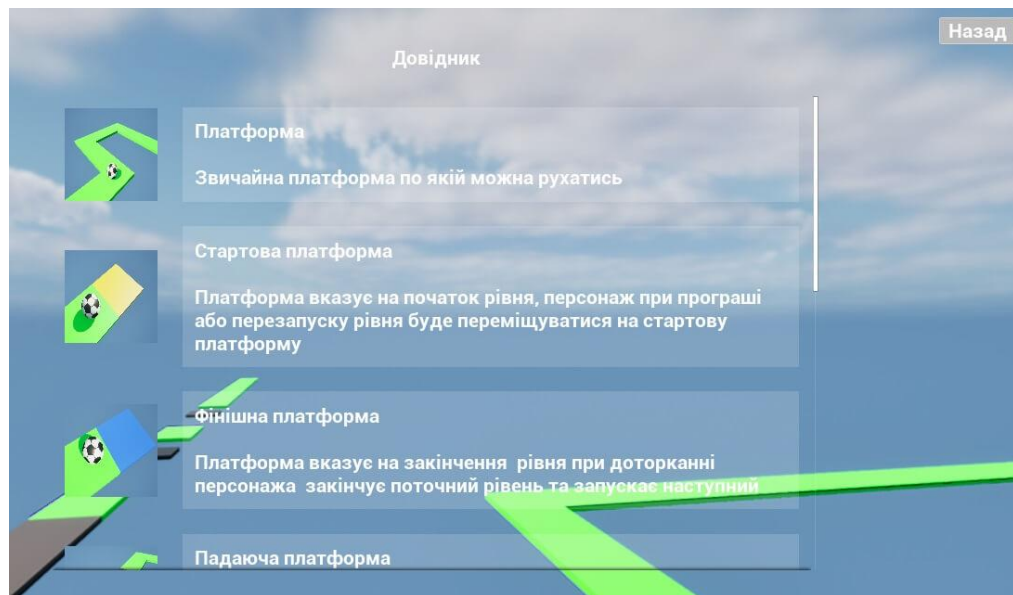


Рисунок 3.6 – Реалізоване вікно довідника

Реалізоване інформаційне вікно управління персонажем наведено на рисунку 3.7.



Рисунок 3.7 – Реалізоване інформаційне вікно управління персонажем

Реалізацію роботи кнопок в головному меню представлено на рисунках 3.8, 3.9.

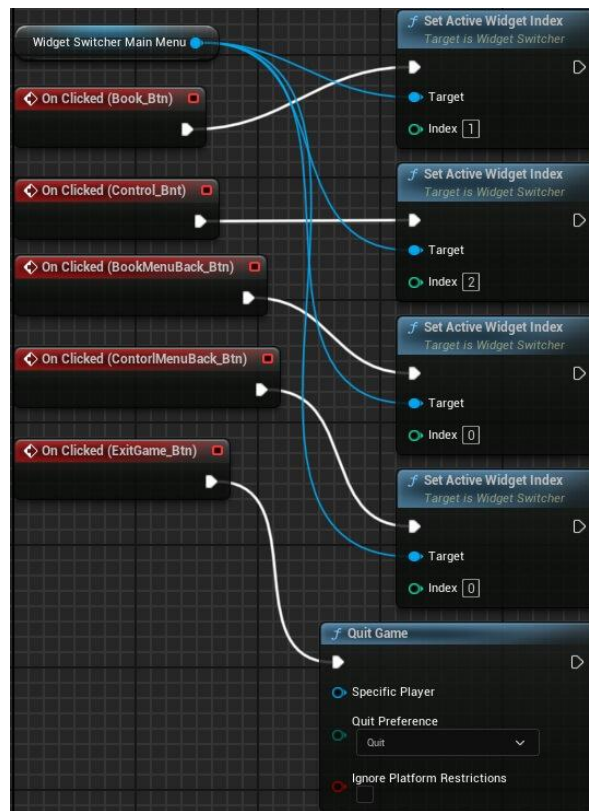


Рисунок 3.8 – Реалізація роботи кнопок в головному меню

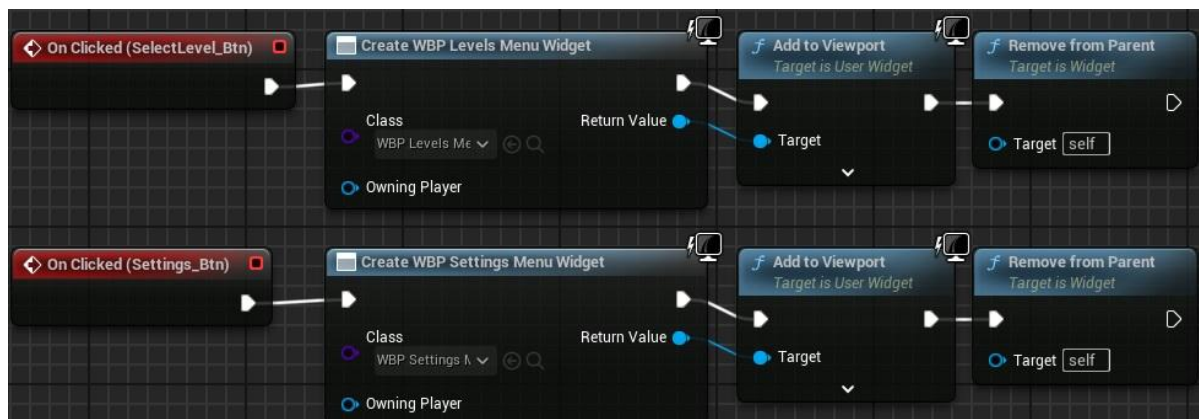


Рисунок 3.9 – Реалізація роботи кнопок в головному меню

Реалізоване меню вибору ігрових локацій наведено на рисунку 3.10.

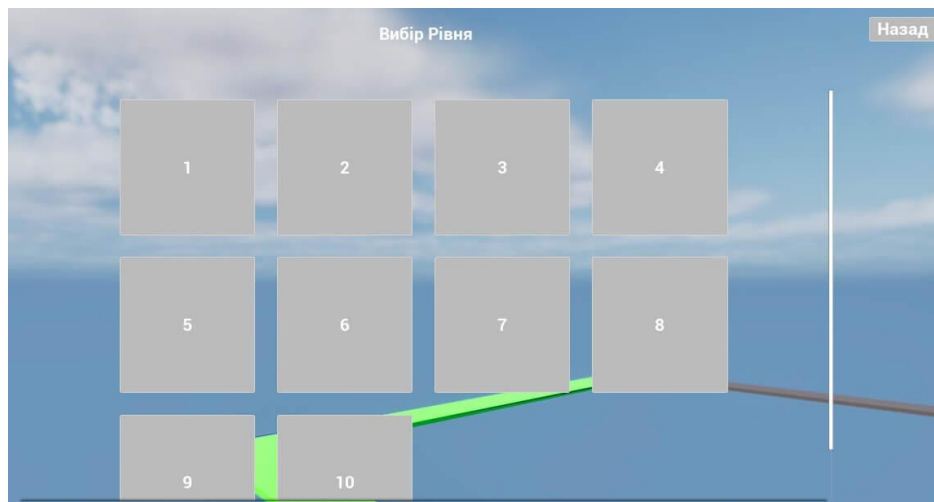


Рисунок 3. 10 – Реалізоване меню вибору ігрових локацій

Реалізацію роботи кнопок в меню вибору ігрових локацій наведено на рисунку 3.11.



Рисунок 3.11 – Реалізація роботи кнопок в меню вибору ігрових локацій

Реалізацію розблокування рівнів наведено на рисунку 3.12.

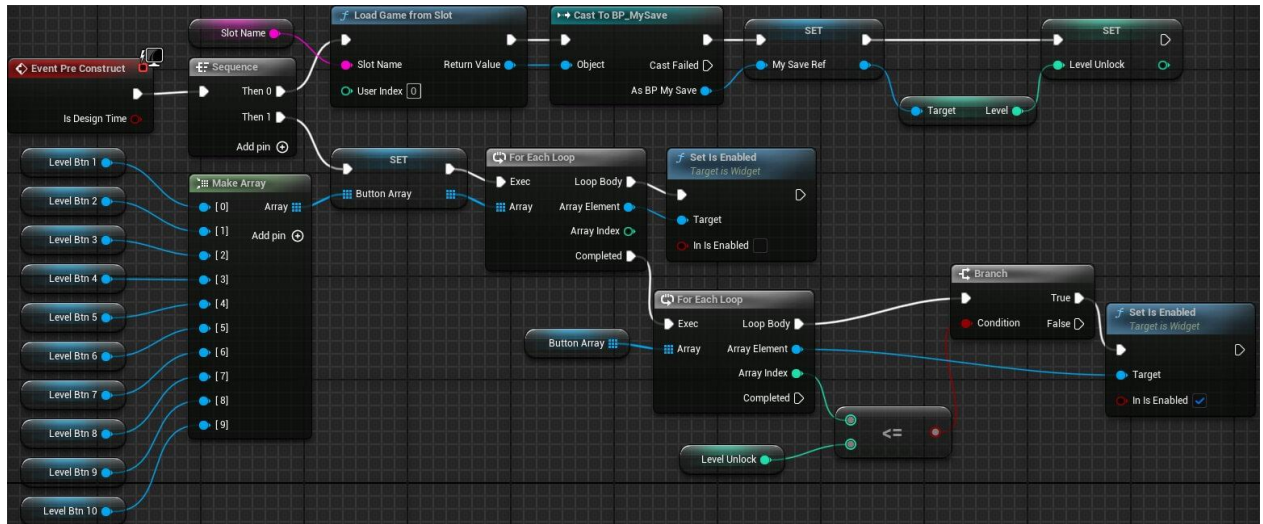


Рисунок 3.12 – Реалізація розблокування рівнів

Реалізоване меню налаштувань гри наведено на рисунку 3.13.

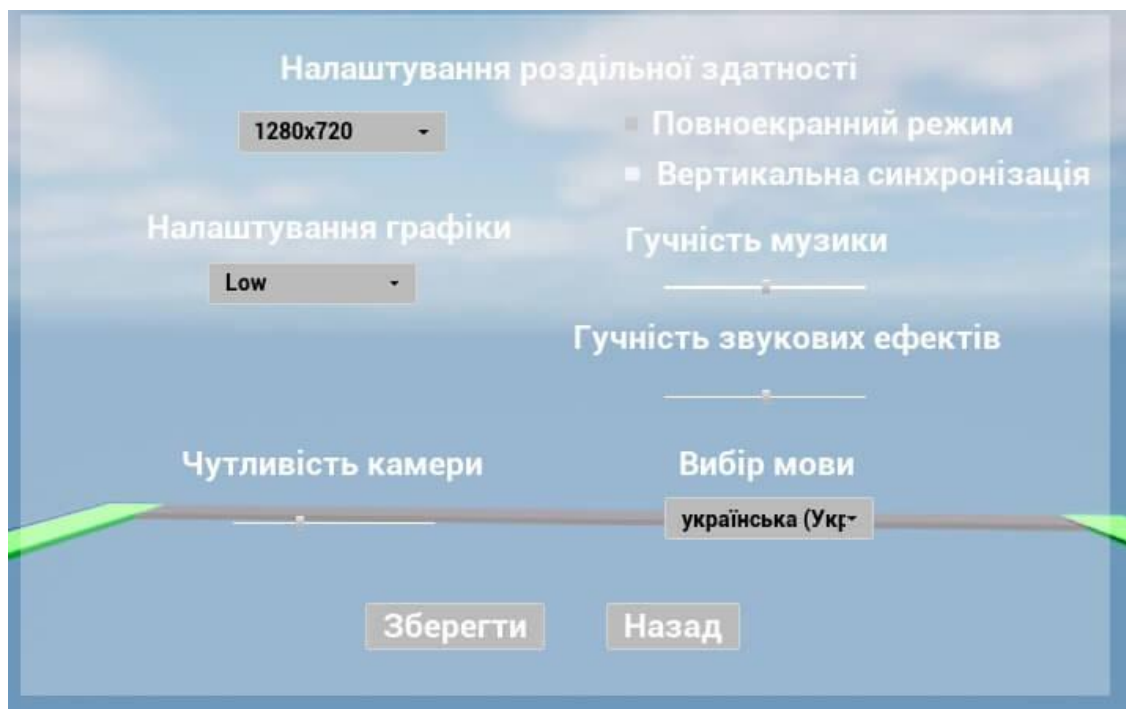


Рисунок 3.13 – Реалізоване меню налаштувань гри

Реалізацію отримання поточних налаштувань гри наведено на рисунках 3.14, 3.15.

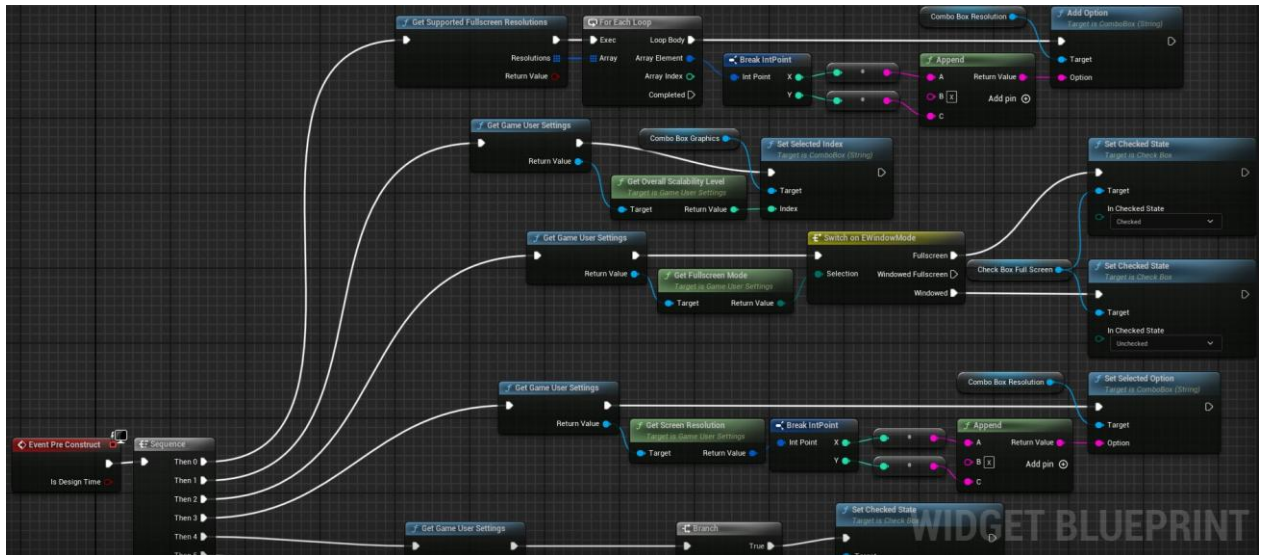


Рисунок 3.14 - Реалізація отримання поточних налаштувань гри

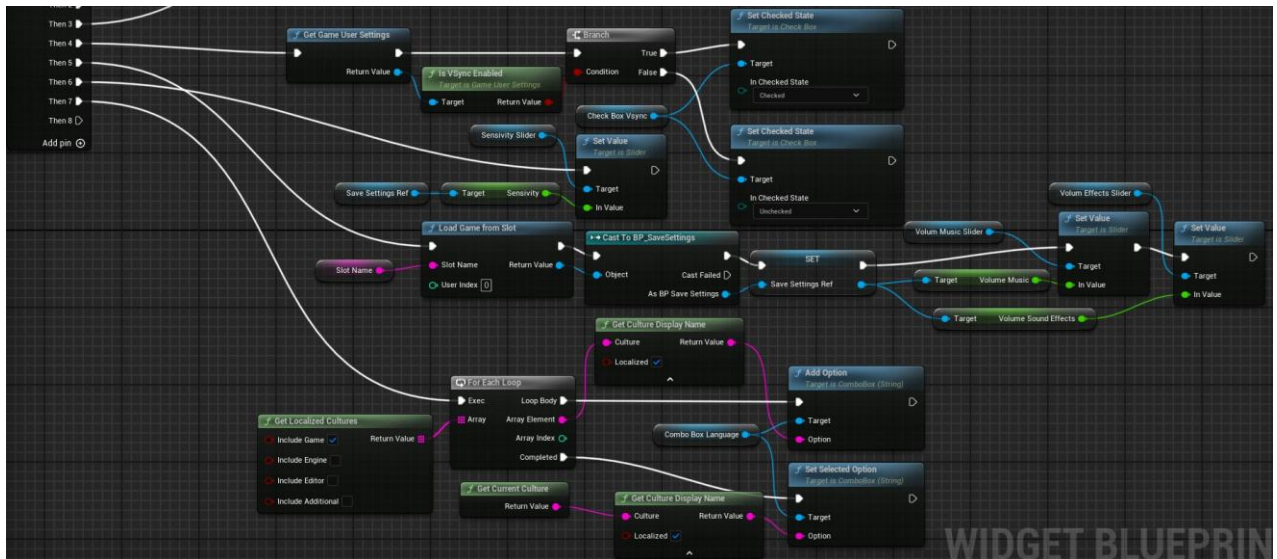


Рисунок 3.15 - Реалізація отримання поточних налаштувань гри

Реалізацію функціоналу зміни налаштувань гри представлено на рисунках 3.16 - 3.20.

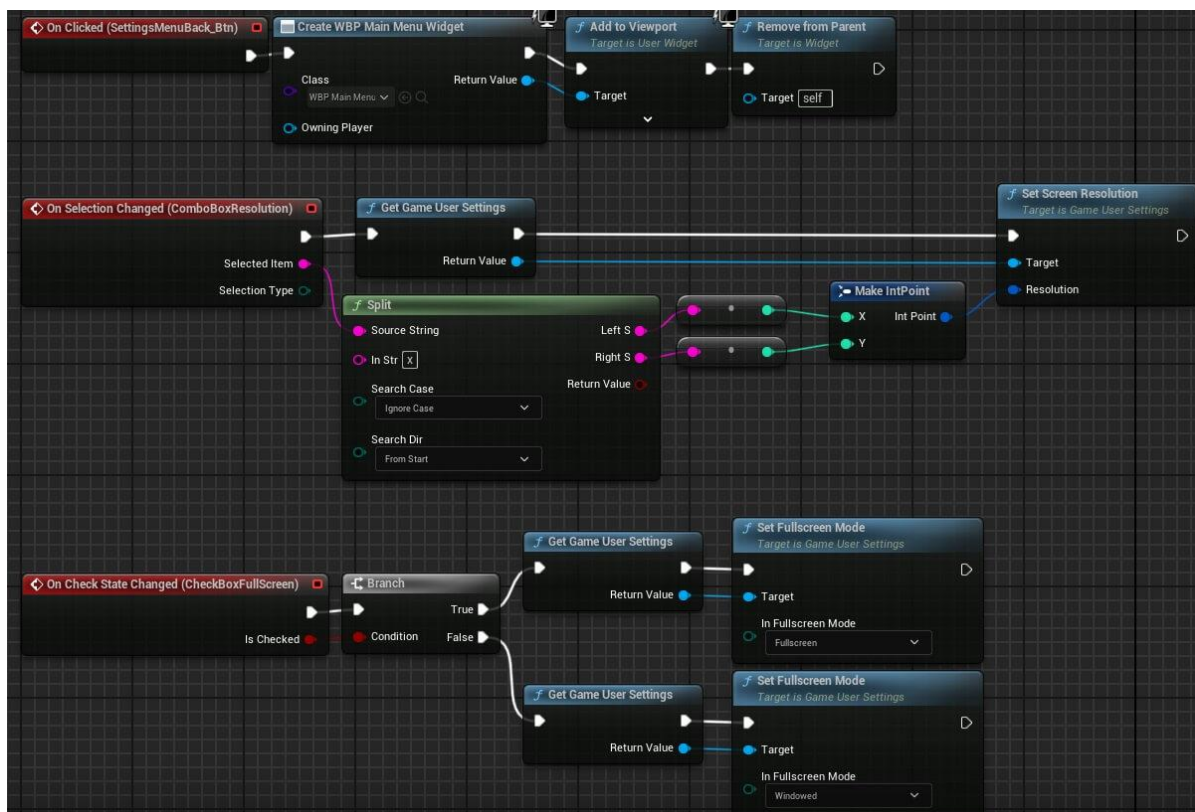


Рисунок 3.16 – Реалізація функціоналу зміни налаштувань гри

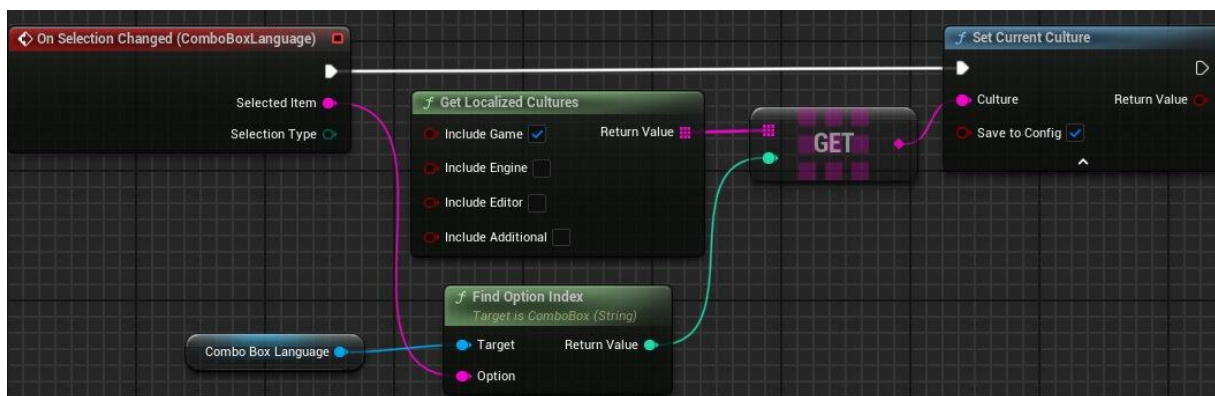


Рисунок 3.17 - Реалізація функціоналу зміни налаштувань гри

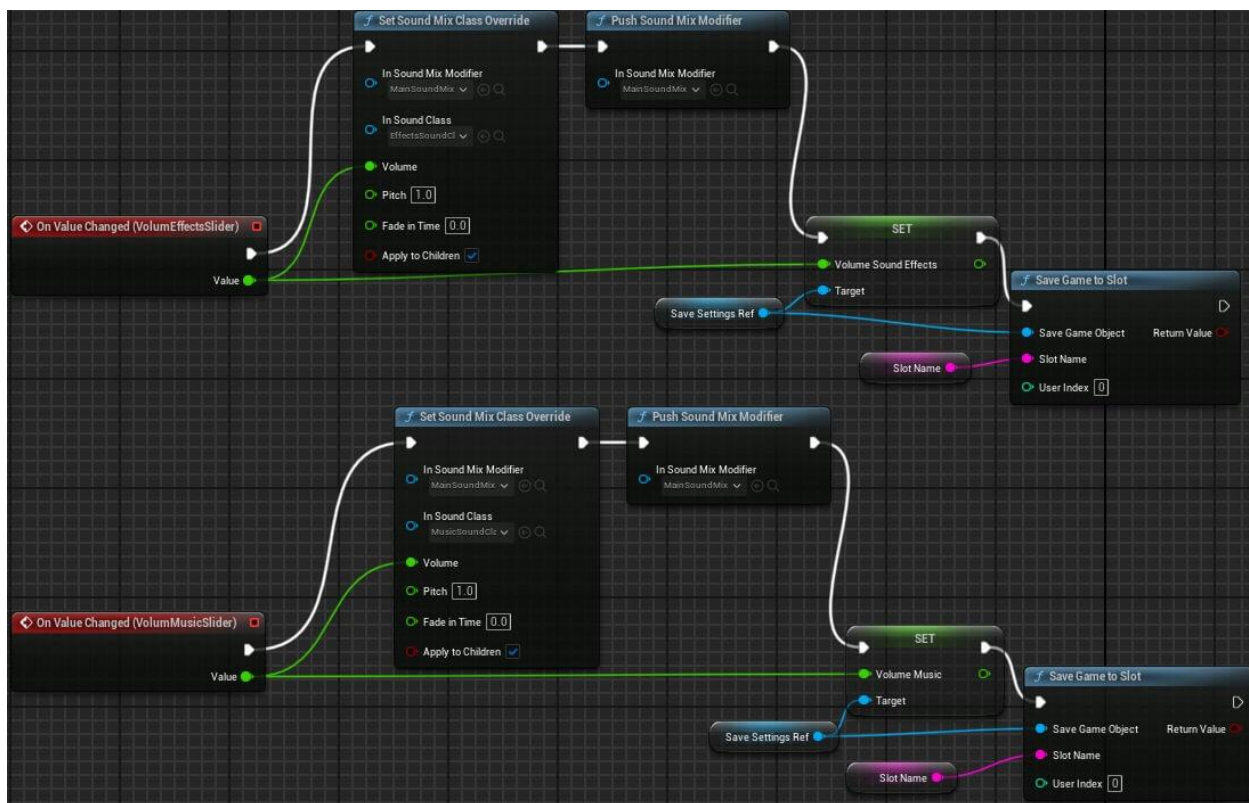


Рисунок 3.18 - Реалізація функціоналу зміни налаштувань гри

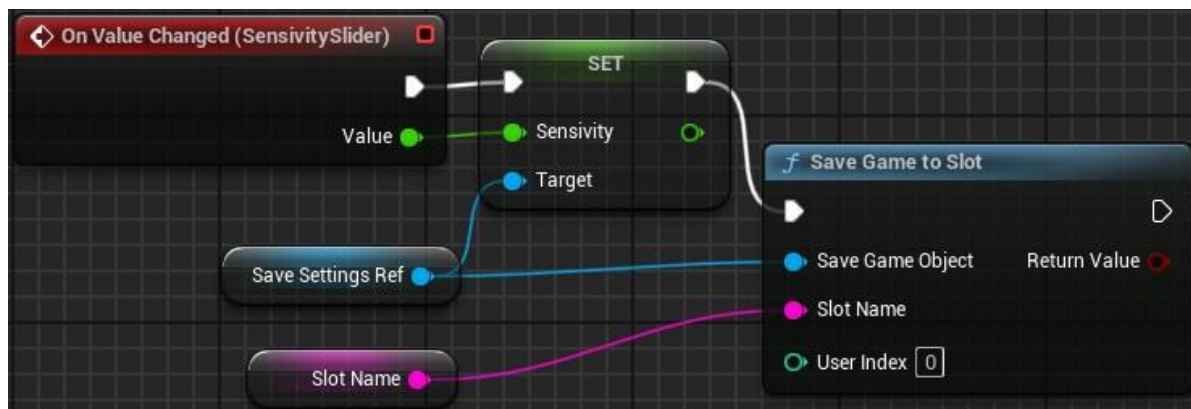


Рисунок 3.19 - Реалізація функціоналу зміни налаштувань гри

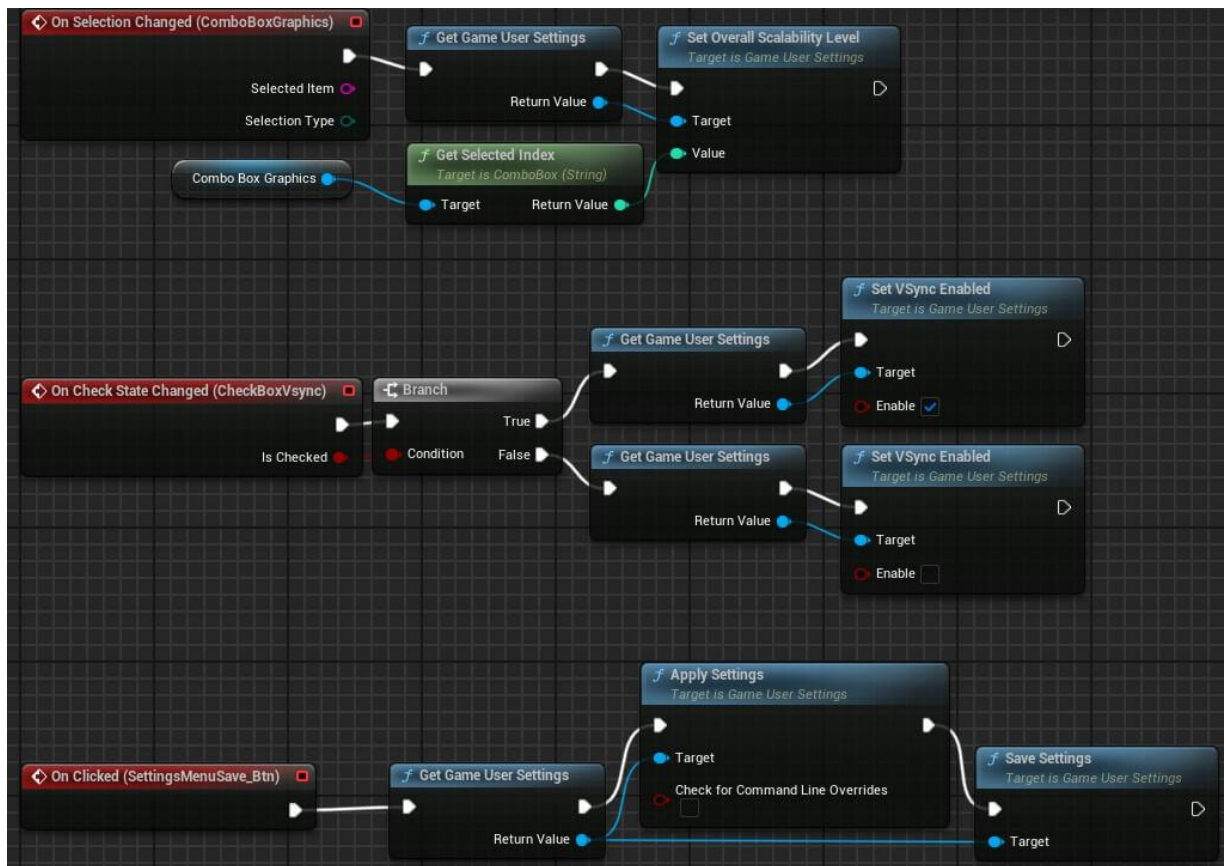


Рисунок 3.20 - Реалізація функціоналу зміни налаштувань гри

Реалізоване меню паузи наведено на рисунку 3.21.

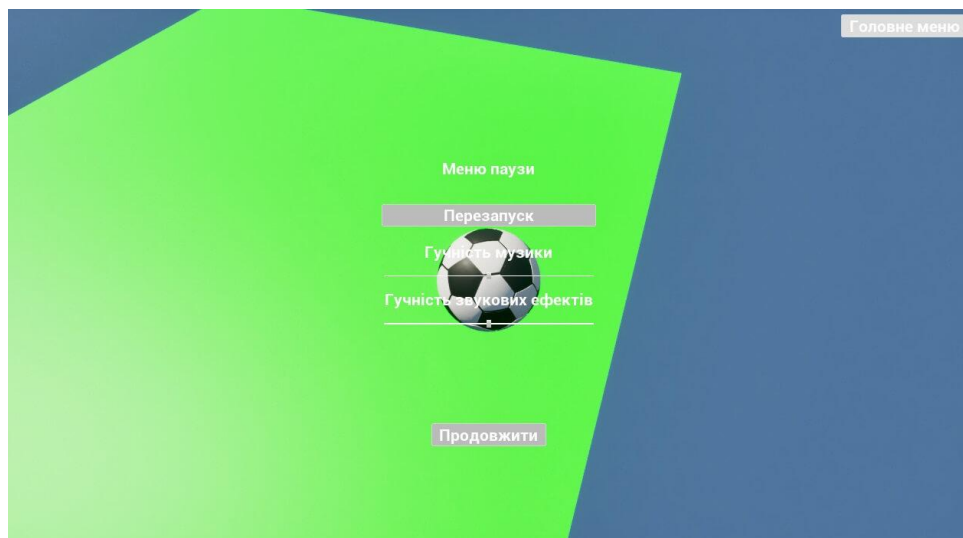


Рисунок 3.21 – Реалізоване меню паузи

Реалізацію функціоналу меню паузи наведено на рисунках 3.22, 3.23.

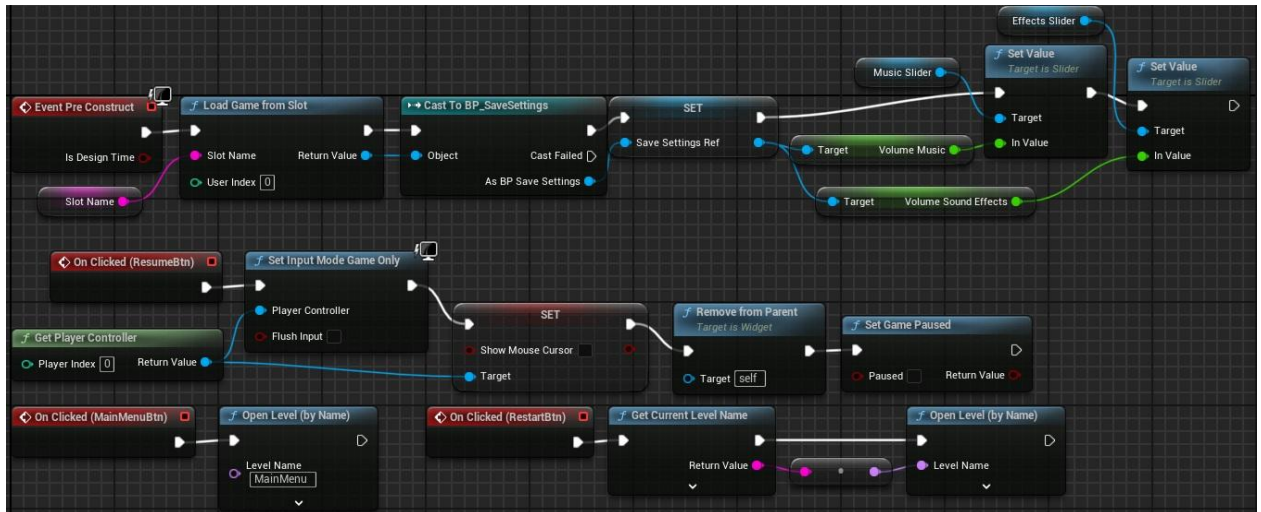


Рисунок 3.22 – Реалізація функціоналу меню паузи

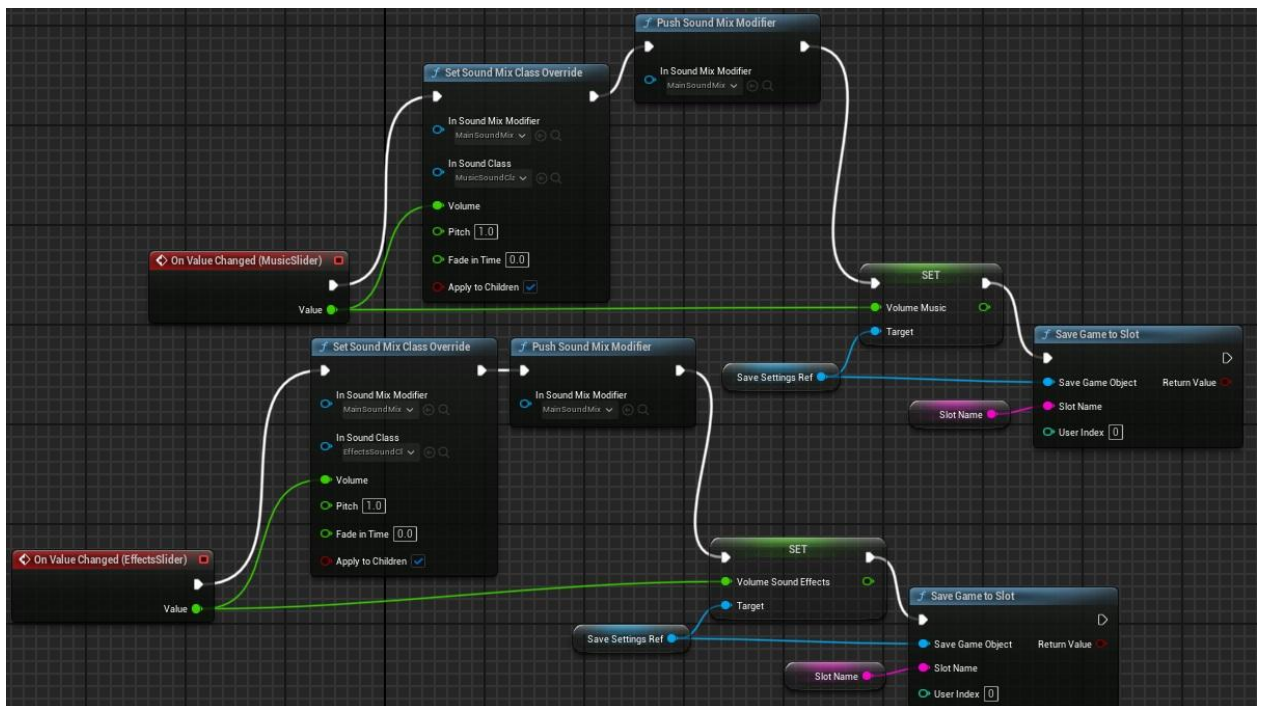


Рисунок 3.23 - Реалізація функціоналу меню паузи

Вікно кінця гри відображується в окремому рівні, який завантажується після проходження останньої ігрової локації. Реалізоване вікно кінця гри наведено на рисунку 3.24.

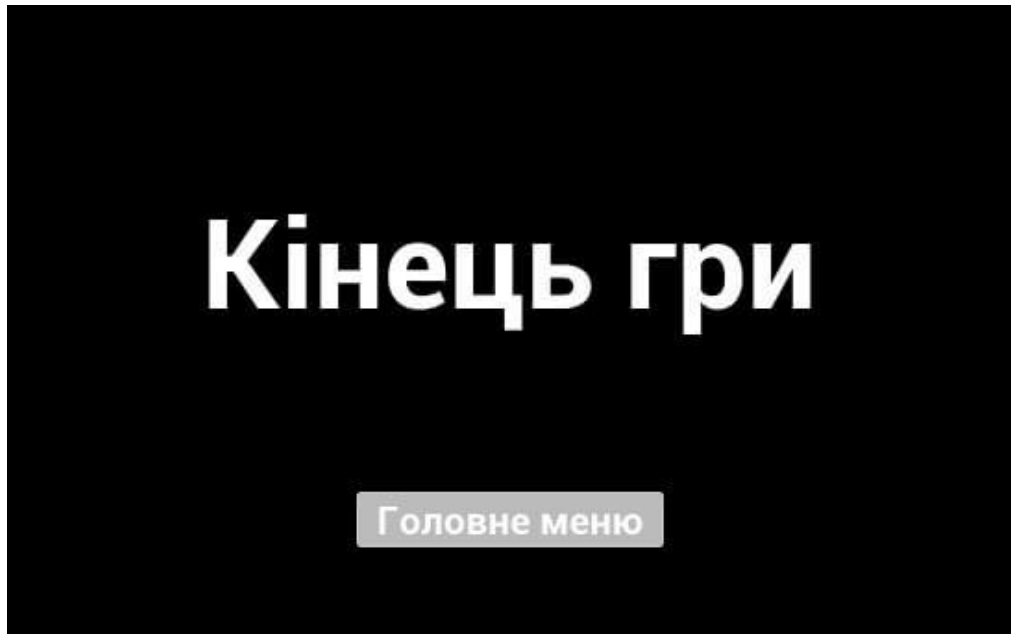


Рисунок 3.24 – Реалізоване вікно кінця гри

Реалізацію кнопки в вікні кінця гри представлено на рисунку 3.25.



Рисунок 3.25 – Реалізація кнопки в вікні кінця гри

3.3 Реалізація ігрових об'єктів та персонажа

Ігрові об'єкти реалізуються на основі класу Actor. Далі наведено опис компонентів, які будуть використовуватися для створення ігрових об'єктів.

Статична сітка (Static mesh) - один із ключових компонентів, що представляє собою 3D-модель. Використовується для побудови рівнів у Unreal Engine. 3D-моделі створюються у спеціальних програмах моделювання, які

потім імпортуються в ігровий рушій. Рівні в основному складаються з акторів з компонентом static mesh.

Камера (Camera) - об'єкт або компонент, який визначає точку огляду з якої гравець бачить віртуальний світ гри.

Пружинний важіль (Spring arm) - компонент утримує дочірні елементи на певній фіксованій відстані від батьківського. В разі зіткнення наближає дочірні елементи до батьківського і повертає їх назад, якщо зіткнень немає.

Коробка зіткнень (Box collision) - компонент, який використовується для виявлення зіткнень або взаємодії між об'єктами в грі. Це прямокутна або кубічна область для перевірки на зіткнення, що дозволяє визначити, коли об'єкт входить або виходить з цієї області.

Сфера зіткнень (Sphere collision) – компонент функціонально аналогічний компоненту box collision, але має форму сфери.

На рисунку 3.26 наведено зовнішній вигляд ігрового об'єкту падаюча платформа.

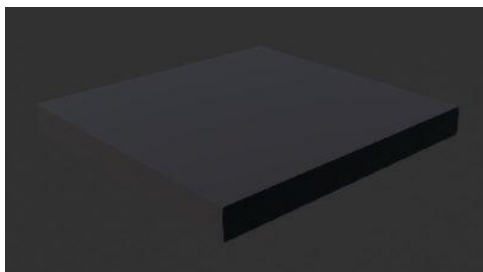


Рисунок 3.26 - Зовнішній вигляд ігрового об'єкту падаюча платформа

Реалізацію механіки падіння платформи наведено на рисунку 3.27. Принцип роботи полягає в перевірці об'єкта, який увійшов в колізію платформи, якщо це ігровий персонаж тоді виконується затримка 1,2 секунди після чого вмикається симуляція фізики і платформа починає падати. Через 5 секунд падіння платформа видаляється.



Рисунок 3.27 – Реалізація механіки падіння платформи

На рисунку 3.28 наведено зовнішній вигляд ігрового об'єкту рухома платформа.

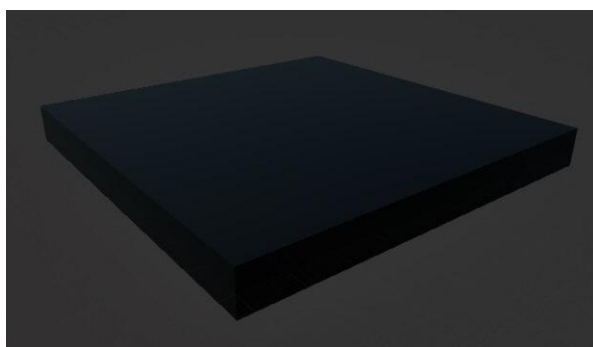


Рисунок 3.28 - Зовнішній вигляд ігрового об'єкту рухома платформа.

Реалізацію механіки руху платформи наведена на рисунку 3.29.

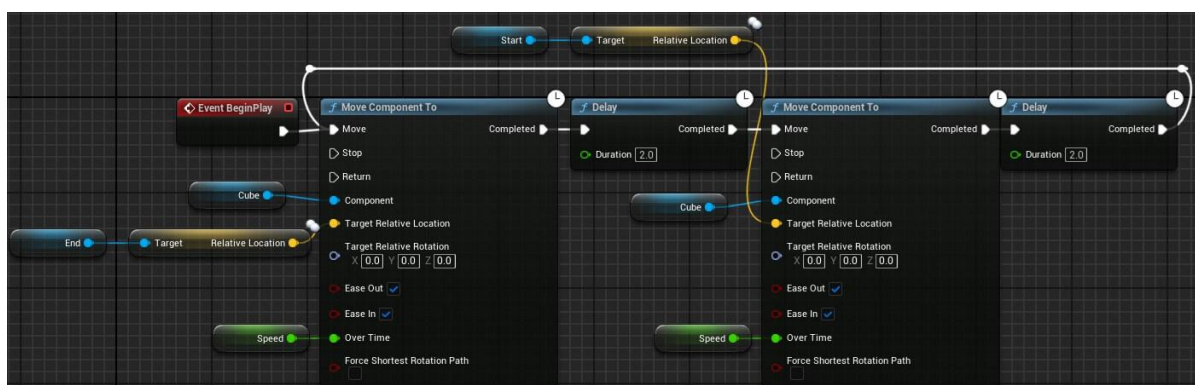


Рисунок 3.29 – Реалізація механіки руху платформи

На рисунку 3.30 наведено зовнішній вигляд ігрового об'єкту обертова платформа.

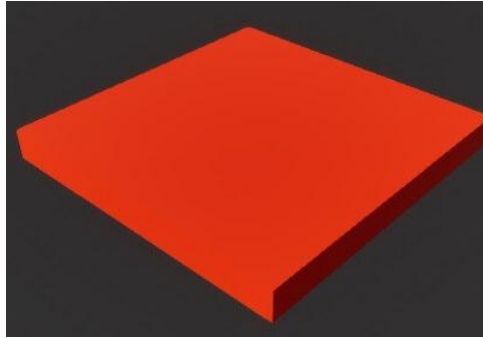


Рисунок 3.30 – Зовнішній вигляд ігрового об’єкту обертова платформа

Реалізацію механіки обертання платформи наведена на рисунку 3.31.

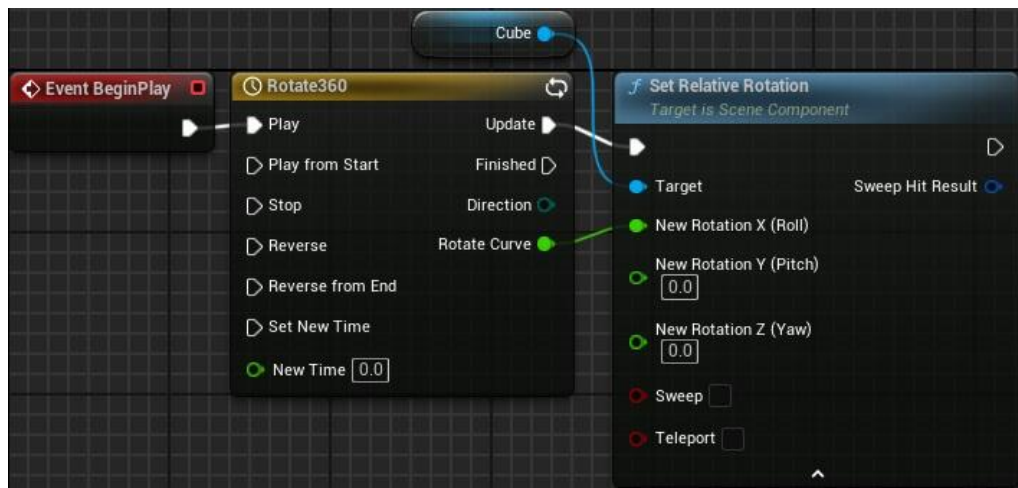


Рисунок 3.31 – Реалізація механіки обертання платформи

На рисунку 3.32 наведено зовнішній вигляд ігрового об’єкту фінішна платформа.

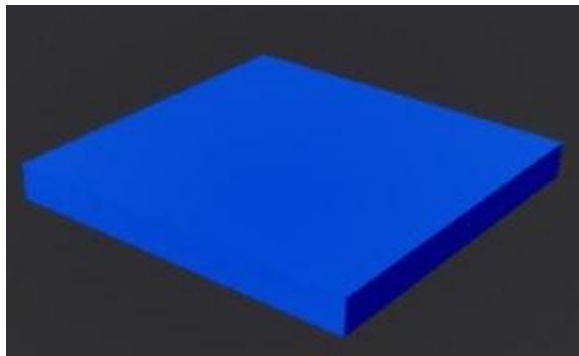


Рисунок 3.32 - Зовнішній вигляд ігрового об’єкту фінішна платформа.

Реалізацію логіки фінішної платформи представлено на рисунку 3.33.

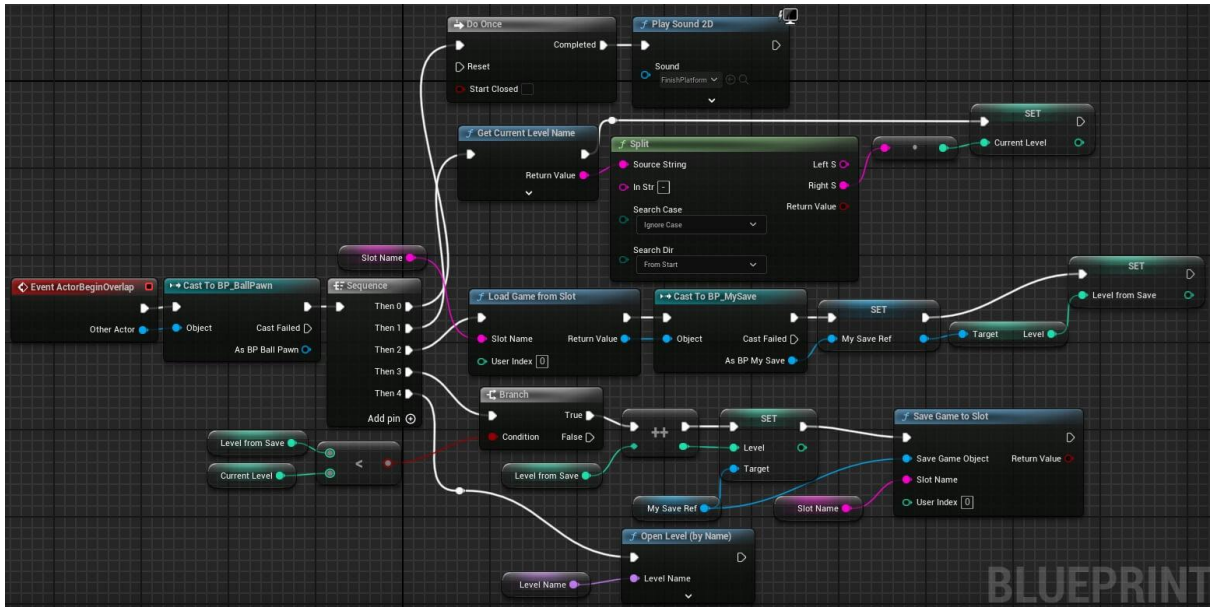


Рисунок 3.33 – Реалізація логіки фінішної платформи

На рисунку 3.34 наведено зовнішній вигляд ігрового об'єкту ящик.

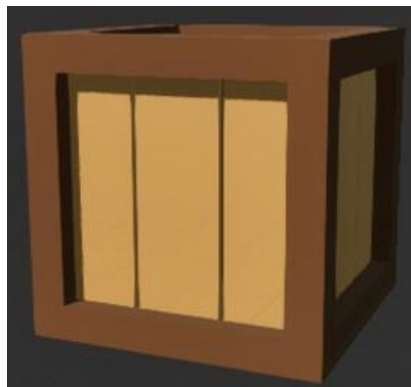


Рисунок 3.34 – Зовнішній вигляд ігрового об'єкту ящик

Реалізацію скрипту видалення ігрового об'єкту ящик при падінні за межі ігрової зони наведено на рисунку 3.35. Принцип роботи полягає в перевірці координат висоти через інтервал часу. Якщо координати висоти об'єкта перевищують порогове значення об'єкт видаляється.

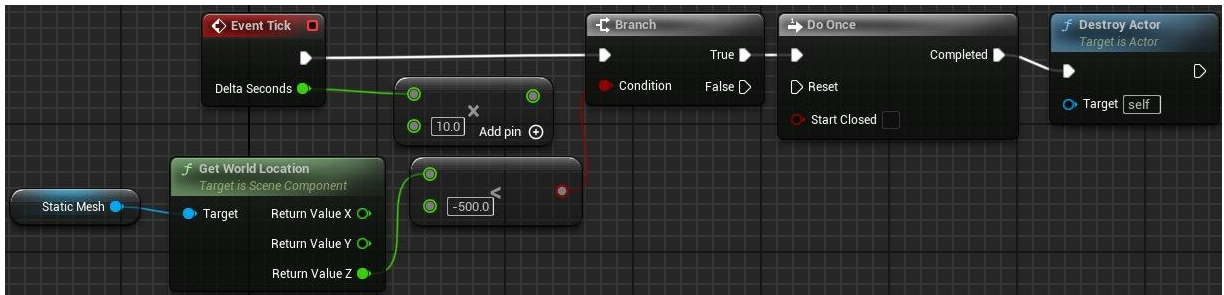


Рисунок 3.35 – Реалізація скрипту видалення ігрового об’єкта ящик при падінні за межі ігрової зони

Ігровий об’єкт ящик симулює фізику для того, щоб ігровий персонаж міг здвинути його. Налаштування фізики представлені на рисунку 3.36.

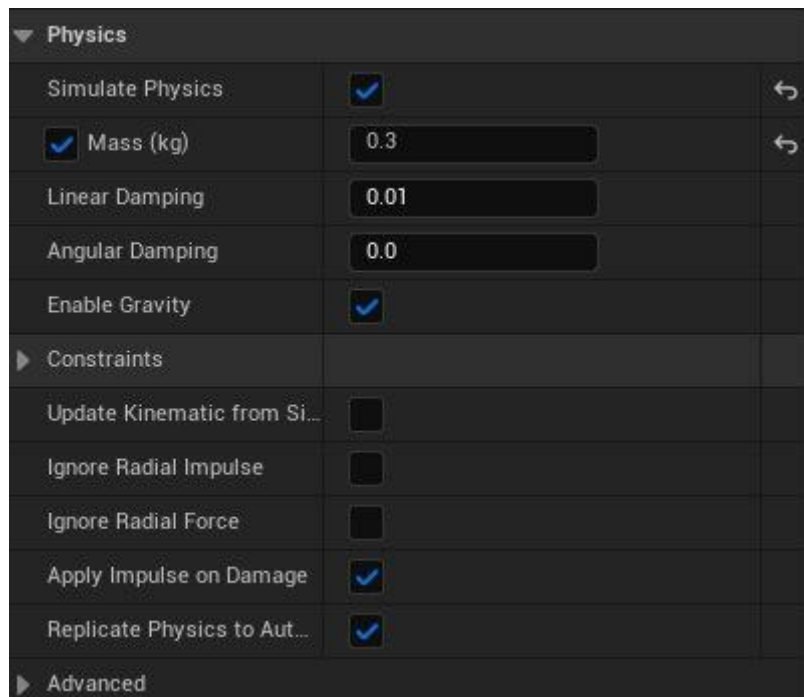


Рисунок 3.36 - Налаштування фізики ігрового об’єкту ящик

На рисунку 3.37 наведено зовнішній вигляд ігрового об’єкту шипи.



Рисунок 3.37 - Зовнішній вигляд ігрового об'єкту шипи

Реалізацію механіки ігрового об'єкту шипи представлено на рисунку 3.38.

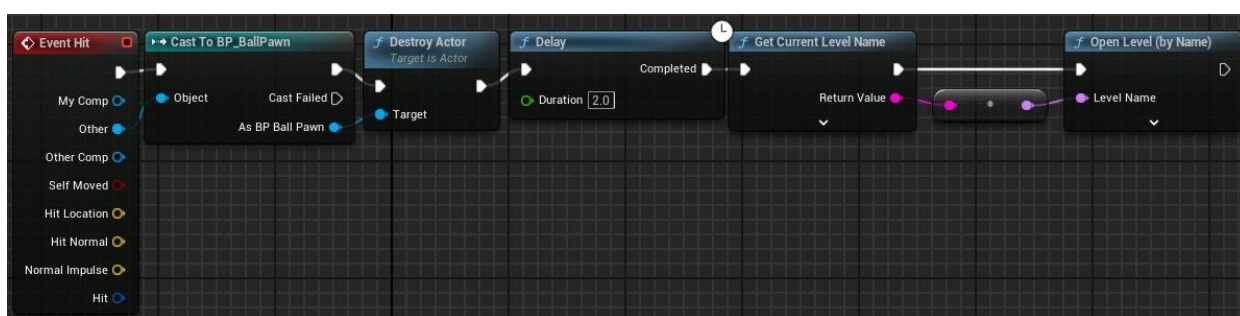


Рисунок 3.38 – Реалізація механіки ігрового об'єкта шипи

На рисунку 3.39 наведено зовнішній вигляд ігрового об'єкту телепорт.

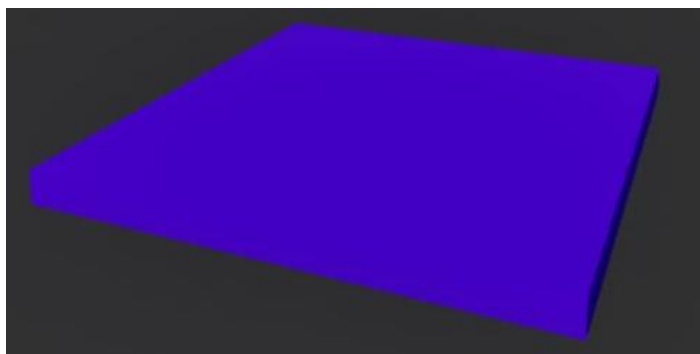


Рисунок 3.39 - Зовнішній вигляд ігрового об'єкту телепорт

Реалізацію механіки телепорту представлено на рисунку 3.40.

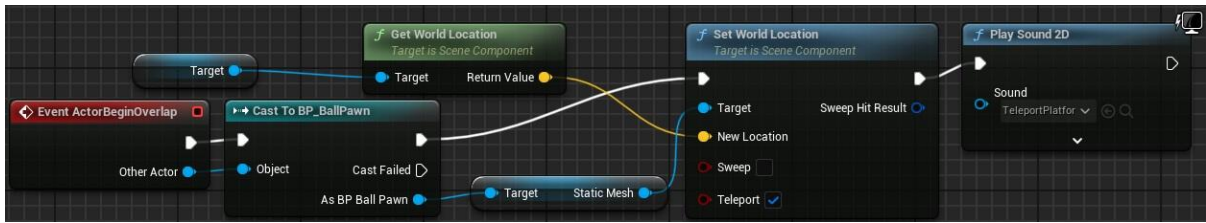


Рисунок 3.40 - Реалізація механіки телепорту

Базові платформи не мають ігрової логіки і призначені лише для того, щоб по ним пересувався ігровий персонаж. Вони мають зелений колір і відрізняються лише розмірами. Стартова платформа також відноситься до базових платформ але відрізняється жовтим кольором. Базові платформи можна побачити на рисунку 3.41.

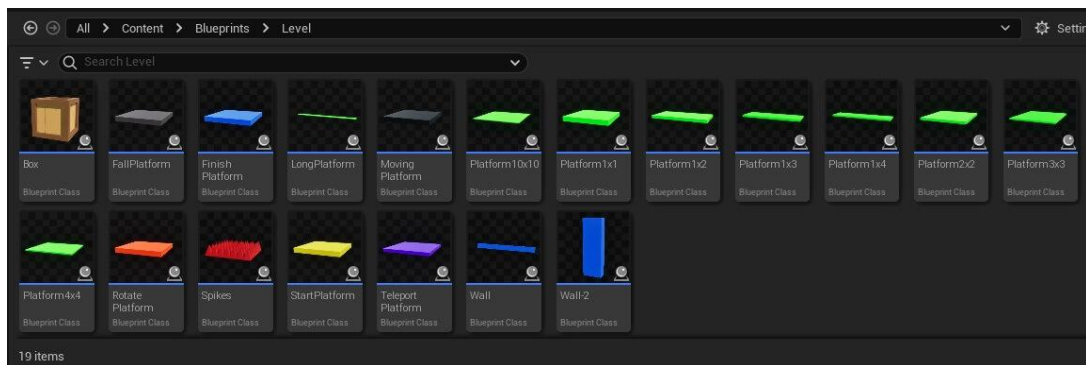


Рисунок 3.41 – Ігрові об'єкти для побудови рівня

Ігровий персонаж реалізовується за допомогою класу Pawn в який додані наступні компоненти:

- static mesh;
- camera;
- spring arm.

Зовнішній вигляд ігрового персонажа представлено на рисунку 3.42.

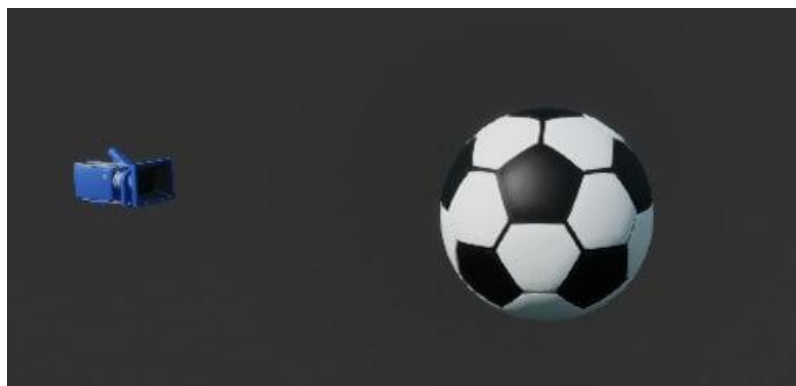


Рисунок 3.42 – Зовнішній вигляд ігрового персонажа

Для управління персонажем необхідно визначити осі, які будуть відповідати за рух ігрового персонажа та управління камерою. Осі задаються в налаштуваннях проєкту. Налаштування осей в проєкті наведено на рисунку 3.43.



Рисунок 3.43 – Налаштування осей в проєкті

Рух персонажа реалізовано за допомогою фізики, тому для компонента static mesh необхідно увімкнути симуляцію фізики та налаштувати параметри фізики. Налаштування фізики ігрового персонажа наведені на рисунку 3.44.

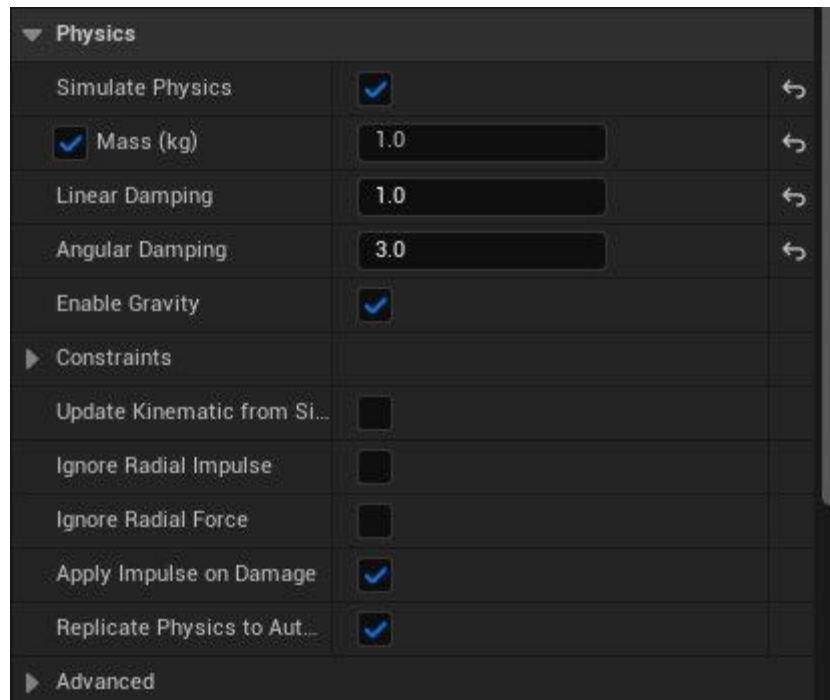


Рисунок 3.44 – Налаштування фізики ігрового персонажа

Реалізацію руху ігрового персонажа наведено на рисунку 3.45.

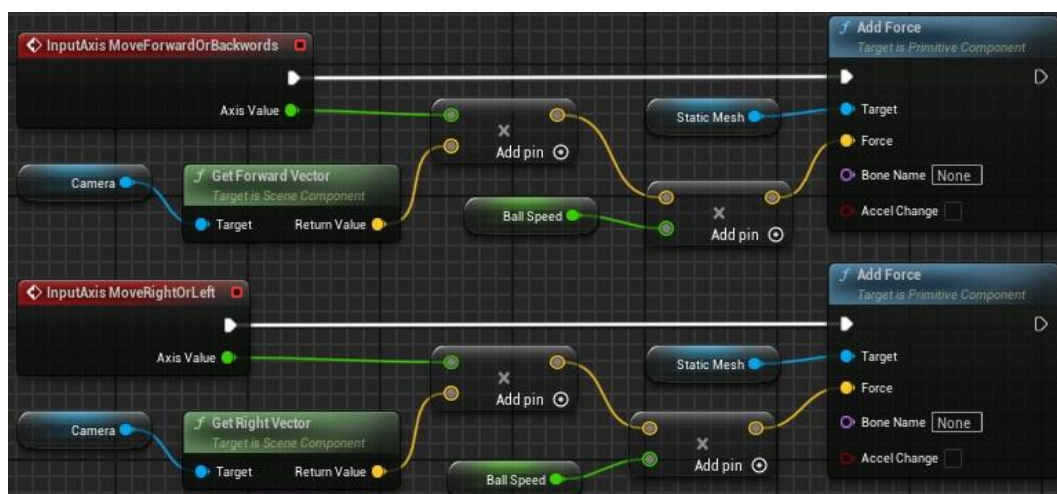


Рисунок 3.45 – Реалізація руху персонажа

Реалізацію управління камерою представлено на рисунку 3.46. Реалізацію наближення та віддалення камери до ігрового персонажа наведено на рисунку 3.47.

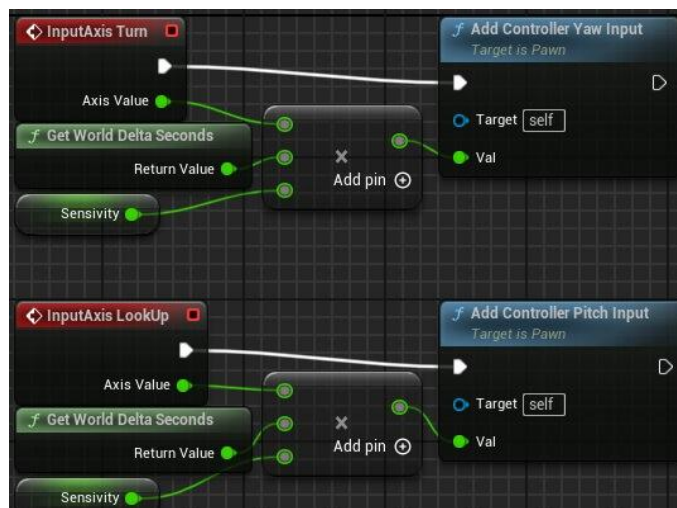


Рисунок 3.46 – Реалізація управління камерою

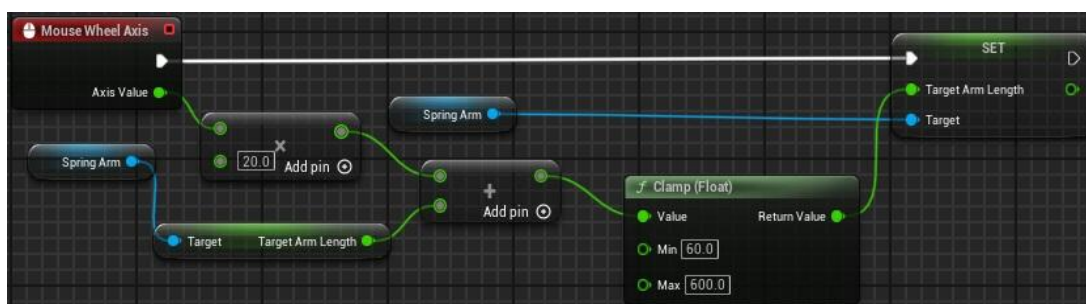


Рисунок 3.47 – Реалізація наближення та віддалення камери до ігрового персонажа

Реалізацію завантаження та встановлення параметру чутливості камери для ігрового персонажа представлено на рисунку 3.48.

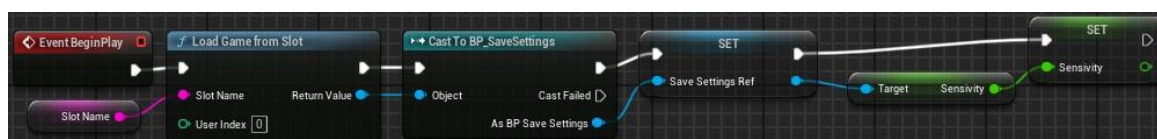


Рисунок 3.48 – Реалізація завантаження та встановлення параметру чутливості камери ігрового персонажа

При ігровому процесі персонаж може впасти з платформ. Для того, щоб перезавантажити рівень реалізовано скрипт, який відстежує координату висоти ігрового персонажа, якщо персонаж впаде нижче заданої позначки висоти то рівень перезавантажиться. Реалізацію скрипту представлено на рисунку 3.49.

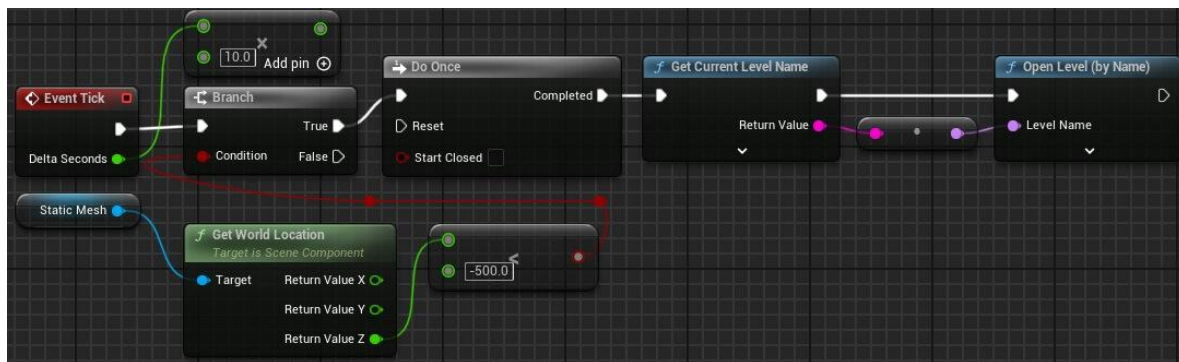


Рисунок 3.49 – Реалізація скрипту перезавантаження рівня при падінні ігрового персонажа

Реалізацію відкриття меню паузи наведено на рисунку 3.50.

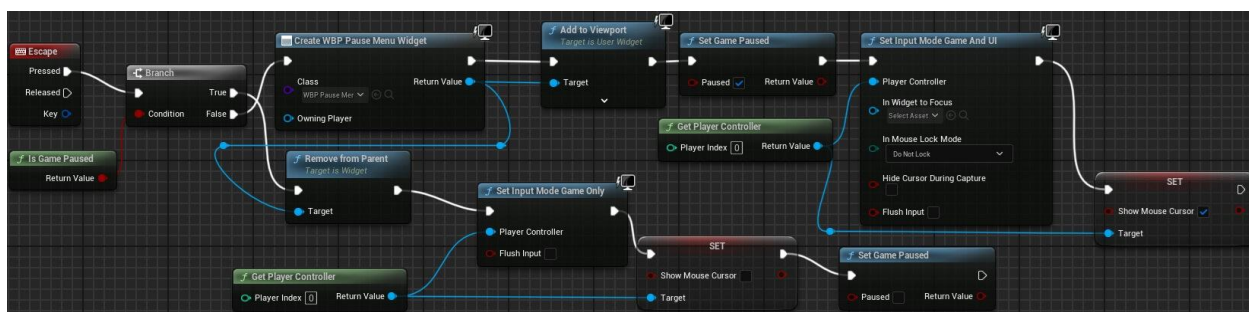


Рисунок 3.50 - Реалізація відкриття меню паузи

Реалізацію розпадання ігрового персонажа на кілька сфер наведено на рисунку 3.51, зовнішній вигляд сфер представлено на рисунку 3.52, реалізацію видалення сфер через проміжок часу наведено на рисунку 3.53.

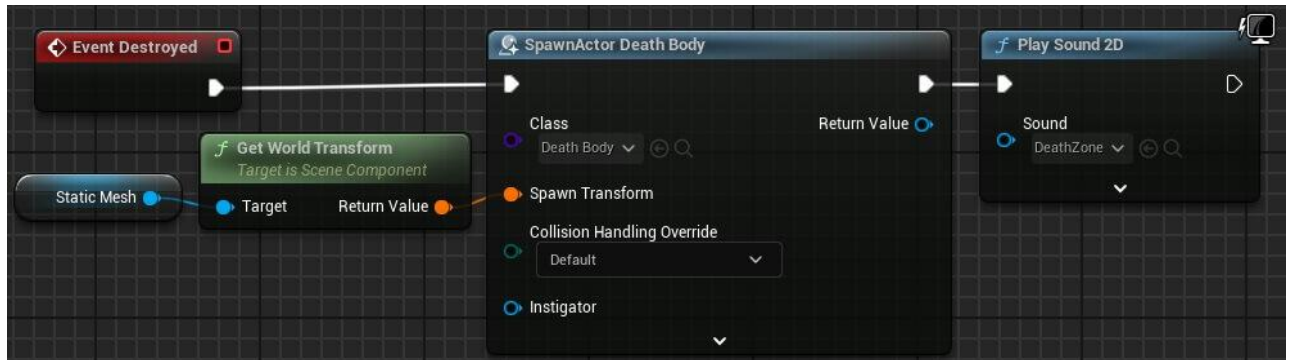


Рисунок 3.51 - Реалізація розпадання ігрового персонажа на кілька сфер



Рисунок 3.52 – Зовнішній вигляд сфер

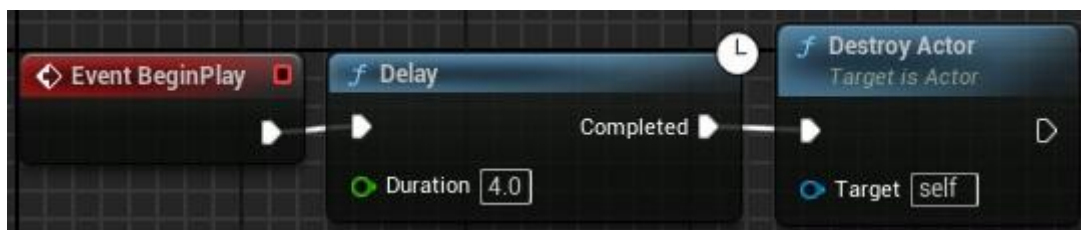


Рисунок 3.53 - Реалізація видалення сфер

3.4 Реалізація ігрових локацій

Використовуючи створені ігрові об'єкти побудовано десять ігрових локацій. Кожна ігрова локація в окремому рівні. Далі наведені побудовані ігрові локації.

Ігрова локація № 1 представлена на рисунку 3.54, вона знайомить гравця з основними ігровими об'єктами: падаючими платформами, обертовими платформами, телепортом, рухомими платформами.

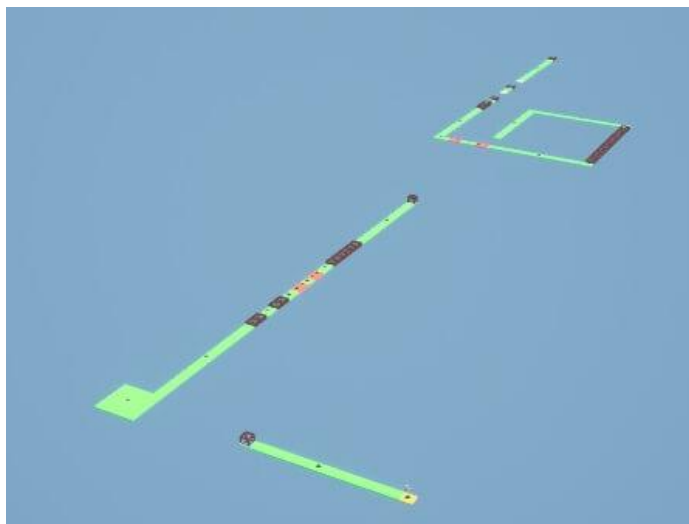


Рисунок 3.54 – Ігрова локація № 1

На рисунку 3.55 наведена ігрова локація № 2. На цій локації гравець знайомиться з ігровим об'єктом «ящик» та вчиться взаємодіяти з ним.

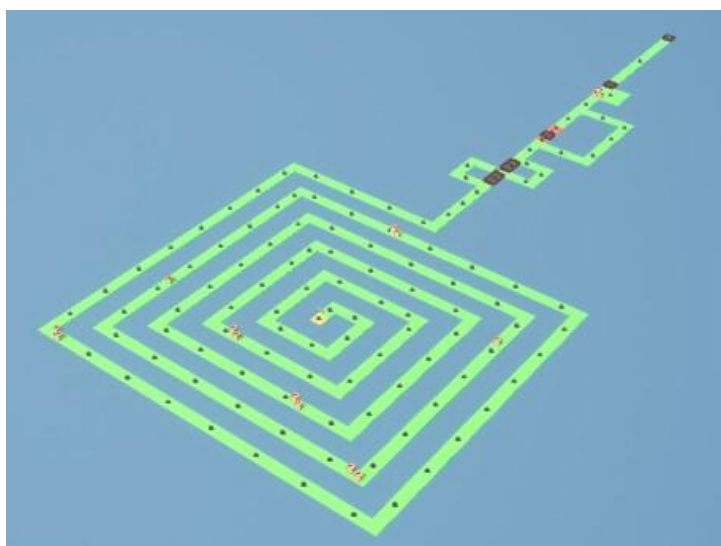


Рисунок 3.55 - Ігрова локація № 2

Ігрова локація № 3 знайомить гравця з ігровим об'єктом «шипи». Ігрова локація № 3 наведена на рисунку 3.56.

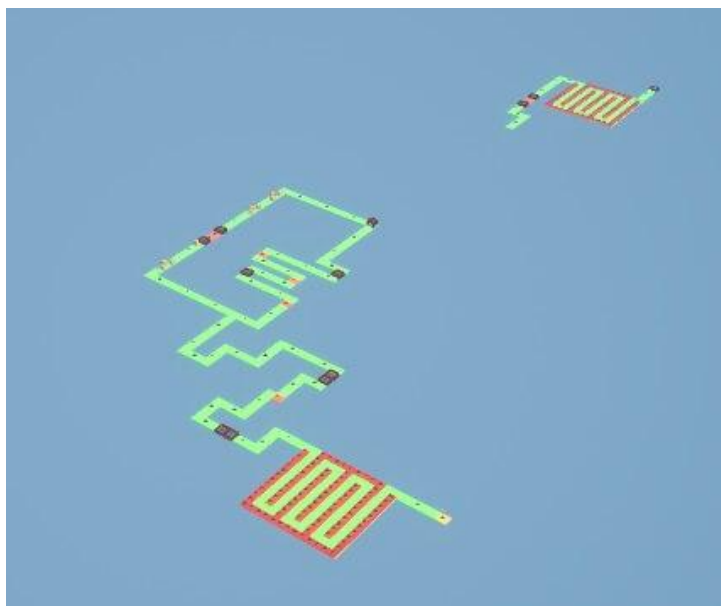


Рисунок 3.56 - Ігрова локація № 3

Ігрова локація № 4 представляє собою лабіринт з падаючих платформ. По краях лабіринту розташовані телепорти, які ведуть на віддалені платформи. Лише один з чотирьох телепортів веде на платформу з фінішем. Ігрова локація № 4 представлена на рисунку 3.57.



Рисунок 3.57 - Ігрова локація № 4

Ігрова локація № 5 наведена на рисунку 3.58.

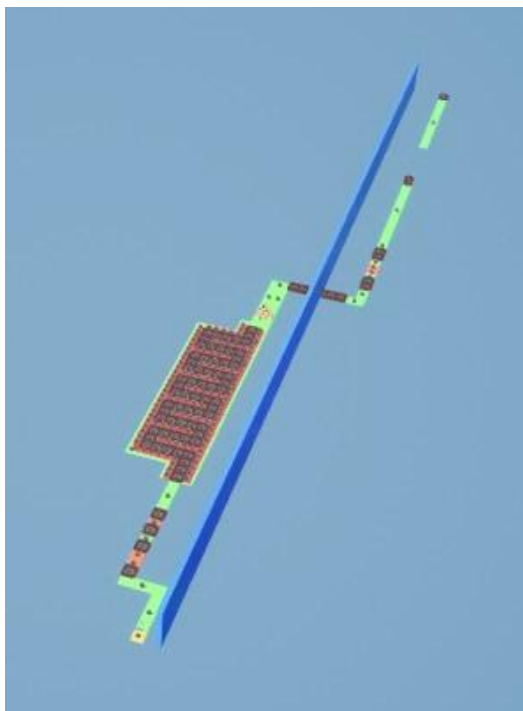


Рисунок 3.58 - Ігрова локація № 5

Ігрова локація № 6 представляє собою лабіринт з телепортів, де гравцю потрібно обрати правильний шлях. Ігрова локація № 6 представлено на рисунку 3.59.

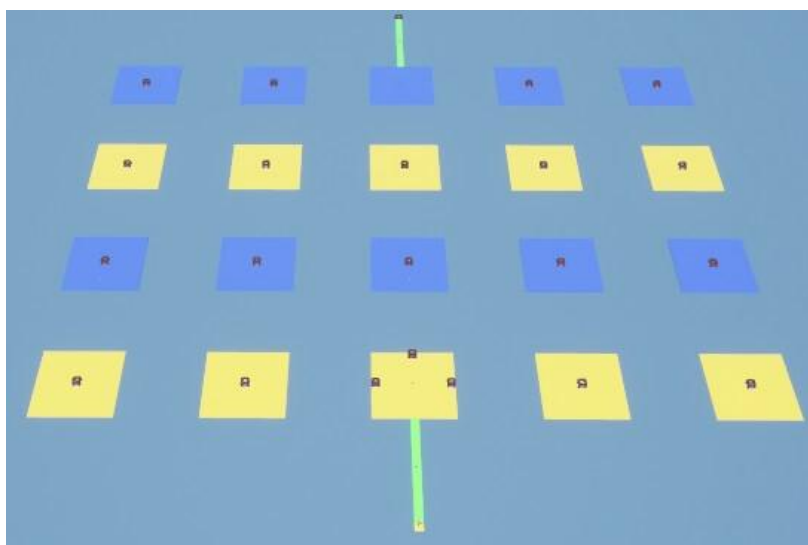


Рисунок 3.59 - Ігрова локація № 6

Ігрова локація № 7 наведена на рисунку 3.60.



Рисунок 3.60 - Ігрова локація № 7

Ігрова локація № 8 представлена на рисунку 3.61.

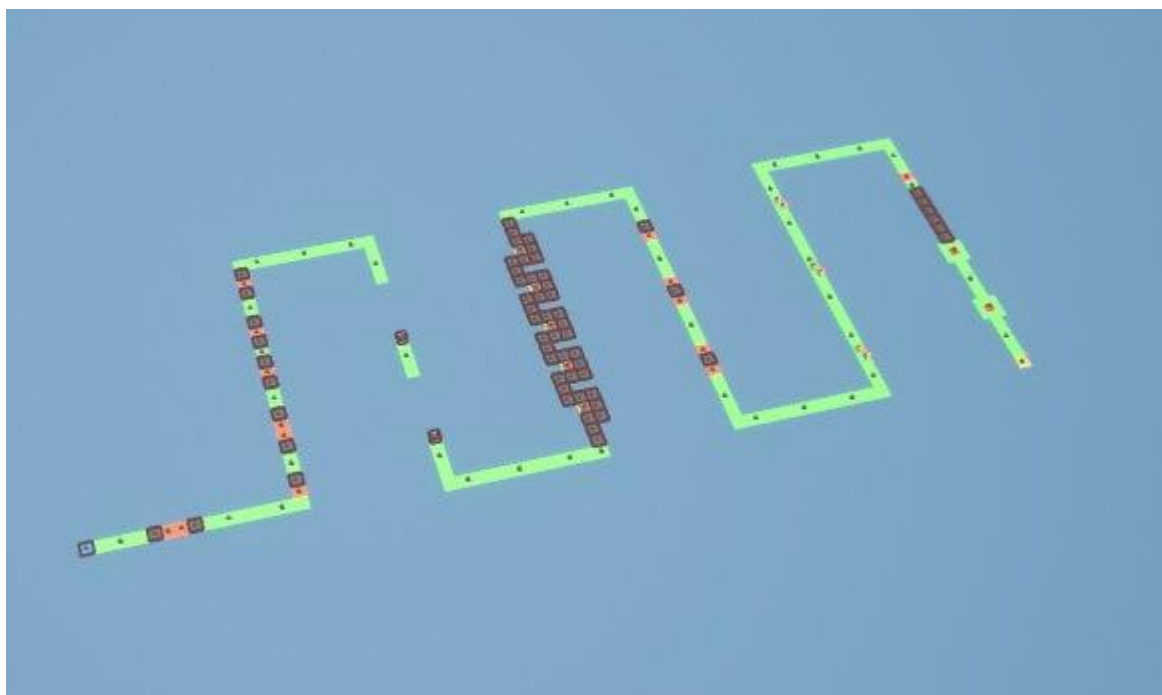


Рисунок 3.61 - Ігрова локація № 8

Ігрова локація № 9, яка наведена на рисунку 3.62 представляє собою класичний лабіринт.

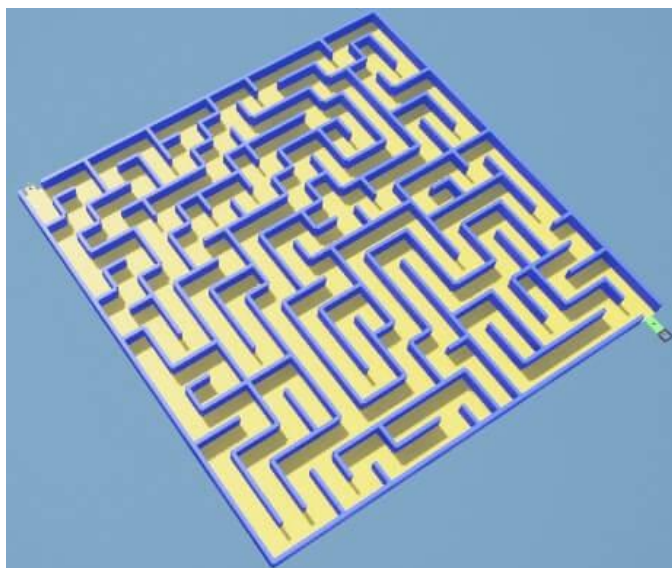


Рисунок 3.62 - Ігрова локація № 9

Ігрова локація № 10 представлена на рисунку 3.63.



Рисунок 3.63 - Ігрова локація № 10

3.5 Збірка проєкту

Зібрати проєкт в Unreal Engine можна скориставшись інструментами збірки. Для збірки проєкту під ОС Windows необхідно вибрати цільову платформу та конфігурацію збірки. Доступно чотири варіанти конфігурації збірки: DebugGame, Development, Shipping, Use Project Setting (Shipping). Перша та друга конфігурація генерують зайві файли. Конфігурація shipping не створює файли, які необхідні для налагодження та тестування гри. Інструменти для збірки проєкту представлено на рисунку 3.64. Процес збірки проєкту для ОС Windows наведено на рисунку 3.65. В результаті отримуємо файли необхідні для роботи та запуску гри, які представлено на рисунку 3.66.

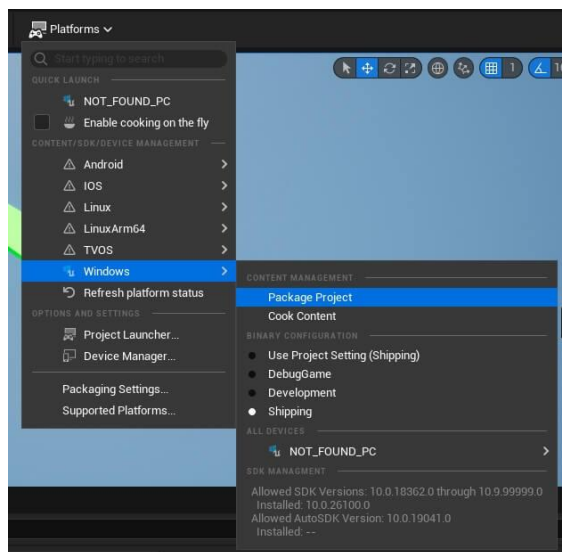


Рисунок 3.64 – Інструменти збірки проєкту

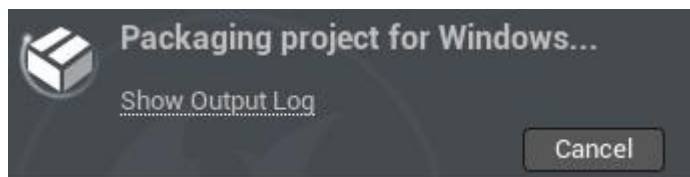
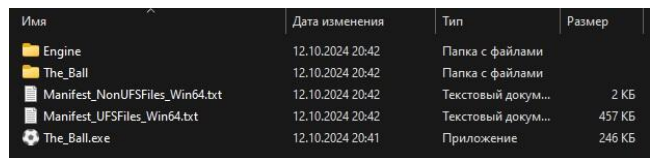


Рисунок 3.65 – Збірка гри для ОС Windows



Имя	Дата изменения	Тип	Размер
Engine	12.10.2024 20:42	Папка с файлами	
The_Ball	12.10.2024 20:42	Папка с файлами	
Manifest_NonUFSFiles_Win64.txt	12.10.2024 20:42	Текстовый докум...	2 КБ
Manifest_UFSFiles_Win64.txt	12.10.2024 20:42	Текстовый докум...	457 КБ
The_Ball.exe	12.10.2024 20:41	Приложение	246 КБ

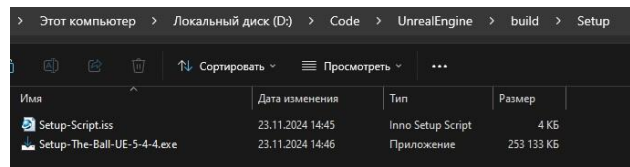
Рисунок 3.66 – Файли комп'ютерної гри

3.6 Реалізація інсталятора

Інсталятор дозволяє спростити встановлення ПЗ на комп'ютерах користувачів. Інсталятори забезпечують: створення ярликів, додавання програми до списку встановлених для зручної деінсталяції, встановлення додаткових компонентів, зміну параметрів встановлення, зменшення розміру ПЗ за допомогою алгоритмів стиснення для швидшого завантаження через мережу.

Inno Setup є однією з багатьох популярних програм для створення інсталятора. Програма надає можливість створити інсталятор для операційних систем Windows за допомогою написання скрипту або майстра створення [18].

Скориставшись програмним забезпеченням Inno Setup створено інсталятор для ОС Windows, який наведено на рисунку 3.67.



Имя	Дата изменения	Тип	Размер
Setup-Script.iss	23.11.2024 14:45	Inno Setup Script	4 КБ
Setup-The-Ball-UE-5-4-4.exe	23.11.2024 14:46	Приложение	253 133 КБ

Рисунок 3.67 – Інсталятор комп'ютерної гри для ОС Windows

Створений інсталятор можна завантажити за посиланням [19]. Відео демонстрації гри на базі Unreal Engine 5.4.4 можна побачити за посиланням [20].

4 ДОСЛІДЖЕННЯ ВПЛИВУ ІГРОВОГО ДВИГУНА НА АПАРАТНІ РЕСУРСИ

4.1 Загальні концепції

Аналогічна гра на базі Unity 2021.3.15f1 розроблена в бакалаврській роботі. Ознайомитися з процесом розробки гри на базі Unity 2021.3.15f1 можна за посиланням [21]. Гра на базі Unity 2021.3.15f1 буде використовуватися для порівняння. Завантажити гру Гра на базі Unity 2021.3.15f1 можна за посиланням [22]. Відео демонстрації гри на базі Unity 2021.3.15f1 можна побачити за посиланням [23].

Зі збільшенням потужностей обчислювальної техніки проблема нагрівання її елементів стала обмежувальним фактором [24]. Сучасні ігри часто потребують значних обчислювальних ресурсів, що викликає інтенсивне навантаження на процесор та графічний процесор. Це, у свою чергу, призводить до їх перегріву Щоб запобігти перегріву та забезпечити стабільну роботу, важливу роль відіграє моніторинг ресурсів комп'ютера.

Моніторинг ресурсів комп'ютера дозволяє відстежувати стан основних компонентів системи (процесора, оперативної пам'яті, накопичувачів, мережі) для аналізу продуктивності, запобігання перевантаженням та виявлення потенційних проблем. Для моніторингу ресурсів комп'ютера використовується спеціальне програмне забезпечення.

MSI Afterburner - безкоштовне програмне забезпечення для розгону відеокарт, апаратного моніторингу ПК. MSI Afterburner працює не лише з продукцією MSI, а й з картами інших брендів [25].

Rivatuner Statistics Server (RTSS) – утиліта для моніторингу продуктивності ПК [26]. Зазвичай використовується для виводу інформації на екран поверх вікна гри. RTSS може отримувати та виводити інформацію з таких програм як: AIDA64, MSI Afterburner, HWiNFO.

Параметри, які відображаються в моніторингу обираються в налаштуваннях MSI Afterburner в розділі моніторингу.

Після налаштування MSI Afterburner та Rivatuner Statistics Server запустивши гру побачимо показники, які були обрані для відображення. Роботу налаштованого моніторингу можна побачити на рисунку 4.1.

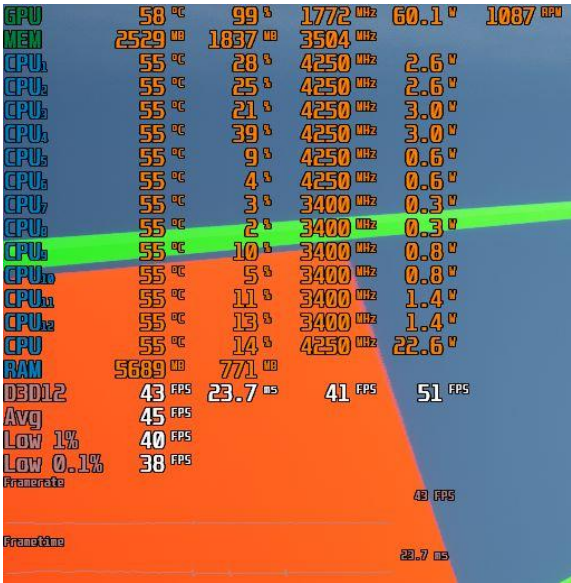


Рисунок 4.1 – Робота моніторингу в грі

Характеристики ПК, який використовується для запуску гри наведено на рисунку 4.2. На ПК встановлена ОС Windows 11 23H2.

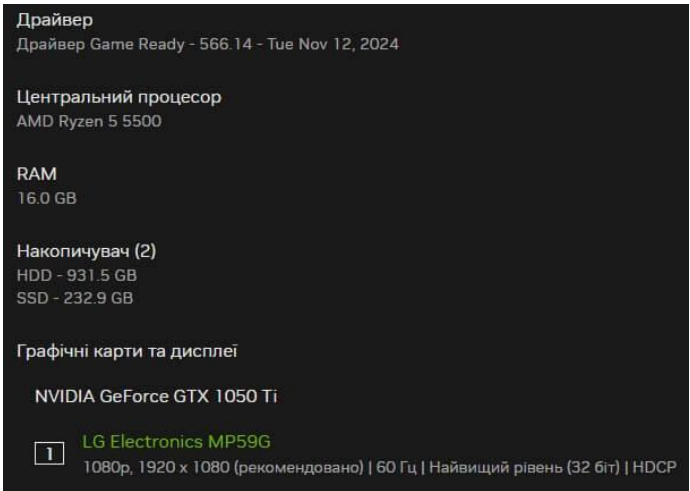


Рисунок 4.2 – Характеристики ПК

Наступним кроком є проведення моніторингу використання ресурсів ПК при різних налаштуваннях графіки гри. Моніторинг проводився з вимкненою вертикальною синхронізацією та з встановленим розширенням екрану 1920 на 1080.

4.2 Результати моніторингу ресурсів ПК

Отримані результати моніторингу гри на базі Unreal Engine 5.4.4 наведено на рисунках 4.3, 4.4, 4.5.

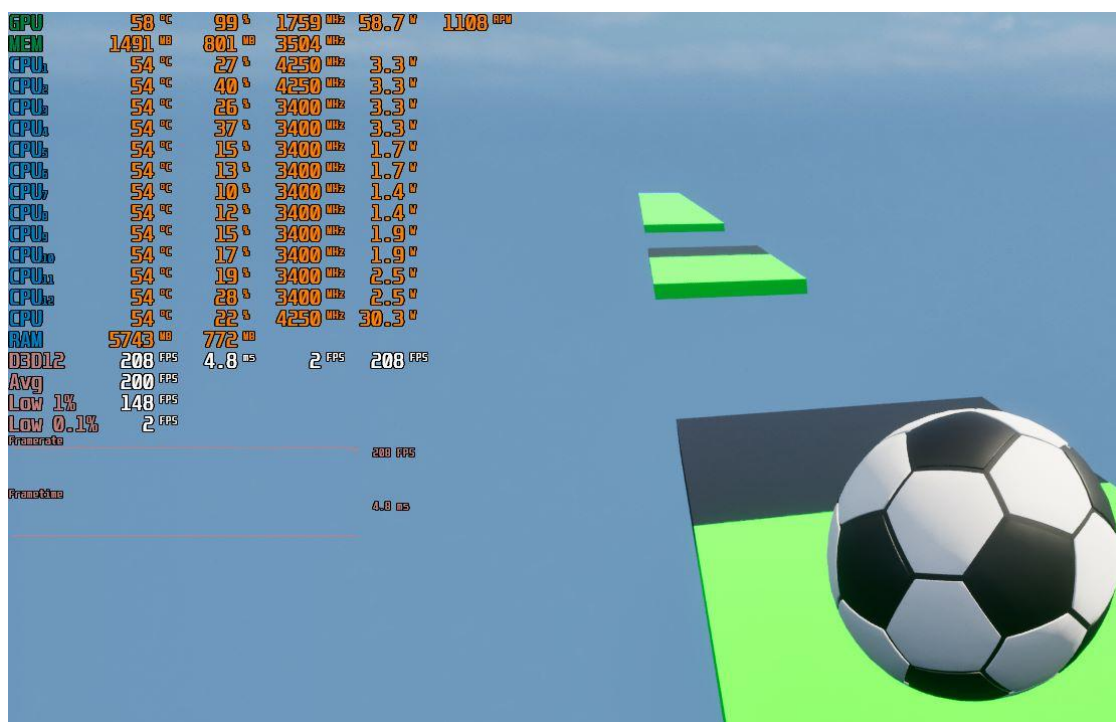


Рисунок 4.3 – Результат моніторингу ресурсів при низьких налаштуваннях графіки гри

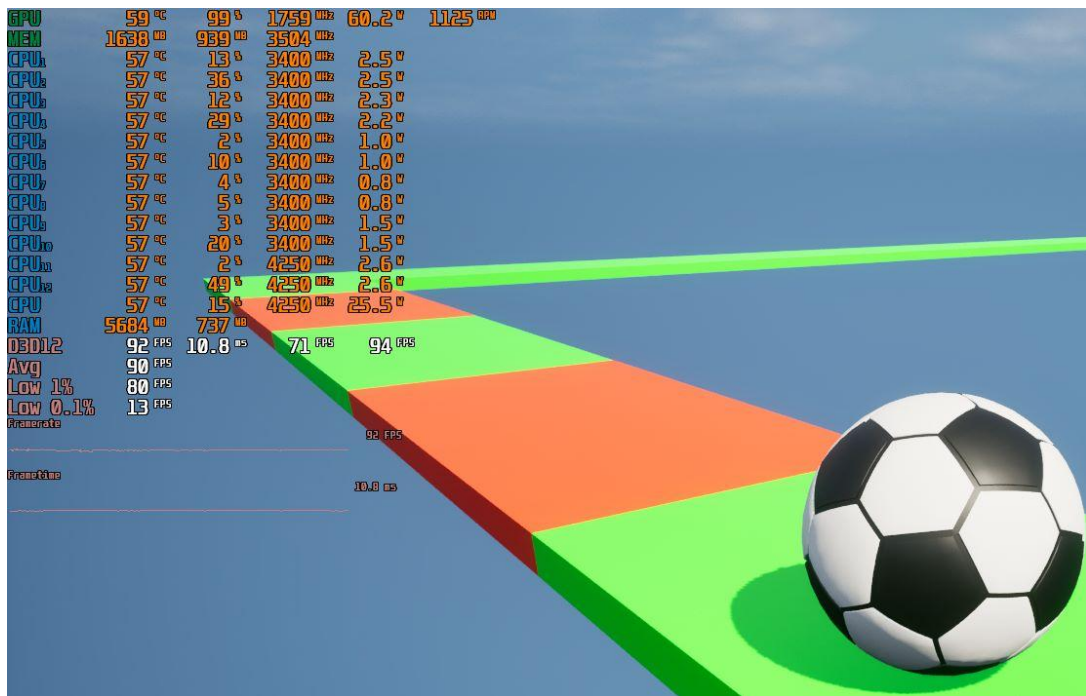


Рисунок 4.4 – Результат моніторингу ресурсів при середніх налаштуваннях графіки гри

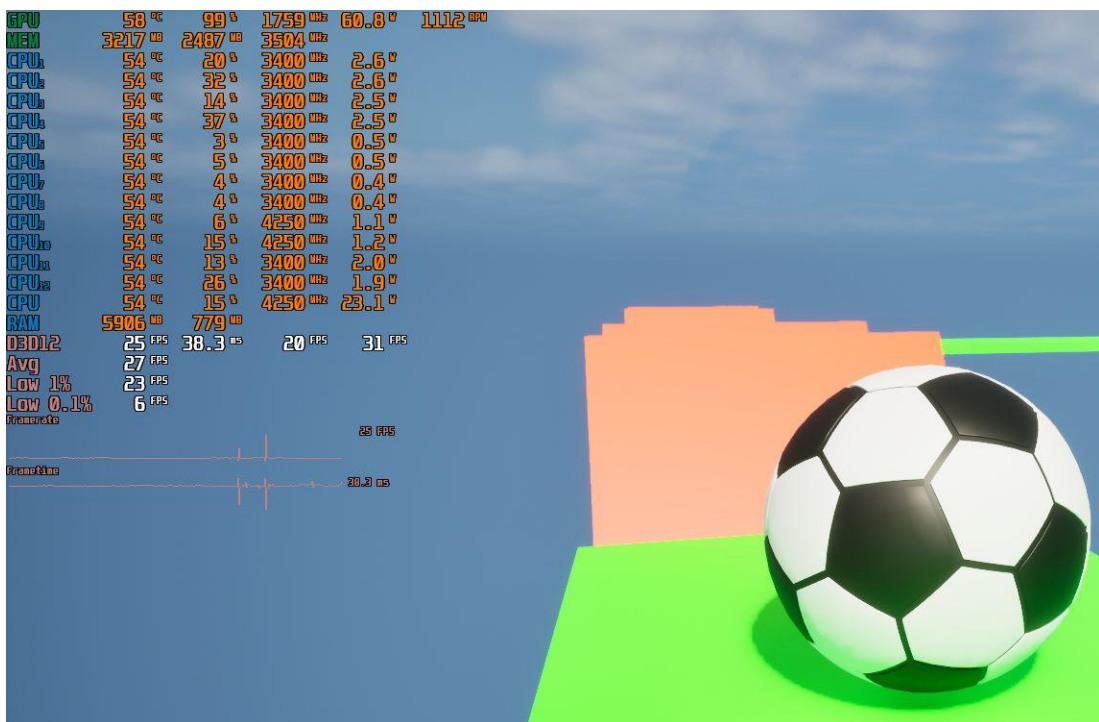


Рисунок 4.5 – Результат моніторингу ресурсів при максимальних налаштуваннях графіки гри

Отримані результати моніторингу гри на базі Unity 2021.3.15f1 наведено на рисунках 4.6, 4.7, 4.8.

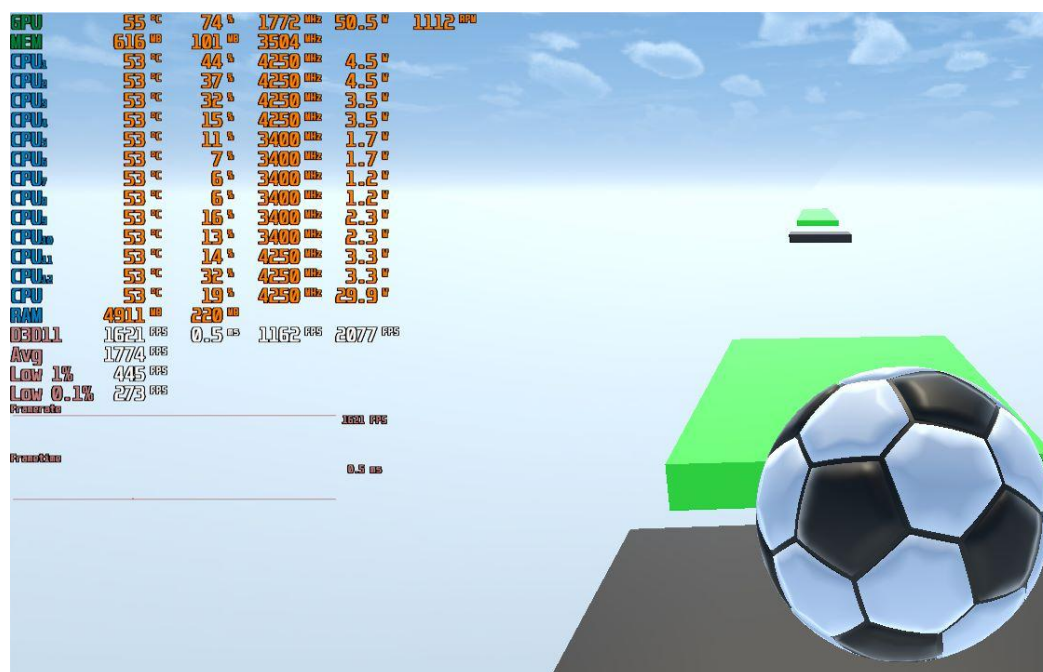


Рисунок 4.6 – Результат моніторингу ресурсів при низьких налаштуваннях графіки гри



Рисунок 4.7 – Результат моніторингу ресурсів при середніх налаштуваннях графіки гри

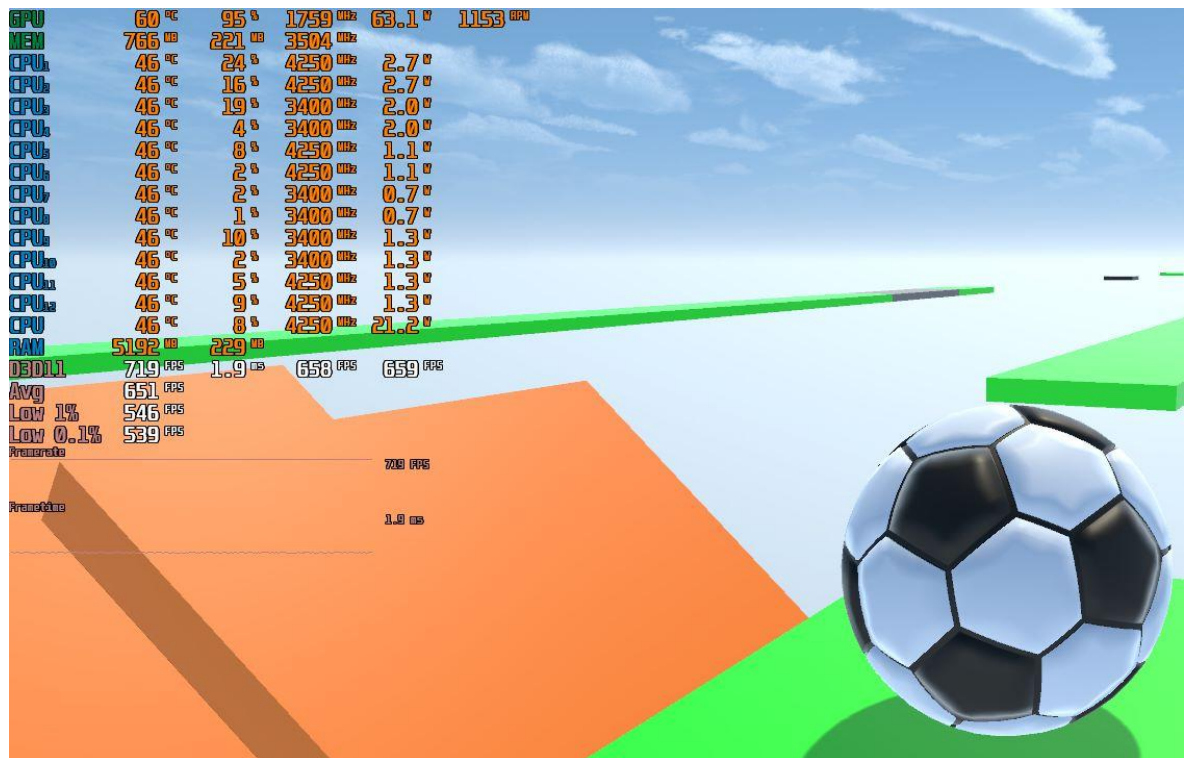


Рисунок 4.8 – Результат моніторингу ресурсів при максимальних налаштуваннях графіки гри

4.3 Аналіз отриманих результатів

4.3.1 Вплив на апаратні ресурси при низьких налаштуваннях графіки

Графік порівняння середньої кількості кадрів на секунду при низьких налаштуваннях графіки гри наведено на рисунку 4.9. На графіку видно, що гра на Unreal Engine відображає в середньому на 1574 кадрів на секунду менше ніж гра на Unity. Для комфортної гри достатньо від 30 до 60 кадрів на секунду.



Рисунок 4.9 – Графік порівняння середньої кількості кадрів на секунду при низьких налаштуваннях графіки гри

Графік порівняння навантаження на центральний процесор при низьких налаштуваннях графіки гри представлено на рисунку 4.10. На графіку видно, що гра на Unreal Engine навантажує центральний процесор на 3% більше ніж гра на Unity.

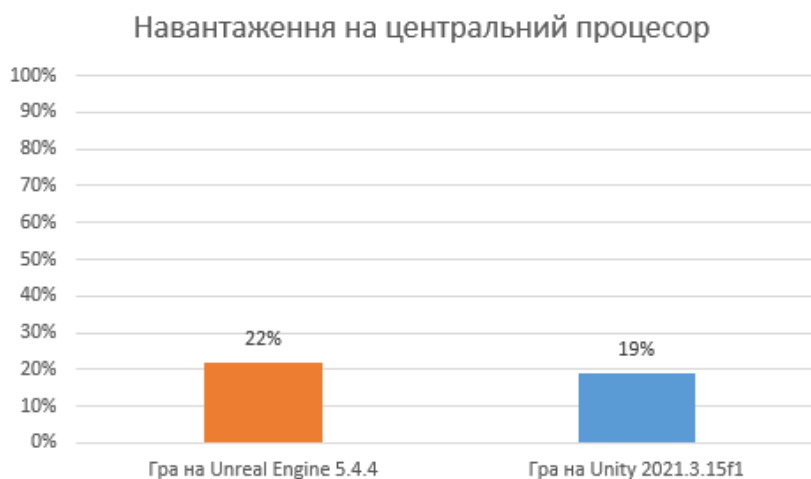


Рисунок 4.10 – Графік порівняння навантаження на центральний процесор при низьких налаштуваннях графіки гри

З графіку порівняння навантаження на відеокарту при низьких налаштуваннях графіки гри наведеному на рисунку 4.11 видно, що гра на Unreal Engine навантажує відеокарту на 25% більше ніж гра на Unity.



Рисунок 4.11 – Графік порівняння навантаження на відеокарту при низьких налаштуваннях графіки гри

Графік порівняння використання грою відеопам'яті при низьких налаштуваннях графіки гри наведено на рисунку 4.12. З графіку видно, що гра на Unreal Engine використовує на 700 мегабайт більше відеопам'яті ніж гра на Unity.



Рисунок 4.12 – Графік порівняння використання грою відеопам'яті при низьких налаштуваннях графіки гри

Графік порівняння загального використання відеопам'яті при низьких налаштуваннях графіки гри представлено на рисунку 4.13.

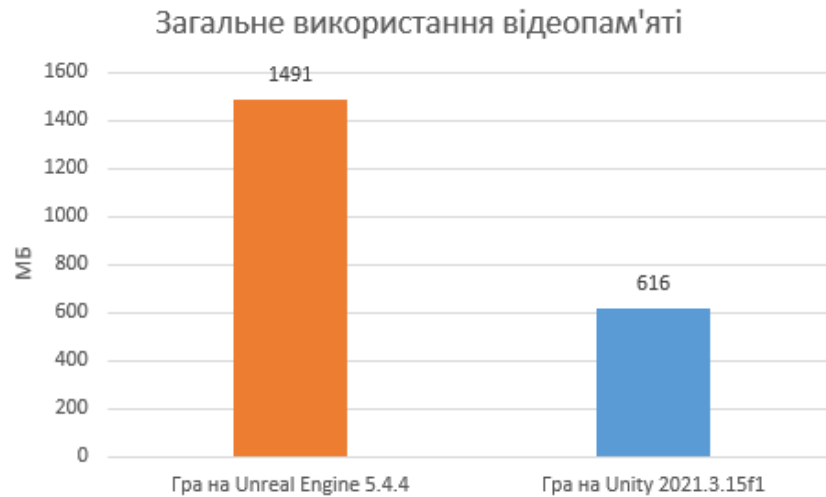


Рисунок 4.13 – Графік порівняння загального використання відеопам'яті при низьких налаштуваннях графіки гри

З графіку представленому на рисунку 4.14 видно, що гра на Unreal Engine споживає на 552 мегабайти оперативної пам'яті більше ніж гра на Unity.

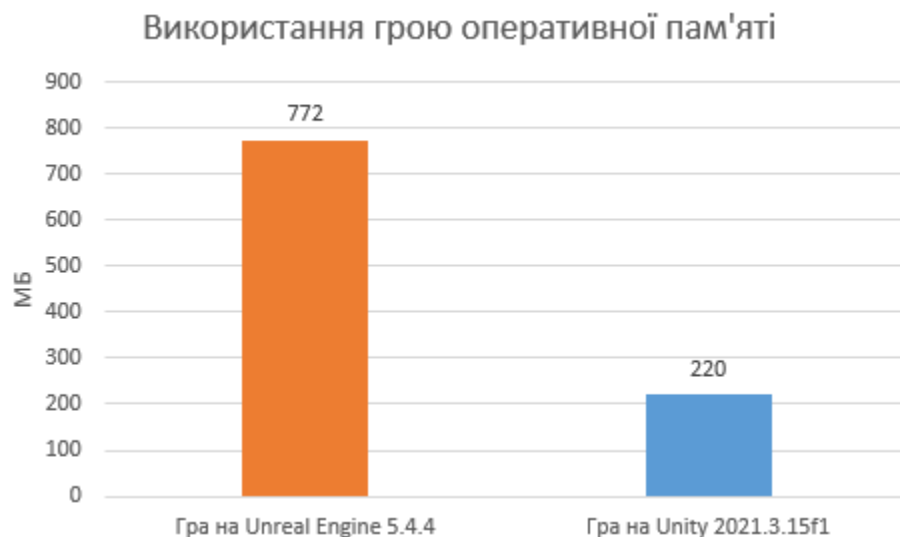


Рисунок 4.14 – Графік порівняння використання грою оперативної пам'яті при низьких налаштуваннях графіки гри

Графік загального використання оперативної пам'яті при низьких налаштуваннях графіки гри наведено на рисунку 4.15.

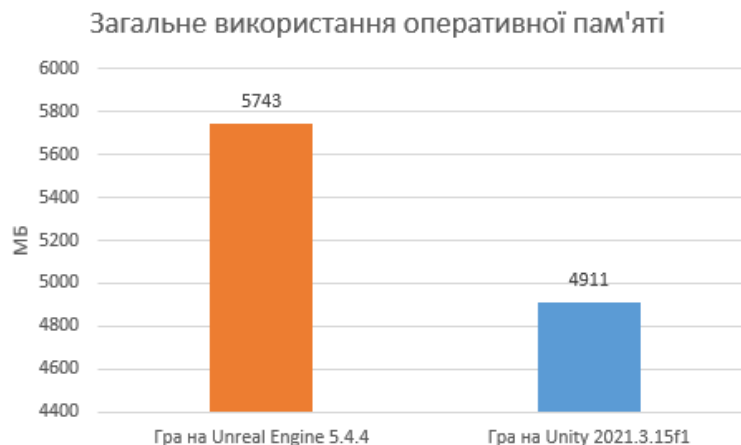


Рисунок 4.15 – Графік порівняння загального використання оперативної пам'яті при низьких налаштуваннях графіки гри

4.3.2 Вплив на апаратні ресурси при середніх налаштуваннях графіки

Графік порівняння середньої кількості кадрів на секунду при середніх налаштуваннях графіки гри наведено на рисунку 4.16. На графіку видно, що гра на Unreal Engine відображає в середньому на 1106 кадрів на секунду менше ніж гра на Unity.



Рисунок 4.16 – Графік порівняння середньої кількості кадрів в секунду грою при середніх налаштуваннях графіки гри

Графік порівняння навантаження на центральний процесор при середніх налаштуваннях графіки гри представлено на рисунку 4.17. На графіку видно, що гра на Unreal Engine навантажує центральний процесор на 2% більше ніж гра на Unity.

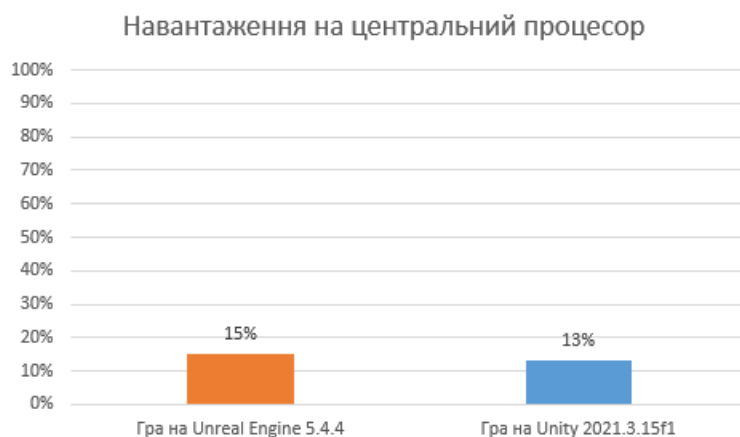


Рисунок 4.17 – Графік порівняння навантаження на центральний процесор при середніх налаштуваннях графіки гри

З графіку порівняння навантаження на відеокарту при середніх налаштуваннях графіки гри наведеному на рисунку 4.18 видно, що гра на Unreal Engine навантажує відеокарту на 12% більше ніж гра на Unity.



Рисунок 4.18 – Графік порівняння навантаження на відеокарту при середніх налаштуваннях графіки гри

Графік порівняння використання грою відеопам'яті при середніх налаштуваннях графіки гри наведено на рисунку 4.19. З графіку видно, що гра на Unreal Engine використовує на 800 мегабайт більше відеопам'яті ніж гра на Unity.



Рисунок 4.19 – Графік порівняння використання грою відеопам'яті при середніх налаштуваннях графіки гри

Графік загального використання відеопам'яті при середніх налаштуваннях графіки гри представлено на рисунку 4.20.

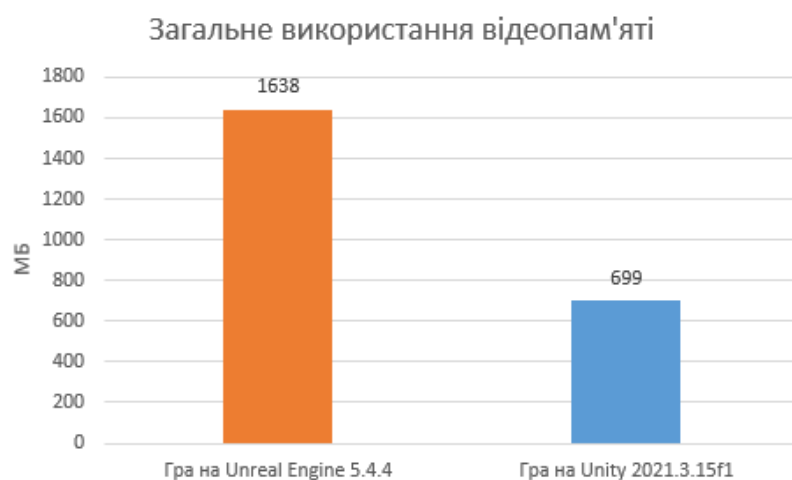


Рисунок 4.20 – Графік порівняння загального використання відеопам'яті при середніх налаштуваннях графіки гри

З графіку представленому на рисунку 4.21 видно, що гра на Unreal Engine споживає на 513 мегабайти оперативної пам'яті більше ніж гра на Unity.

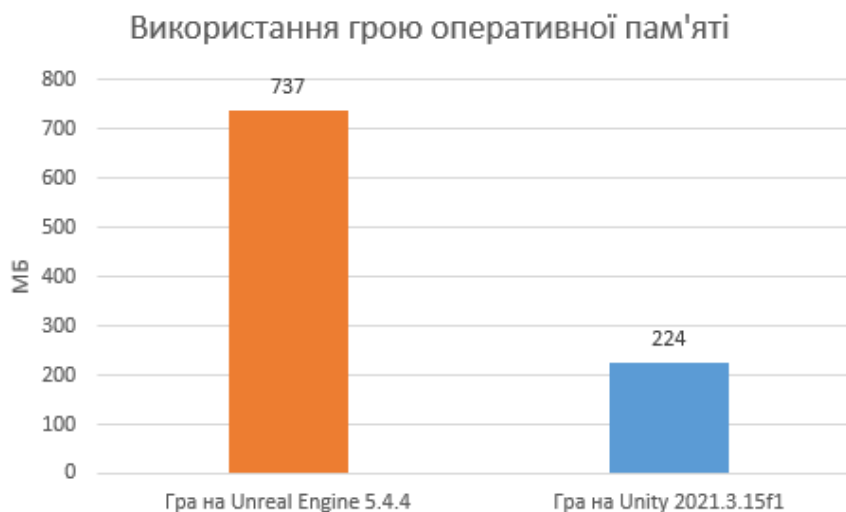


Рисунок 4.21 – Графік порівняння використання грою оперативної пам'яті при середніх налаштуваннях графіки гри

Графік загального використання оперативної пам'яті при середніх налаштуваннях графіки гри наведено на рисунку 4.22.

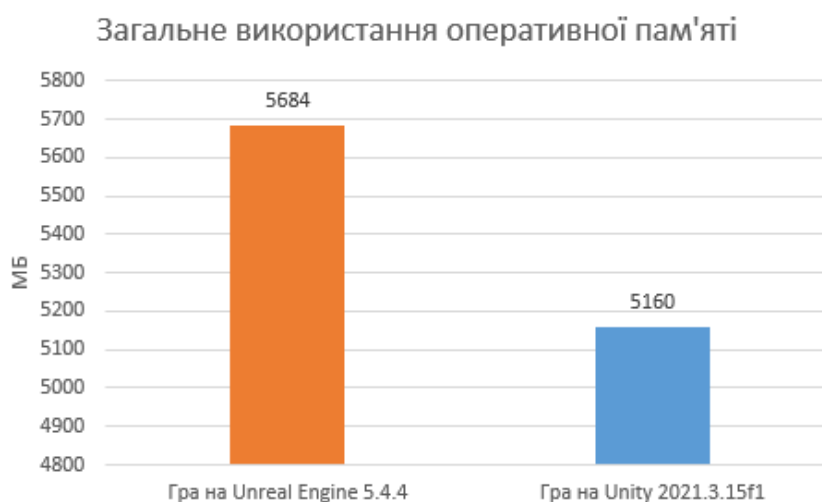


Рисунок 4.22 – Графік порівняння загального використання оперативної пам'яті при середніх налаштуваннях графіки гри

4.3.3 Вплив на апаратні ресурси при максимальних налаштуваннях графіки

Графік порівняння середньої кількості кадрів на секунду при максимальних налаштуваннях графіки гри наведено на рисунку 4.23. На графіку видно, що гра на Unreal Engine відображає в середньому на 624 кадри на секунду менше ніж гра на Unity.



Рисунок 4.23 – Графік порівняння середньої кількості кадрів на секунду при максимальних налаштуваннях графіки гри

Графік порівняння навантаження на центральний процесор при максимальних налаштуваннях графіки гри представлено на рисунку 4.24. На графіку видно, що гра на Unreal Engine навантажує центральний процесор на 7% більше ніж гра на Unity.

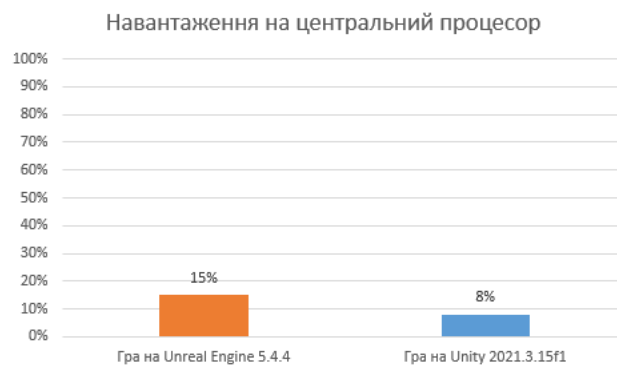


Рисунок 4.24 – Графік порівняння навантаження на центральний процесор при максимальних налаштуваннях графіки гри

З графіку порівняння навантаження на відеокарту при максимальних налаштуваннях графіки гри наведеному на рисунку 4.25 видно, що гра на Unreal Engine навантажує відеокарту на 4% більше ніж гра на Unity.

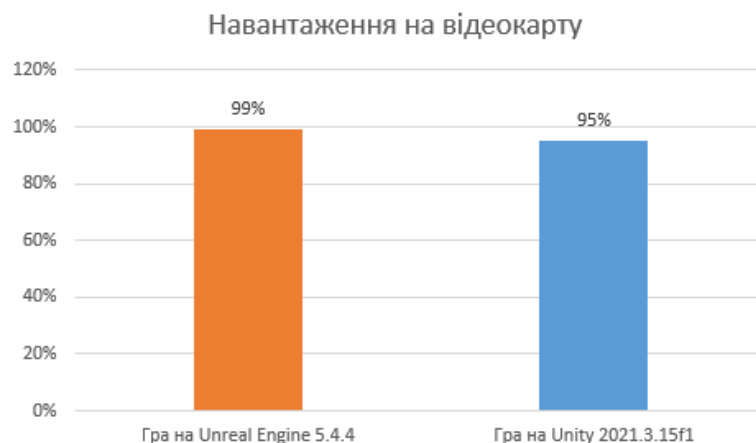


Рисунок 4.25 – Графік порівняння навантаження на відеокарту при максимальних налаштуваннях графіки гри

Графік порівняння використання грою відеопам'яті при максимальних налаштуваннях графіки гри наведено на рисунку 4.26. З графіку видно, що гра на Unreal Engine використовує на 2266 мегабайт більше відеопам'яті ніж гра на Unity.



Рисунок 4.26 – Графік порівняння використання грою відеопам'яті при максимальних налаштуваннях графіки гри

Графік загального використання відеопам'яті при максимальних налаштуваннях графіки гри представлено на рисунку 4.27.

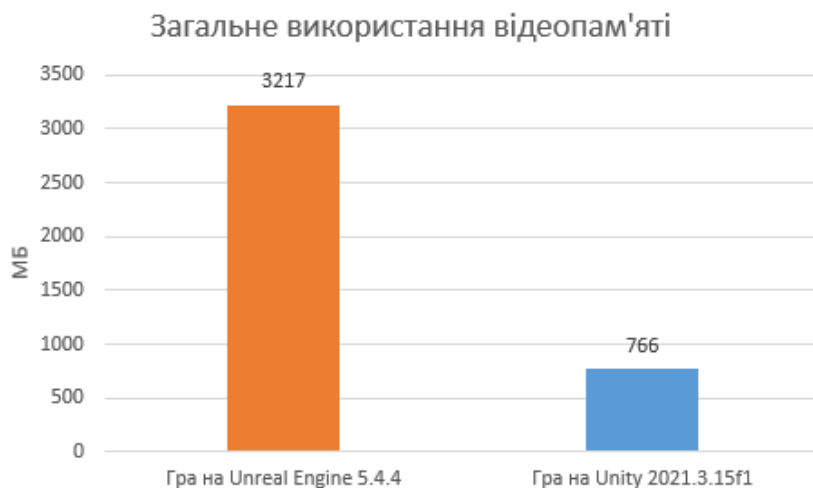


Рисунок 4.27 – Графік порівняння загального використання відеопам'яті при максимальних налаштуваннях графіки гри

З графіку представленому на рисунку 4.28 видно, що гра на Unreal Engine споживає на 550 мегабайти оперативної пам'яті більше ніж гра на Unity.

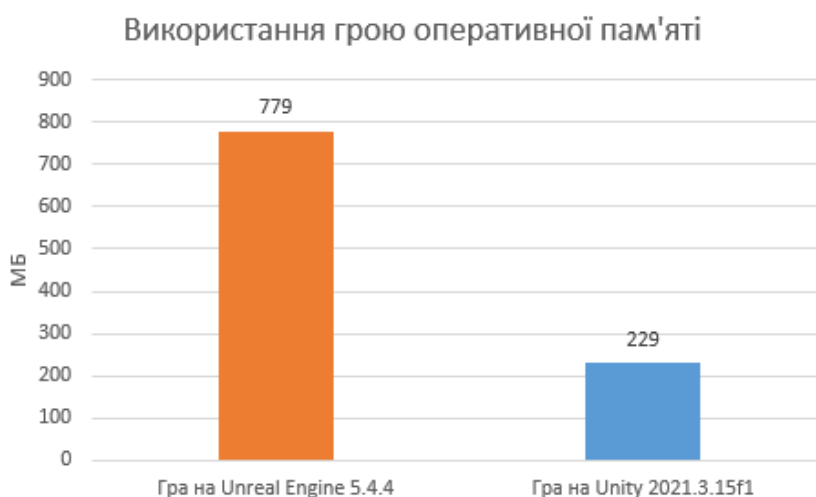


Рисунок 4.28 – Графік порівняння використання грою оперативної пам'яті при максимальних налаштуваннях графіки гри

Графік загального використання оперативної пам'яті при максимальних налаштуваннях графіки гри наведено на рисунку 4.29.

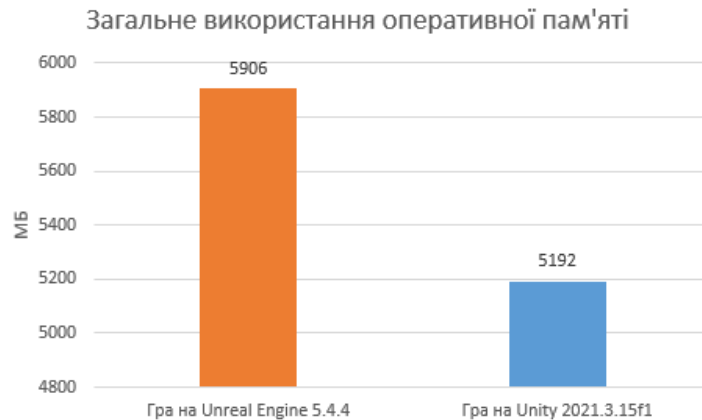


Рисунок 4.29 – Графік порівняння загального використання оперативної пам'яті при максимальних налаштуваннях графіки гри

4.3.4 Аналіз розміру файлів гри на накопичувачі

Розмір файлів гри є важливим показником. Сучасні ігри займають багато місця, наприклад комп'ютерна гра S.T.A.L.K.E.R. 2: Heart of Chernobyl від української студії GSC Game World займає приблизно 160 гігабайтів простору на накопичувачі.

З графіку представленому на рисунку 4.30 видно, що розмір файлів гри на Unreal Engine на 312 мегабайт більше ніж на Unity.

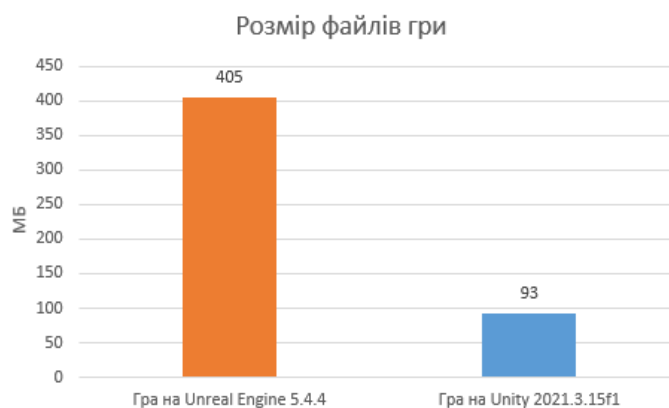


Рисунок 4.30 - Графік порівняння розміру файлів гри

ВИСНОВКИ

Таким чином у магістерській роботі розроблена комп'ютерна гра у жанрі 3D-платформера на базі ігрового двигуна Unreal Engine 5.4.4 та досліджено вплив реалізованої гри на різних ігрових рушіях на апаратні ресурси комп'ютера.

Проведене дослідження показало, що гра на Unreal Engine 5.4.4 споживає більше ресурсів ПК порівняно з грою на Unity 2021.3.15f1. Візуальна якість гри на Unreal Engine 5.4.4 була кращою порівняно з Unity 2021.3.15f1.

Слід відмітити, що для запуску гри на базі Unreal Engine 5.4.4 необхідна відеокарта, яка підтримує мінімум Shader model 5.0 та DirectX 11, а краще Shader model 6.0 та DirectX 12. Через високі вимоги до відеокарти та відеопам'яті менше користувачів зможуть запустити гру на базі Unreal Engine 5.4.4.

У підсумку, вибір ігрового двигуна має великий вплив на споживання ресурсів ПК і кількість користувачів, які зможуть запустити гру. Тому розробники ігор часто шукають баланс між якістю графіки та вимогами до апаратних ресурсів.

Під час виконання магістерської роботи виконано:

- аналіз предметної області;
- аналіз та порівняння ігрових двигунів;
- розробка концепції гри;
- розробка інтерфейсу гри;
- розробка ігрових об'єктів та персонажа;
- розробка ігрових локацій;
- дослідження впливу ігрового двигуна на апаратні ресурси;
- аналіз отриманих результатів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Основи комп'ютерних ігор та ігрових програм : довідник модуля. / С. О. Нікітін, Л. О. Нікітіна. – Х. : «Друкарня Мадрид», 2018. – 138 с.
2. Організація PEGI URL: <https://pegi.info/uk/page/pegi-organisation> (дата звернення: 05.09.2024)
3. Що означає маркування? URL: <https://pegi.info/uk/what-do-the-labels-mean> (дата звернення: 05.09.2024)
4. Логотип рейтингу PEGI URL: <https://www.pinterest.com/pin/284852745152092625/> (дата звернення: 05.09.2024)
5. Організація ESRB URL: <https://www.esrb.org/about/> (дата звернення: 05.09.2024)
6. Довідник рейтингу ESRB URL: <https://www.esrb.org/ratings-guide/> (дата звернення: 05.09.2024)
7. Офіційний сайт USK URL: <https://usk.de/en/> (дата звернення: 05.09.2024)
8. Віковий рейтинг USK URL: <https://usk.de/en/the-usk-age-ratings/> (дата звернення: 05.09.2024)
9. Офіційний сайт CERO URL: <https://www.cero.gr.jp/en/publics/index/17/> (дата звернення: 05.09.2024)
10. Логотип рейтингу CERO URL: <https://www.cero.gr.jp/en/> (дата звернення: 05.09.2024)
11. Касьян К. М. Порівняння сучасних ігрових двигунів / К. М. Касьян, М. М. Касьян, Є. О. Книрик // Тиждень науки-2023. Факультет комп'ютерних наук і технологій. Тези доповідей науково-технічної конференції, Запоріжжя, 24-28 квітня 2023 р. / Редкол. : В. Шаломєєв (відпов. ред.) Електрон. дані. – Запоріжжя : НУ «Запорізька політехніка», 2023.

12. Unity URL: <https://unity.com/products/unity-engine> (дата звернення: 07.09.2024)
13. Unreal Engine URL: <https://www.unrealengine.com/en-US/features> (дата звернення: 07.09.2024)
14. Godot Engine URL: <https://docs.godotengine.org/en/stable/about/introduction.html> (дата звернення: 07.09.2024)
15. Репозиторій Redot Engine URL: <https://github.com/Redot-Engine/redot-engine> (дата звернення: 07.09.2024)
16. Документація Redot Engine URL: <https://docs.redotengine.org/en/stable/#> (дата звернення: 07.09.2024)
17. Unreal Engine documentation URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/blueprints-visual-scripting-in-unrealengine?application_version=5.4 (дата звернення: 08.09.2024)
18. Головна сторінка Inno Setup URL: <https://jrsoftware.org/isinfo.php> (дата звернення: 20.10.2024)
19. Репозиторій гри на базі Unreal Engine URL: <https://github.com/Knirik-Yegor/Build-The-Ball-Unreal-Engine> (дата звернення: 01.10.2024)
20. Відео демонстрації гри на базі Unreal Engine 5.4.4 URL: <https://youtu.be/-DDOsbDYkCk> (дата звернення: 02.10.2024)
21. Розробка комп'ютерної гри мовою C# за допомогою середовища Unity URL: <https://eir.zp.edu.ua/items/1d1d37be-9f76-402c-8531-4985f0978f23> (дата звернення: 06.11.2024)
22. Репозиторій гри на базі Unity URL: <https://github.com/Knirik-Yegor/Build-The-Ball> (дата звернення: 06.11.2024)
23. Відео демонстрації гри на базі Unity 2021.3.15f1 URL: <https://youtu.be/vCo4lLlscDc> (дата звернення: 06.11.2024)

24. Книрик Є.О., Тягунова М.Ю. Модуль активного охолодження Airjet MINI. Тиждень науки-2024. Факультет комп'ютерних наук і технологій : наук.-техн. конф., 15-19 квітня 2024 р.: тези доп. – / Редкол. : Вадим ШАЛОМЄЄВ (відпов. ред.) Електрон. дані. – Запоріжжя : НУ «Запорізька політехніка», 2024.

25. Офіційний сайт MSI Afterburner URL: <https://www.msi.com/Landing/afterburner/graphics-cards> (дата звернення: 07.11.2024)

26. Офіційний сайт Rivatuner Statistics Server URL: <https://www.guru3d.com/download/rtss-rivatuner-statistics-server-download/> (дата звернення: 07.11.2024)