Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет інформаційних радіотехнологій та технічного захисту інформації
(повна назва)

Кафедра медіаінженерії та інформаційних радіоелектронних систем
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти *другий (магістерський)*
(позначення документа)

Розробка ігрового додатку "Ритм" на ігровому рушії Unreal Engine 5

(тема)

Виконав:
студент 2 курсу, групи *СТМм-22-1*
Володимир ЛАБУНСЬКИЙ
(прізвище, ініціали)

Спеціальність *171 Електроніка*
(код і повна назва спеціальності)

Тип програми *освітньо-професійна*
(освітньо-професійна або освітньо-наукова)

Освітня програма *Системи, технології і комп'ютерні засоби мультимедіа*
(повна назва освітньої програми)

Керівник *ас. Вікторія КОЛІСНИК*
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ Володимир КАРТАШОВ
(підпис) (прізвище, ініціали)

2023 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційних радіотехнологій та технічного захисту інформації
Кафедра Медіаінженерії та інформаційних радіоелектронних систем

Рівень вищої освіти другий (магістерський)
Спеціальність 171 Електроніка
(код і повна назва)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма "Системи, технології і комп'ютерні засоби мультимедіа"

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Студентові *Лабунському Володимиру Валентиновичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *Розробка ігрового додатку "Ритм" на ігровому рушії Unreal Engine 5*
затверджена наказом по університету від " *20* " *11* *2023 р. №* *1371 СТ*
2. Термін подання студентом роботи *08.01.2024 р.*
3. Вихідні дані до проекту (роботи)
1. Розробити ігровий додаток "Ритм" на ігровому рушії Unreal Engine 5
2. Розробити 3D-моделі та 2D-елементи оточення і інтерфейсу для подальшого використання в ігровому додатку "Ритм"
3. Розробити музичне наповнення ігрового додатку "Ритм"

4. Перелік питань, що потрібно опрацювати в роботі
ВСТУП
1. Аналітичний огляд існуючих ритм ігор
2. Вибір програмного забезпечення для створення ритм гри
3. Розробка гри
ВИСНОВКИ
ПЕРЕЛІК ПОСИЛАНЬ
ДОДАТКИ
5. Перелік графічного матеріалу із зазначенням обов'язкових креслеників, схем, плакатів, комп'ютерних ілюстрацій
1. Постановка задачі; 2. Види ритм ігор; 3. Ігрові рушії; 4. Програми для роботи з 3D графікою; 5. Програми для роботи з 2D графікою; 6. Програми для створення звуку; 7. Головний герой; 8. Головний злодій; 9. Концепт та модель зброї; 10. Елементи оточення; 11. Елементи інтерфейсу; 12. Готовий проект; 13. Висновки.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Аналітичний огляд існуючих ритм ігор	21.11.23–28.11.23	
2.	Вибір програмного забезпечення для створення ритм гри	21.11.23–28.11.23	
3.	Розробка гри	28.11.23–07.12.23	
4.	Графічна частина роботи	07.12.23–08.12.24	
5.	Перевірка керівником	07.12.23-08.12.24	
6.	Перевірка на академічний плагіат	08.12.23-09.12.24	
7.	Графічна частина роботи	07.12.23–08.12.24	

Дата видачі завдання *21.11.2023 р.*

Студент _____ *Володимир ЛАБУНСЬКИЙ*
(підпис)

Керівник роботи _____ *ас. Вікторія КОЛІСНИК*
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи має: 99 с., 141 рис., 0 табл., 2 додатки, 61 джерело.

КОМП’ЮТЕРНА ГРА, РОЗРОБКА, РУШІЙ, 3D-ГРАФІКА, UNREAL ENGINE, ДИЗАЙН РІВНЯ, ОБРОБКА ЗВУКА, АНІМАЦІЯ

Об'єкт дослідження – створення музично-ритмічного ігрового контенту.

Предмет дослідження – розробка ігор на рушії Unreal Engine 5.

Мета кваліфікаційної роботи – розробити комп’ютерну музично-ритмічну гру «Ритм» на ігровому рушії Unreal Engine 5

Методи дослідження – дослідженню та реалізації концепції музично-ритмічної гри на основі новітнього графічного движка Unreal Engine 5.

У даній роботі наведено дослідження та реалізацію концепції музично-ритмічної гри на основі новітнього графічного рушія Unreal Engine 5. Робота вивчає ключові аспекти розробки ігор, зосереджуючись на створенні захоплюючого та емоційного геймплею.

Основні завдання дослідження включають вивчення можливостей інструментів Unreal Engine 5, аналіз сучасних тенденцій у розробці ігор, а також розробка прототипу музично-ритмічної гри "Ритм". Робота аналізує аспекти графіки, анімації, звукового дизайну та інтерактивних механік, що є важливими для створення сучасних ігор.

Робота підтверджує важливість вивчення та впровадження новітніх технологій у розробці комп'ютерних ігор, а також демонструє можливості Unreal Engine 5 у створенні захоплюючих та емоційних ігрових досвідів.

ABSTRACT

The explanatory note of the qualification paper has: 99 p., 141 figures, 0 tables, 2 appendices, 61 sources.

PC GAME, DEVELOPMENT, ENGINE, 3D GRAPHICS, UNREAL ENGINE, LEVEL DESIGN, SOUND PROCESSING, ANIMATION

The object of research is a computer game.

The subject of the research is a computer game powered by unreal engine 5.

The purpose of the qualification work is to develop a computer "rhythm" game using the unreal engine 5 game engine

Research methods - research and implementation of the concept of a musical and rhythmic game based on the latest graphics engine Unreal Engine 5.

This work presents the research and implementation of the concept of a music-rhythmic game based on the latest graphics engine Unreal Engine 5. The work studies the key aspects of game development, focusing on the creation of exciting and emotional gameplay.

The main tasks of the research include the study of the capabilities of Unreal Engine 5 tools, the analysis of modern trends in game development, as well as the development of a prototype of the music-rhythmic game "Rhythm". The paper analyzes the aspects of graphics, animation, sound design and interactive mechanics that are essential to the creation of modern games.

The work confirms the importance of studying and implementing the latest technologies in the development of computer games, and also demonstrates the capabilities of Unreal Engine 5 in creating exciting and emotional gaming experiences.

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

2D – двовимірний опис об’єкта;

3D – тривимірний опис об’єкта;

UE - Unreal Engine;

Ассет, asset - цифровий об'єкт, який представляє частину ігрового контенту і має деякі властивості;

Бітмапи – елемент на який потрібно натиснути в особливий час, частіше за все під ритм музики;

Блупрінт, blueprint – система програмування в ігровому рушії Unreal Engine;

БПМ, BPM - кількість ударів або тактів в музичній композиції, яку можна обчислити протягом однієї хвилини;

Ріггінг - процес створення внутрішньої структури або "скелету" для 3D-моделі, який дозволяє контролювати рухи цієї моделі;

Семпл - це фрагмент аудіозапису, який використовується в новому контексті чи композиції;

Скін - зовнішній вигляд персонажа, предмета чи об'єкта у грі;

Скрипт, script – це невелика програма, яка містить послідовність дій, створених для автоматичного виконання завдання;

Спрайт, sprite – двовимірне зображення в комп’ютерній графіці;

Тайл, tile – поодинока графічна плитка, що є частиною цілісного зображення;

Таймлайн, timeline — створена у хронологічному порядку подієва стрічка;

ШІ – штучний інтелект.

ЗМІСТ

ВСТУП

У сучасному світі комп'ютерні ігри завойовують все більшу популярність, що відображається не лише у зростанні числа гравців, але й у технічному та естетичному рівні розробки. Ігрова індустрія постійно зростає, вдосконалюючи графіку, механіку та взаємодію гравця з ігровим світом. В рамках цього контексту, кваліфікаційна робота присвячена розробці ігрового додатку "Ритм" на основі ігрового рушія Unreal Engine 5.

Ця робота має на меті дослідити та реалізувати концепцію музично-ритмічної гри, яка використовує потужні можливості нового покоління графічного рушія Unreal Engine 5. Основний акцент роботи робиться на створенні інноваційного геймплею, який об'єднує в собі відмінну графіку, анімацію та звуковий дизайн, забезпечуючи гравцям захоплюючий та емоційний досвід.

У рамках цього дослідження буде вивчено ключові аспекти розробки гри на Unreal Engine 5, включаючи створення візуальних ефектів, анімації персонажів, звукового супроводу та інтерактивних механік. Крім того, кваліфікаційна робота буде включати аналіз сучасних тенденцій у розробці ігор та їх вплив на створення інноваційних ігрових рішень.

Завданням даної роботи є розробка прототипу гри "Ритм", який демонструє потенціал Unreal Engine 5 в сфері музично-ритмічних ігор, а також внесення вкладу в розуміння та оптимізацію процесу розробки ігор на основі сучасних технологій та методів.

У підсумку, кваліфікаційна робота має на меті не лише реалізувати конкретний ігровий проект, але й вивчити, аналізувати та адаптувати найновіші досягнення в сфері розробки ігор для створення захоплюючих, інноваційних та технологічно розвинутих ігрових світів.

1 АНАЛІТИЧНИЙ ОГЛЯД ІСНУЮЧИХ РИТМ ІГОР

Жанр ритм ігор - це вид відеоігор, в яких гравцеві доводиться виконувати певні дії в синхронізації з музичним ритмом. Ці ігри зазвичай містять різноманітні музичні композиції, а ігровий процес будується навколо точного слідування ритму, який представлений у музиці.

Основні елементи ритм-ігор включають в себе:

- ноти чи маркери. Під час гри на екрані з'являються ноти, маркери або інші візуальні елементи, що вказують на необхідність виконання певної дії в певний момент часу;
- музична підтримка. Основним компонентом геймплею є музична композиція. Ритм ігри часто пропонують широкий вибір треків різних стилів та напрямків;
- оцінка продуктивності. Гравці отримують оцінку за точність і своєчасність своїх дій. Зазвичай використовуються різні ранги, такі як "прекрасно", "добре", "задовільно" або відсоток точності слідкування ритму;
- рівні складності. Ігри можуть пропонувати різні рівні складності, починаючи від простого і поступово ускладнюючись, щоб задовольнити різні рівні гравців;
- соціальний аспект. Деякі ритм-ігри мають онлайн режими, що дозволяє гравцям змагатися або співпрацювати з іншими гравцями. Це створює соціальний аспект гри, де люди можуть об'єднатися навколо спільного інтересу до музики та геймінгу.

Ритм ігри стають популярними завдяки комбінації цікавого геймплею, візуального та звукового оформлення, а також широкого вибору музичних треків.

В цьому розділі ми розглянемо приклади популярних ритм-ігор, які дадуть нам більше розуміння в жанрі.

1.1 Аналіз гри PaRappa the Rapper

PaRappa the Rapper - це музична ритм гра, яка вийшла в 1996 році для консолі Sony PlayStation. Гра набула великої популярності завдяки своєму унікальному стилю та цікавому геймплею (рис 1.1).

Рисунок 1.1 – Гра PaRappa the Rapper [1]

Основний персонаж гри - Параппа, антропоморфний пес, який має здібності говорити та робити різноманітні речі. Головна мета гравця - допомогти Параппі подолати різноманітні життєві труднощі, використовуючи ритмічні команди.

Гра вирізняється такими особливостями як музичні етапи. Кожен рівень гри представляє собою певну сцену або історію, яку гравець повинен пройти, виконуючи ритмічні команди за допомогою кнопок контролера.

Унікальний візуальний стиль: "PaRappa the Rapper" визначається своїм яскравим та карикатурним візуальним стилем, що надає грі неповторності. Гравець слідує за історією Параппи та інших персонажів, розкриваючи різні епізоди їхнього життя.

Гра пропонує різноманітні жанри музики, включаючи хіп-хоп, реггі, рок і інші. Та відзначається нестандартним підходом до геймплею і привертає увагу

гравців своєю неповторною атмосферою та оригінальністю. Гра стала класикою в жанрі ритм-ігор та залишається улюбленою серед шанувальників відеоігор.

1.2 Аналіз гри Guitar Hero

Guitar Hero - це серія музичних ритм-ігор, яка стала великим хітом у своєму жанрі. Вперше випущена в 2005 році компанією Harmonix Music Systems. Гра вивела взаємодію гравця з музикою на новий рівень.

Однією з ключових особливостей цієї гри є використання спеціального гітарного контролера, який моделює гітару. Гравець має взяти участь в "живому виступі", натискуючи кнопки і грати на струнах, синхронізуючи свої дії з музичним ритмом.

Гра пропонує величезний вибір різноманітних музичних треків різних жанрів, від класичного року до сучасних хітів. Guitar Hero має різні режими гри, включаючи кампанію для одного гравця, режим для багатьох гравців і онлайн-змагання (рис 1.2).

Рисунок 1.2 – Гра Guitar Hero [2]

З розвитком серії з'явилися нові елементи, такі як барабанні контролери, мікрофони для вокалістів та інші інструменти, що дозволяли більш глибоко поглибити ігровий процес.

Guitar Hero внесла значний внесок у світ відеоігор та музичну індустрію. Ця серія стала не тільки інноваційною в геймдизайні, а й стимулювала інтерес до вивчення музики та розвитку музичних навичок у гравців. Навіть після того, як серію призупинили, Guitar Hero залишає слід у світі відеоігор як одна з найуспішніших у своєму жанрі.

1.3 Аналіз гри Osu!

Osu! – це відома ритм-гра, яка була у 2007 році та стала відомою своєю високою складністю, а також різноманіттям музичних треків і режимів гри. "osu!" є вільною та відкритою для гравців та спільноти.

Гра заснована на концепції кіл (бітмапів), які з'являються на екрані. Гравці повинні виконувати різні дії, такі як клацання, утримання або прокрутка миші або тачпада відповідно до ритму музики (рис 1.3).

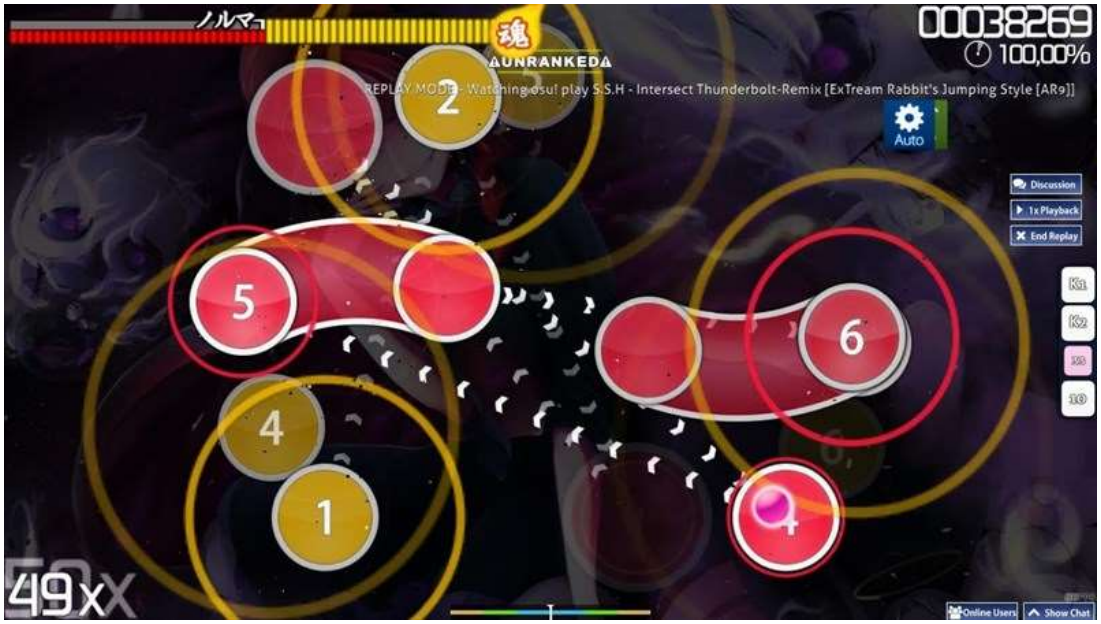

Рисунок 1.3 – Гймплей гри “osu!” [3]

Музичний вибір в "osu!" славиться своїм розмаїттям музичних жанрів, включаючи популярні треки, ремікси, варіації та композиції, надані спільнотою гравців. Гравці отримують оцінку за кожну карту, яку вони грають, і мають

можливість покращити свої навички, вибиваючи рекорди і піднімаючи свій рейтинг.

Гра активно підтримує доповнення та кастомізацію гри. Спільнота гравців розробляє свої карти, створює скіни та різноманітні додаткові режими гри. "osu!" підтримує гравців різного рівня, від початківців до досвідчених майстрів ритм-ігор. Професіонали можуть брати участь в різноманітних турнірах та змаганнях, що додає конкурентний аспект до гри.

Завдяки своїй активній спільноті, постійним оновленням і розширеним можливостям кастомізації гри залишається однією з найпопулярніших ритм-ігор та продовжує привертати увагу гравців усього світу (рис 1.4).

Рисунок 1.4 – Чемпіонат світу з гри osu! [4]

1.4 Аналіз гри Rhythm Heaven

Rhythm Heaven - це серія ритм-ігор від японської компанії Nintendo. Яка вперше вийшла на консолі Nintendo Game Boy Advance у 2006 році в Японії і відтоді отримала кілька продовжень та версій для різних консолей.

Центральний елемент гри - це виконання різноманітних ритмічних завдань або "міні-ігор". Гравець виконує вказані рухи або клацання на екрані в такт музиці, щоб успішно завершити завдання. Гра славиться за свій веселий та захоплюючий саундтрек, який включає в себе різні жанри музики. Кожна міні-гра має власну унікальну музичну тему.

Rhythm Heaven має яскравий та кольоровий візуальний стиль, який створює веселий та живий атмосферний фон. Гра містить різноманітні персонажі та міні-ігри, кожен з яких має свою унікальну механіку та виклики. Також має режими тренувань, що допомагають гравцеві вдосконалювати свої навички та викликають бажання грати далі (рис 1.5).

Рисунок 1.5 – Гра Rhythm Heaven [5]

Rhythm Heaven відома своєю простотою, але водночас цікавим геймплеєм та захопливими музичними композиціями. Гра отримала позитивні відгуки від гравців та критиків за її оригінальний підхід до ритм-ігор та неповторний стиль.

1.5 Аналіз гри Muse Dash

Muse Dash - це музична ритм-гра, яка вийшла в 2018 році для різних платформ, включаючи ПК та мобільні пристрої. Гра розроблена і видана компанією PeroPeroGames, і вона швидко завоювала популярність завдяки своєму яскравому візуальному стилю, енергійному геймплею та різноманітним музичним темам.

Гравці керують персонажем, який рухається вліво або вправо по доріжці, реагуючи на різні музичні ноти та ворогів. Завдання полягає в уникненні перешкод та виконанні різноманітних ритмічних дій в такт музиці (рис 1.6).

Рисунок 1.6 – Гра Muse Dash [6]

Muse Dash славиться своїм звуковим наповненням, яке включає в себе різні музичні жанри, такі як поп, рок, електро та інші. Гравці можуть насолоджуватися різноманіттям треків під час гри. Гра має яскравий та кольоровий анімаційний візуальний стиль, що робить геймплей більш захоплюючим та енергійним.

Гра пропонує гравцям різні рівні складності, від простих для початківців до викликів для досвідчених. Гравці можуть вибрати свого улюбленого персонажа та вдосконалити його вигляд за допомогою різноманітних костюмів

та аксесуарів. Гра підтримує мультиплеерний режим, де гравці можуть змагатися між собою у реальному часі.

Muse Dash набула великої популярності у ігровому товаристві завдяки своєму привабливому геймплею, гарному звуковому оформленню та веселому візуальному стилю. Гра продовжує отримувати позитивні відгуки і збільшувати свою аудиторію через різноманітні додатки та оновлення вмісту.

1.6 Аналіз гри Beat Saber

Beat Saber - це віртуальна реальність ритм-гра, яка вийшла в 2018 році і стала однією з найпопулярніших ігор для VR-платформ.

Гравець отримує два світлових мечі, які виглядають як світлові шпажки. Завдання полягає в тому, щоб в у ритмі музики розрубати пролітаючі куби та інші об'єкти, натискуючи на них визначеним чином. Гравець отримує бали за кожен правильний рух (рис 1.7).

Beat Saber славиться своїм енергійним та різноманітним саундтреком.

Рисунок 1.7 – Геймплей гри Beat Saber [7]

Гравці можуть вибирати з різних музичних треків різних жанрів, а також встановлювати додаткові пакунки з новою музикою. Гра пропонує різні рівні складності, від простих для новачків до дуже складних для досвідчених гравців.

Крім основного режиму гри, існують режими редагування, що дозволяють гравцям створювати власні рівні, а також режими для змагань з іншими гравцями. Гра має активну спільноту гравців, яка розробляє власні рівні, а також моди та додатки для розширення можливостей гри.

Beat Saber впроваджує технологію віртуальної реальності, дозволяючи гравцям зануритися в ігровий світ і відчути максимальний рівень взаємодії з музикою та об'єктами гри.

Завдяки своєму геймплею, стильному дизайну та використанню віртуальної реальності швидко завоювала популярність серед любителів ритм-ігор та VR-ігор. Гра надала новий рівень розваг у світі відеоігор.

1.7 Аналіз гри Friday Night Funkin'

Friday Night Funkin' - це популярна музична ритм-гра. Вона вийшла в жовтні 2020 року та швидко стала популярною в соціальних мережах і отримала велику кількість шанувальників, завдяки своєму харизматичному герою та веселому геймплею.

Гравець управляє Бойфрендом, головним героєм, який випадково потрапив у музичний змагальний батл. Головне завдання - перемогти різних музичних опонентів, виконуючи різні ритмічні комбінації (рис 1.8).

Рисунок 1.8 - Гра – Friday Night Funkin' [8]

Friday Night Funkin' славиться своїм заразливим та різноманітним саундтреком. Кожен опонент представляє різний музичний жанр, від хіп-хопу до електронної музики. Гра захоплює своїм анімаційним стилем та креативним дизайном персонажів. Все виглядає яскраво, енергійно та оригінально.

Проект активно підтримує моддинг, що дозволяє гравцям створювати власні рівні, персонажів і музичні треки. Це великою мірою сприяє творчому вдосконаленню гри та розширює її вміст.

Friday Night Funkin' відзначається нестандартним підходом до жанру ритм-ігор, привабливим дизайном і заразливою музикою. Гра вже стала культовою серед шанувальників і продовжує здобувати нових прихильників, розширюючи свою популярність в онлайн-спільноті.

1.8 Аналіз гри Project SEKAI: Colorful Stage!

Project SEKAI: Colorful Stage! - це музична ритм-гра, розроблена та випущена компанією SEGA для мобільних платформ, таких як iOS та Android. Гра створена на основі популярного віртуального вокального ідола Японії, Hatsune Miku (рис 1.9).

Рисунок 1.9 – Гра Project SEKAI: Colorful Stage! [9]

Гравці управляють своїм персонажем і взаємодіють із музичними нотами, які відтворюють популярні пісні з використанням вокального синтезатора. Гра базується на жанрі ритм-ігор і вимагає від гравців точності та реакційності для успішного виконання пісень.

Гра має багате музичне наповнення, яке включає в себе пісні від різних артистів, а особливо ті, які пов'язані з Hatsune Miku. Це може включати як оригінальні композиції, так і популярні треки від спільноти.

Project SEKAI захоплює кольоровим та яскравим дизайном. Персонажі виглядають анімаційно та милозвучно, а фони та анімації доповнюють загальну візуальну привабливість гри. Гра може мати різні режими, такі як кампанія для одного гравця, онлайн-режими для змагань з іншими гравцями та інші розважальні режими.

1.9 Аналіз гри Cytus

Cytus - це інноваційна музична ритм-гра для мобільних пристроїв, розроблена компанією Rayark Inc. Вийшла вперше в 2012 році, гра отримала позитивні відгуки за свій унікальний підхід до жанру та вражаючий аудіовізуальний досвід.

Гравець взаємодіє з різноманітними об'єктами, які з'являються на екрані в такт музиці. Об'єкти можуть бути нотами, кругами або лініями, і гравець повинен торкатися, утримувати або слідувати за ними відповідно до ритму (рис 1.10).

Рисунок 1.10 – Гра Cytus [10]

Гра включає в себе широкий спектр музичних жанрів, від електроніки і популярної музики до інді композицій.

Кожен рівень має власну унікальну атмосферу, яка відображається в графіці та анімаціях. Гра включає в себе елементи історії та введення персонажів, які додають глибину сюжету для гравців.

Проект пропонує різні рівні складності, від простих для початківців до дуже важких для досвідчених гравців. Гравці можуть змагатися онлайн, а також очікувати регулярних оновлень, що додають нові треки та рівні.

Cytus відзначається високою якістю виконання і стала улюбленцем серед гравців, які цінують ритмічні виклики та музичний вражаючий вміст.

1.10 Аналіз гри BPM: Bullets Per Minute

BPM (Bullets Per Minute) [11] - це відеогра у жанрі roguelike-шутера з великим акцентом на ритм та стрільбу (рис 1.11).

Рисунок 1.11 – Гра BPM (Bullets Per Minute) [11]

Унікальність "BPM" полягає в тому, що гравець повинен діяти в такт музиці. Всі події в грі, включаючи ворогів, стрільбу та рух, синхронізовані з ритмічними ударами музичного бекграунду.

Гра комбінує в собі елементи випадкової генерації рівня, шутера від першої особи та ритм-ігор. Гравець досліджує випадково генеровані рівні, збирає зброю та зустрічає різноманітних ворогів.

"BPM" має енергійний саундтрек, який підтримує темп гри. Візуальний стиль також створює унікальну атмосферу, доповнену світловими ефектами та динамічними бойовими сценами. Гравець може обирати різних персонажів, кожен з яких має унікальні властивості, а також використовувати широкий вибір зброї.

BPM вирізняється своєю оригінальною концепцією ритмічної стрільби та комбінацією різних жанрів. Гра знаходить свою аудиторію серед тих, хто шукає виклик у поєднанні інтенсивної стрільби та музичного ритму.

1.11 Аналіз гри Rhythm Doctor

Rhythm Doctor - це незвичайна ритм-гра, яка вирізняється своєю унікальною механікою гри, де гравець використовує точність та таймінг для виконання різних ритмічних завдань.

Гравець використовує простий механізм клацання, але з складністю - виконання дій відбувається на 7-й такт. Гравець має натискати кнопку або клавішу відповідно до такту музики.

Гра пропонує різноманітні завдання та музичні стилі. Також славиться своїм чудовим саундтреком, який включає в себе різноманітні музичні жанри та допомагає створити захоплюючий геймплей (рис 1.12).

Рисунок 1.12 – Гра Rhythm Doctor

Оригінальний візуальний стиль вражає дизайном різних рівнів та персонажів, створюючи унікальну атмосферу для кожного завдання. У грі є історійний режим, який розповідає історію головного героя.

Створені користувачами, а також мультиплеєрні режими, що додають різноманіття до геймплею.

Rhythm Doctor стала популярною серед гравців, які шукають унікальний ритмічний виклик та захоплюючий музичний досвід. Її оригінальна механіка гри та креативний підхід до ритм-ігор зробили її визнаною в цьому жанрі.

1.12 Висновки до розділу

Розглянувши існуючи ритм ігри дійшли такого висновку. Гра має буди приємною на вигляд, мінімалістичною. Моделі повинні бути простимим, щоб не відволікати гравця. На екрані не повинно бути багато візуальних ефектів. Звук повинен бути зв’язаний з грою, щоб гравець міг орієнтуватись на нього, а не тільки на картинку.

2 ВИБІР ПРОГРАМНОГО ЗАБЕСПЕЧЕННЯ ДЛЯ СТВОЕННЯ РИТМ ГРИ

Розглянемо інструменти, що спеціалізуються на різних жанрах гри. Деякі платформи краще підходять для ігор в жанрі ритму, інші - для екшн чи пригод. Визначаємо, який жанр потрібно реалізувати та які функціональні вимоги будуть необхідні.

Проаналізуємо мови програмування, які використовуються в рушіях розробки. Та обрати ті з якими буде зручніше працювати.

При виборі програмного забезпечення для створення гри важливо враховувати потреби, навички та мету проекту. Правильний вибір інструментів допоможе вам максимально використати свій творчий потенціал та досягти успіху в розробці гри.

2.1 Обґрунтування вибору ігрового рушія

Вибір ігрового рушія для створення ритм гри є важливим етапом в розробці проекту. Принципи вибору відображають основні критерії, які слід враховувати при цьому виборі.

Важливо, щоб ігровий рушій підтримував легке втілення ритмічних ефектів та анімацій. Він повинен забезпечувати зручний інтерфейс для синхронізації подій гри із музичним ритмом.

Відповідний аудіо-рушій є ключовим для ритм-гри. Він повинен підтримувати високоякісну роботу зі звуком, можливість синхронізації аудіо та геймплею.

Необхідно обрати рушій, який комфортний у використанні та має зручні інструменти для розробки ритм гри. Наявність документації та активної спільноти користувачів також є ключовим фактором. Рушій повинен бути

ефективним та оптимізованим для роботи з ритмічними взаємодіями та графікою, особливо якщо планується велика кількість анімацій.

Враховуючи ці принципи, давайте розглянемо та виберемо рушій, який відповідає нашим потребам та дозволяє ефективно реалізувати концепцію ритм гри.

Unity - це один з найпопулярніших інтегрованих середовищ розробки та ігрових рушіїв для створення різних видів ігор, включаючи ритм-гри. Він дозволяє розробляти гри для різних платформ, таких як Windows, macOS, Linux, Android, iOS, та інші. Це робить його гарним вибором для розробників, які хочуть охопити різні цільові аудиторії (рис 2.1).

Рушій має велику та активну спільноту розробників. Це означає, що можна легко знайти допомогу, поради та різні ресурси для навчання. Також існує обширна документація, яка спрощує роботу з ігровим рушієм.

Існує багато плагінів та розширень для Unity, які дозволяють розробникам легко впроваджувати нові функції та покращувати робочий процес. Unity Asset Store - це великий ринок, де розробники можуть придбати або продавати різноманітні ресурси, плагіни, моделі, текстури та інше. Це дозволяє прискорити розробку, використовуючи готові ресурси.

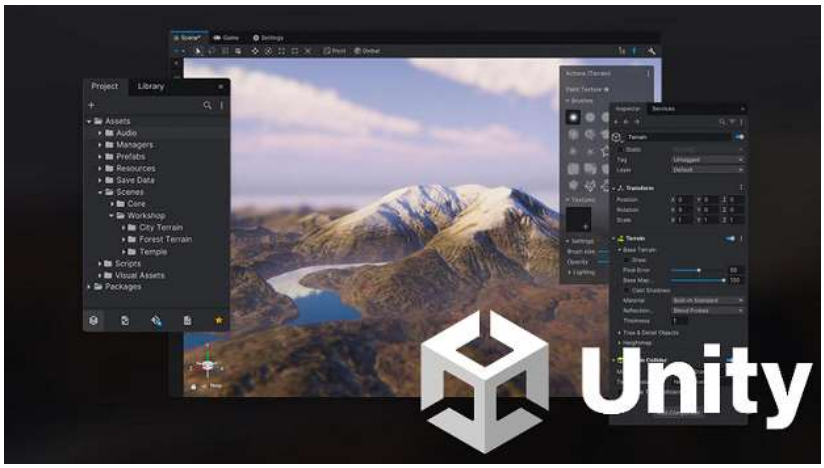

Рисунок 2.1 – Рушій Unity [13]

Але, рушій може вимагати значних ресурсів комп'ютера для розробки та перевірки ігор, особливо якщо проект є складним або має велику кількість

данних. Деякі розробники вказують на те, що код, створений в Unity, може бути об'ємним, особливо для великих проектів.

Рушій є залежним від мови C#. За деякими повідомленнями, також, Unity може мати обмеження при реалізації деяких ігрових механік або ефектів.

Хоча Unity має свої недоліки, він залишається одним з найпопулярніших інструментів для розробки ігор через свою доступність та розширюваність. Вибір між Unity та іншими рушіями повинен залежати від конкретних потреб та особливостей вашого проекту.

Unreal Engine - потужний ігровий рушій, розроблений компанією Epic Games. Він використовується для створення високоякісних ігор та інтерактивних візуальних проектів. Unreal Engine надає широкий функціонал, включаючи потужну графіку, детальний фізичний рушій, інструменти для розробки віртуальної реальності, а також можливості кросплатформеного розгортання. Рушій використовує мову програмування C++ для розширення можливостей та надає користувачам доступ до відкритого коду для більшої гнучкості та налаштувань. Unreal Engine також славиться своєю великою та активною спільнотою, а також наявністю безкоштовної версії для особистого використання.

Поріг входу для Unreal Engine є досить низьким. Навіть людина без досвіду програмування зможе розібратися з ним для своїх цілей. Навігація та налаштування багато в чому схожі на інші програми. Легкість входу обумовлена функцією Blueprint (рис 2.2). Це візуальний інтерфейс для створення логіки гри без програмування. Функція дозволяє створювати логіку гри за допомогою блоків, які представлені графічними елементами та з'єднані лініями, що відображають порядок виконання. Це дозволяє розробникам швидко створювати та реалізовувати різноманітні ігрові механіки без необхідності в глибоких знаннях програмування.

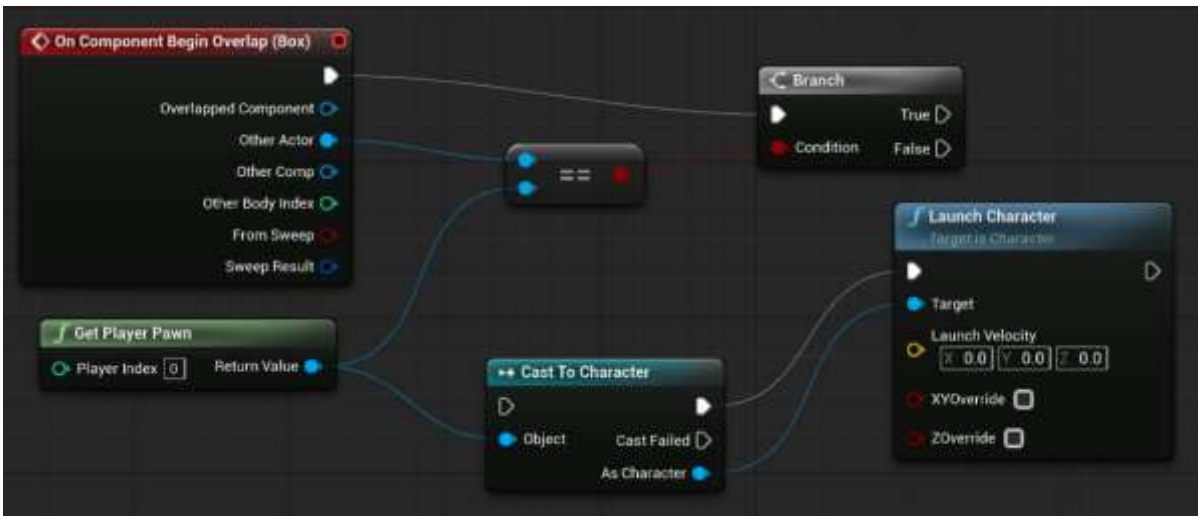

Рисунок 2.2 – Система візуального скриптингу Blueprint [14]

Unreal Engine підтримує різні платформи, включаючи ПК, консолі та мобільні пристрої, що робить його універсальним інструментом для створення ігор для різних аудиторій. Рушій володіє вражаючою гнучкістю, що дозволяє реалізовувати різноманітні ідеї та ігрові механіки. Завдяки великій спільноті розробників та документації, Unreal Engine є доступним із підтримкою для широкого спектру технологій і мов програмування. Можливість безкоштовного використання та гнучка модель оплати за успіх роблять цей рушій привабливим для розробників будь-якого рівня навичок (рис 2.3).

Рисунок 2.3 – Рушій Unreal Engine [15]

GameMaker — це інтегроване середовище розробки та ігровий рушій, створений компанією YoYo Games. Цей інструмент призначений для створення 2D-ігор на різних платформах, включаючи Windows, macOS, iOS, Android та інші.

GameMaker пропонує простий для використання інтерфейс, який дозволяє новачкам створювати ігри без програмування. Однак, для більш складних проектів доступний вбудований скриптовий мова.

Високорівнева мова програмування, спеціально розроблена для створення ігор у GameMaker. Яка базується на C і дозволяє розробникам створювати складні ігрові логіки та механіки (рис 2.4).

Рушій має вбудовану підтримку фізики, що дозволяє легко створити реалістичні фізичні ефекти, такі як гравітація, зіткнення та рух об'єктів.

Рисунок 2.4 – Рушій GameMaker [16]

Програма підтримує розгортання ігор на різних платформах, від настільних комп'ютерів до мобільних пристроїв. Це робить його відмінним вибором для розробки кросплатформових ігор.

GameMaker пропонує широкий набір вбудованих ресурсів, таких як анімації, звуки, спрайти та інші, що полегшує процес розробки.

В рушія є активна спільнота розробників, форуми, документація та навчальні ресурси, які допомагають новачкам та досвідченим розробникам вирішувати проблеми та ділитися знаннями.

У підсумку, GameMaker є потужним інструментом для створення 2D-ігор з високим рівнем доступності для різних категорій розробників. Завдяки своїм функціональним можливостям та широкій спільноті, GameMaker продовжує залишатися популярним вибором серед творців ігор.

2.2 Обґрунтування вибору програми для модулювання 3D графіки

Вибір правильної програми для 3D-моделювання є критичним кроком на шляху до досягнення цілей. З урахуванням різноманітних функціональних можливостей, інтерфейсів, цінових категорій та специфіки використання, вибір правильної програми може вплинути на ефективність роботи, якість результуючих моделей та загальний успіх проекту.

Розглянемо ключові аспекти, які слід враховувати при виборі програмного забезпечення для 3D-моделювання. Дізнаємось різницю між різними програмами, їх основні функціональні можливості, а також оцінемо, як зробити оптимальний вибір відповідно до індивідуальної потреби та професійних навичок.

Blender – це інструмент для 3D-моделювання, який став вкрай популярним серед творчих спеціалістів, від початківців до професіоналів.

Однією з ключових переваг Blender є те, що це вільне та відкрите програмне забезпечення. Вільний доступ до коду дозволяє користувачам не тільки користуватися програмою безкоштовно, але й вносити свої власні зміни, що сприяє постійному вдосконаленню і розвитку.

Blender дивує своїм широким функціоналом, що включає в себе моделювання, текстурування, анімацію, рендеринг, арматурну анімацію, водночас з можливістю редагування звуку та відео. За допомогою програми можна створювати динамічні анімаційні сцени, використовуючи скелетну

анімацію, часткову систему, симуляцію тканин та інші інструменти. Це надає користувачам комплексний набір інструментів для втілення різноманітних ідей та проектів (рис 2.5).

Рушій має активну глобальну спільноту користувачів, яка завжди готова допомогти та поділитися досвідом. На порталах, таких як «Blender Artists» або «Stack Exchange», ви зможете знайти відповіді на будь-які питання та знайти натхнення в роботах інших творців.

Рисунок 2.5 – Програма для роботи з графікою Blender [17]

Blender є багатоплатформеним, підтримуючи операційні системи, такі як Windows, macOS і Linux. Це робить його доступним для широкого кола користувачів, незалежно від їхньої операційної системи.

Розробники постійно вдосконалюють програму, випускаючи нові версії з інноваційними функціями та поліпшеннями. Це гарантує, що ви завжди працюєте з останніми технологіями та можливостями.

Blender — це потужний та універсальний інструмент для 3D-моделювання, який відкриває безліч можливостей для творчості, навчання та професійного

росту. Незалежно від вашого досвіду та амбіцій, Blender може стати відмінним вибором для реалізації ваших 3D-проектів.

Autodesk Maya - є однією з найбільш визнаних програм для 3D-моделювання, анімації та візуалізації. Завдяки своїм інструментам, вона широко використовується в індустрії відеоігор, кіно, телебачення та архітектури.

Maya надає широкий спектр інструментів для створення високоякісних 3D-моделей, включаючи полігональне моделювання, скульптинг та роботу з поверхнями. Продвинуті інструменти для скелетної анімації, морфінгу, кінематографії та симуляції дозволяють створювати реалістичні та динамічні анімаційні сцени (рис 2.6).

Рисунок 2.6 – Програма для роботи з графікою Autodesk Maya [18]

Програма інтегрується з рендер рушієм Arnold, який забезпечує відмінну якість візуалізації, реалістичне освітлення, матеріали та ефекти. В програмі присутні спеціалізовані інструменти для водної симуляції, волосся, тканин, а також динамічних ефектів, які допомагають створювати складні та деталізовані сцени.

Maya є вибором для багатьох професіоналів у сферах кіно, відеоігор та телебачення завдяки своїм інструментам та можливостям.

Завдяки доступним версіям для освітніх установ та навчальних програм, студенти можуть вивчати Maya та готуватися до професійної кар'єри в сфері 3D-графіки.

Autodesk Maya є інструментом для 3D-моделювання, анімації та візуалізації, який забезпечує широкі можливості для творчості, професійного росту та реалізації навіть найбільш амбітних проектів. Його рішення використовуються професіоналами по всьому світу, демонструючи високу ефективність та надійність у найвимогливіших умовах.

ZBrush, розроблений компанією Pixologic, є визнаним лідером у сфері цифрового скульптування та 3D-моделювання. Ця програма перетворила підхід до створення 3D-графіки, надаючи художникам можливість працювати з віртуальним гончаром, матеріалом для створення високоякісних деталізованих моделей (рис 2.7).

Рисунок 2.7 – Програма для роботи з графікою ZBrush

За допомогою програми користувачі можуть створювати деталізовані 3D-моделі, використовуючи потужні інструменти скульптування, присутні

спеціалізовані інструменти для створення волосся, шерсті, води та інших природних елементів, що робить моделювання ще більш реалістичним.

ZBrush дозволяє створювати текстури та матеріали, а також використовувати їх у інших програмах для подальшого рендерингу. Програма може інтегруватися з іншими популярними програмами для 3D-моделювання та рендерингу, такими як Autodesk Maya, Blender та інші.

Програма часто використовується в індустрії відеоігор та кіно для створення персонажів, анімаційних сцен та спеціальних ефектів. Вона відкриває нові горизонти у світі цифрового скульптування та 3D-моделювання, надаючи художникам надзвичайно потужний інструмент для творчості. Його високоякісні інструменти, гнучкість та інноваційний підхід роблять його незамінним для професіоналів у сфері візуальних мистецтв, анімації та розробці ігор.

2.3 Обґрунтування вибору програми для роботи з 2D-графікою

Створення 2D-графіки є важливою складовою процесу розробки комп'ютерних ігор. Вибір правильної програми може суттєво вплинути на якість, ефективність та творчий потенціал проекту Розглянемо кілька ключових програм для роботи з 2D-графікою та їхні основні характеристики.

Adobe Photoshop є визнаним лідером у галузі растрової графіки та став необхідним інструментом для багатьох художників та дизайнерів, що працюють у сфері від створення ілюстрацій до розробки ігор.

Основні функціональності Adobe Photoshop для 2D-графіки це - Створення та Редагування Спрайтів програма дозволяє легко створювати та редагувати спрайти - основні елементи графіки у 2D-іграх. Ви можете працювати з окремими шарами, групами та застосовувати різноманітні ефекти.

За допомогою Photoshop можна створювати текстури для об'єктів та фонів гри. Інструменти текстурного редагування дозволяють долучати деталі та створювати різні стилі. Також програма підтримує створення анімації через

роботу з кадрами. Ви можете створювати анімовані спрайти, переходи та інші рухомі ефекти (рис 2.8).

Система шарів в Photoshop дозволяє структурувати проекти, роблячи їх легшими для редагування та управління. Кожен об'єкт може бути розташований на окремому шарі. Завдяки різноманіттю функцій, Photoshop є ідеальним інструментом для створення графіки для комп'ютерних ігор різного жанру та стилю.

Adobe Photoshop залишається ключовим інструментом для тих, хто працює з 2D-графікою у сфері розробки ігор. Від його потужних інструментів до легкості використання, ця програма дозволяє художникам та розробникам створювати вражаючі графічні елементи, які впливають на візуальний аспект ігор.

Рисунок 2.8 – Програма для роботи з 2D графікою Adobe Photoshop [19]

Krita є безкоштовною програмою з відкритим вихідним кодом, яка набула популярності серед художників, ілюстраторів та розробників ігор завдяки своїм потужним інструментам для створення 2D-графіки. Давайте розглянемо, чому Krita може бути відмінним вибором для роботи з 2D-графікою в створенні комп'ютерних ігор (рис 2.9).

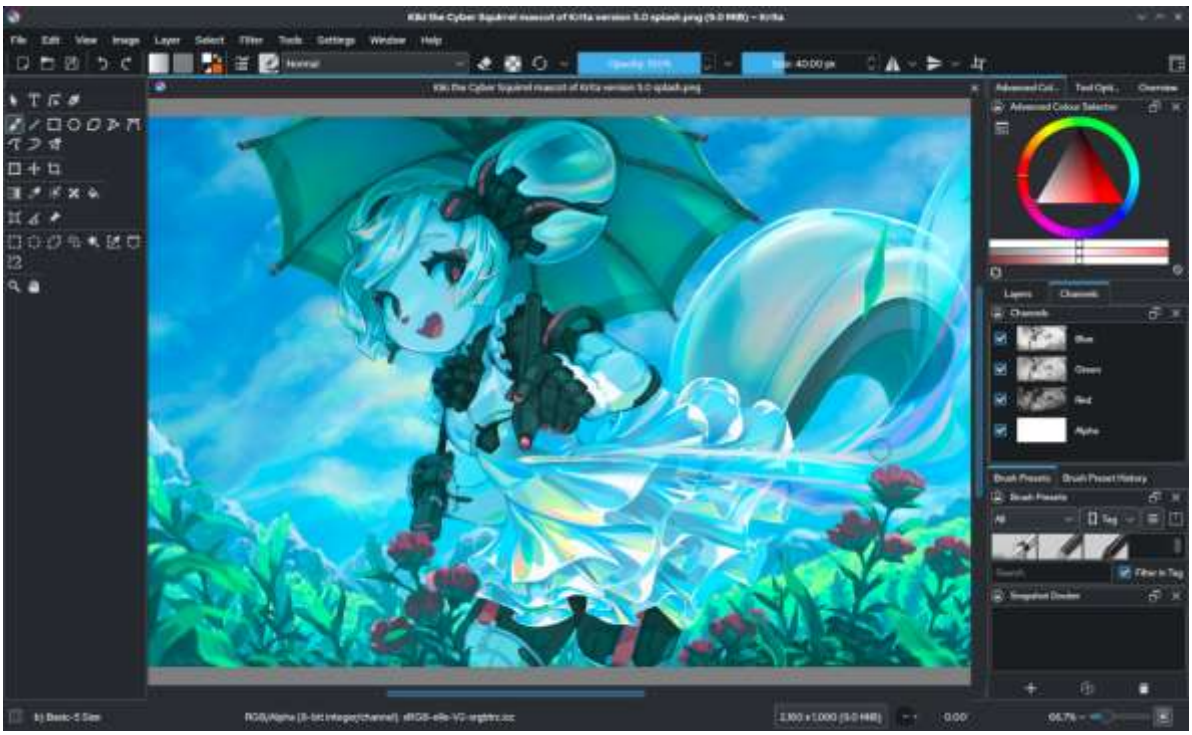

Рисунок 2.9 – Програма для роботи з 2D графікою Krita [20]

Krita пропонує ряд інструментів для цифрового малювання, включаючи пензлі, кисті, текстури та інші ресурси, що дозволяють створювати високоякісні 2D-ілюстрації та анімацію. Подібно до інших програм для 2D-графіки, Krita підтримує систему шарів, а також має інструменти для створення анімацій, що робить його відмінним вибором для створення спрайтів та інших 2D-елементів гри.

2.4 Обґрунтування вибору програми для створення музики

Музика в ритм-іграх є ключовим елементом, що визначає атмосферу, ритмічність та загальний досвід гравця. Важливо віддати особливу увагу створенню гарної музики для таких ігор.

Музика в ритм-іграх повинна бути синхронізована з геймплеєм, дозволяючи гравцю відчувати ритм та потік гри. Правильно підібрана музика може стимулювати гравця дотримуватися правильного ритму та досягати кращих результатів.

Створення гарної музики для ритм-ігор є критично важливим елементом, що визначає якість, емоційний зв'язок та загальний успіх проекту. Правильний музичний супровід може підсилити ритм, атмосферу та загальний досвід гри, роблячи її більш захоплюючою та пам'ятною для гравців.

Розглянемо програми, які дозволять нам створити відповідну музику.

FL Studio є однією з найбільш популярних та визнаних програм для створення музики. Цей цифровий аудіоредактор набув великої популярності серед професійних музикантів, продюсерів та любителів музики завдяки своїм функціям та інтуїтивно зрозумілому інтерфейсу.

Програма пропонує многодоріжковий інтерфейс, що дозволяє створювати, редагувати та організовувати різноманітні музичні композиції з великою кількістю доріжок. Також, FL Studio містить велику бібліотеку вбудованих інструментів, включаючи синтезатори, семпли, ефекти звуку та інші інструменти для створення унікальних звукових образів.

FL Studio інтегрується з різноманітними плагінами та іншими зовнішніми інструментами, що розширює його функціональні можливості та дозволяє користувачам створювати різноманітні музичні проекти (рис 2.10).

Рисунок 2.10 – Програма для створення музики FL Studio [21]

Додаток є гнучким інструментом для музичного творчості, що підходить для професійних музикантів, продюсерів та новачків. З його великою бібліотекою інструментів, ефектів, секвенсерів та інших функцій, FL Studio дозволяє користувачам реалізувати свої творчі ідеї та створювати високоякісну музику у різних жанрах та стилях.Ableton Live є однією з провідних програм у сфері музичного виробництва, яка відзначається своїм унікальним підходом до створення музики та живого виступу. Цей програмний інструмент набув популярності серед професійних музикантів, діджеїв та продюсерів завдяки своїм функціям та гнучкому інтерфейсу.

Додаток пропонує унікальну концепцію сесійного та режиму аранжування, що дозволяє користувачам експериментувати з лупами, зразками та звуковими елементами в реальному часі, а також організовувати та структурувати їх в аранжуванні пісні (рис 2.11).

Рисунок 2.11 – Програма для створення музики Ableton Live [22]

Програма має великий набір вбудованих інструментів, синтезаторів, ефектів звуку та зразків, що дозволяє користувачам створювати унікальні звукові палітри та текстури.

Ableton Live є гарним інструментом для музичного виробництва та живого виступу, що пропонує широкі можливості для створення, редагування та виконання музики в різних стилях та жанрах. З його унікальними функціями, гнучким інтерфейсом та інтеграцією з різноманітними інструментами і платформами [23].

Для створення звука я обрав програму FL Studio. Вона відповідає усім нашим вимогам, та дозволяє створити необхідну музику для нашого проєкту.

2.5 Висновки до розділу

Розробка комп'ютерної гри є складним та комплексним процесом, що вимагає специфічних навичок, інструментів та ресурсів. Використання спеціалізованих програм, що відповідають потребам нашого проекту, може значно підвищити ефективність розробки та збільшити продуктивність. Відповідні інструменти дозволяють прискорити процеси створення, тестування та оптимізації гри.

Вибір правильних програм для розробки графіки, анімації та звукового супроводу є ключовим для створення високоякісної та привабливої гри. Спеціалізовані інструменти дозволяють розробникам створювати реалістичні, деталізовані та естетично приємні графічні ефекти.

Важливо вибирати програми, які легко інтегруються між собою та іншими інструментами, що використовуються у проекті. Це забезпечує гладкий потік роботи, уніфікований стандарт розробки [23].

Правильний вибір програм може зменшити витрати часу на розробку, забезпечуючи доступ до необхідних функцій та інструментів без зайвих витрат на додаткове програмне забезпечення або ресурси.

Тому у якості рушія для проекту я обрав Unreal Engine. Для створення 3D графіки та анімацій програму Blender. Для роботи з 2D графікою Adobe Photosop, а для роботи зі звуком FL Studio.

3 РОЗРОБКА МУЗИЧНО-РИТМІЧНОГО ІГРОВОГО КОНТЕНТУ

3.1 Створення ігрового персонажа

Розробка гри це комплексний процес, під час якого потрібно намалювати концепт, створити інтерфейс майбутньої гри та елементи оточення. Зробити 3D-модель головного героя та інших персонажів, написати музику та звук, а далі перенести це все на ігровий рушії, у якому вже зробити з цього всього гру.

Почнемо зі створення концептів майбутніх елементів гри. Концепт це ідея на папері, на яку опираються при подальшій розробці. У ньому потрібно передати форми, розміри та основні деталі об’єкту, який буде візуалізовано в подальшому.

Спершу треба знайти референс та намалювати концепт нашого персонажа (рис 3.1). для подальшого переносу у 3D графіку. Для цього користуємося програмою Adobe Photoshop. Зробимо стиль схожий до гри Muse Dash, яку ми розглядали раніше.

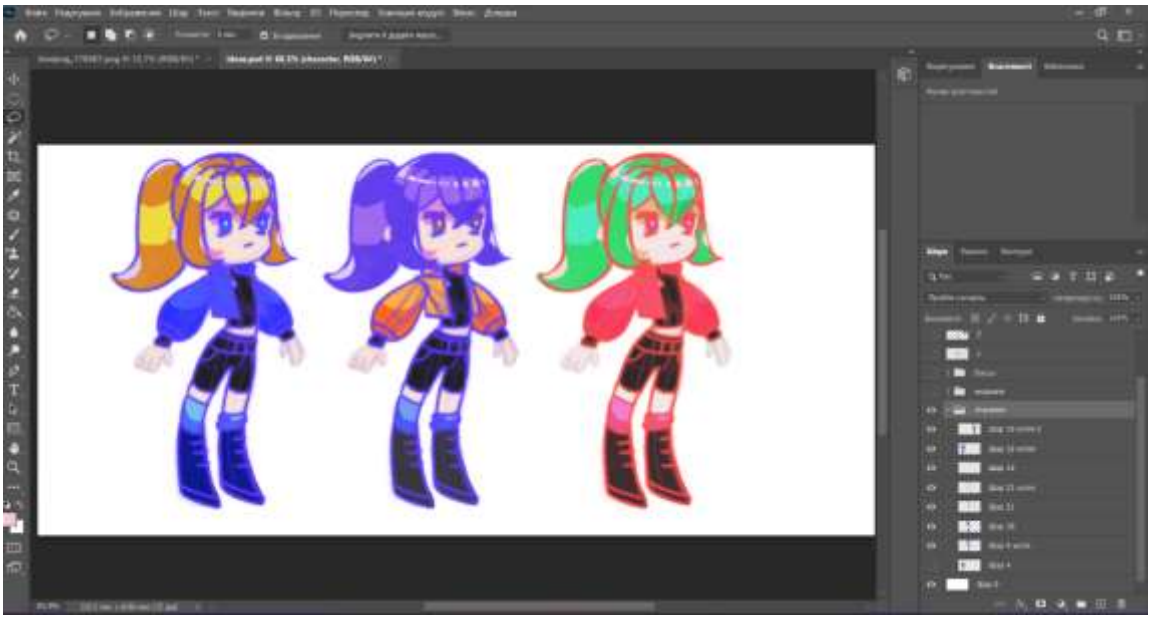

Рисунок 3.1 – Концепт головного героя

Розробляючи концепт головного героя для гри, він була намальований у трьох різних кольорових палітрах, у подальшому будемо використовувати їх як додатковий одяг для нашого персонажа [24].

Далі потрібно створити 3D-модель персонажа (рис 3.2). Для цього запускаємо програму для роботи з тривимірною графікою Blender.

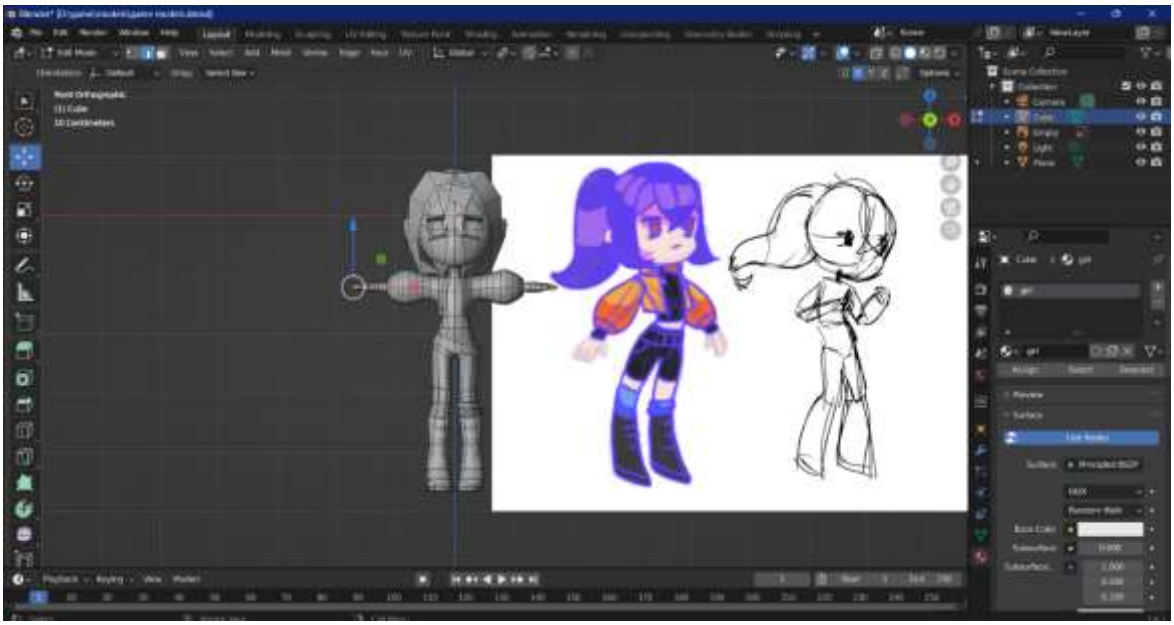

Рисунок 3.2 – Створення моделі головного героя з референса

Стилістика проекту є низько полігональна графіка. Тому моделюємо персонажа з нуля, екстрадуючи полігони. У Blender є можливість розмістити рисунок прямо у робочій зоні. Тому ми відкриваємо малюнок і при моделюванні намагаємось в точності відтворити намальований раніше концепт [25].

Після цього необхідно зробити розгортання (рис 3.3). Розгортання - це процес створення текстур для 3D-моделі, який дає можливість відобразити її тривимірну поверхню на плоскості.

Для початку потрібно перейти в режим редагування моделі, щоб мати доступ до її вершин, ребер та граней. Після цього виділяємо ту частину моделі, яку потрібно розгорнути [26].

Однією з основних команд для розгортання є "Unwrap". Це дозволяє програмі автоматично розгорнути виділену частину моделі, розміщуючи її на плоскості текстури.

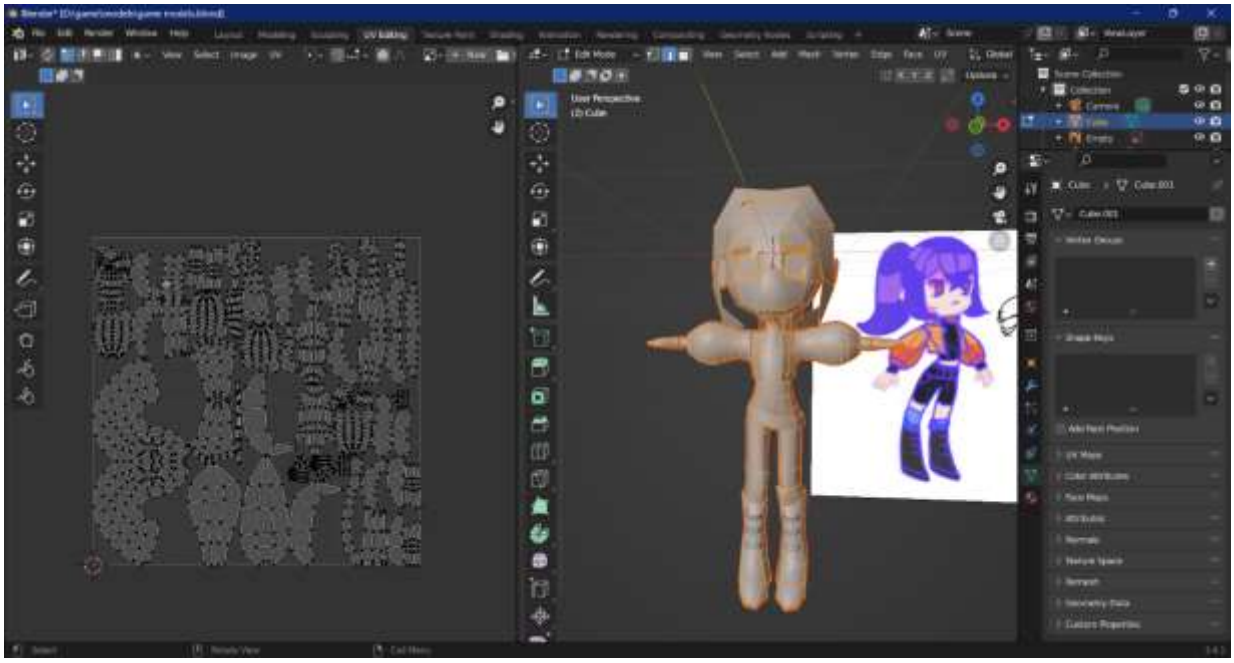

Рисунок 3.3 – Розгортка моделі головного героя

Після розгортання переходимо в режим редагування “UV Editing”, де можна переглядати та редагувати розгортку як 2D-зображення. Важливо максимально оптимізувати розгортку, розміщуючи її елементи так, щоб забезпечити зручність подальшого редагування текстур [27]. Переходимо у режим "Draw", та починаємо розмальовувати нашу модель.

Після виправлень і редагувань зберігаємо розгортку. Потім переходимо у режим 3D для застосування створеної текстури до моделі. Цей процес відображає текстуру на відповідних частинах 3D-моделі, враховуючи її форму та деталі.

Тепер необхідно під’єднати намальовані текстури до моделі (рис 3.4). Для подальшого експорту заходимо у розділ “Shading”. Далі перетягуємо текстуру у нижню зону екрану – саме тут відбуваються усі маніпуляції з матеріалми і текстурами моделі [28]. Обравши нашого персонажа під’єднуємо пін “Color” до пін “Surface”. Тобто, колір до самої поверхні моделі.

Через те що наша гра буде у плоскому стилі більше нам не потрібно додавати деталі. Якщо б гра була реалістичною, ми під’єднували б нормалі, які створюють деталізацію не навантажуючи модель, та інші елементи за потреби.

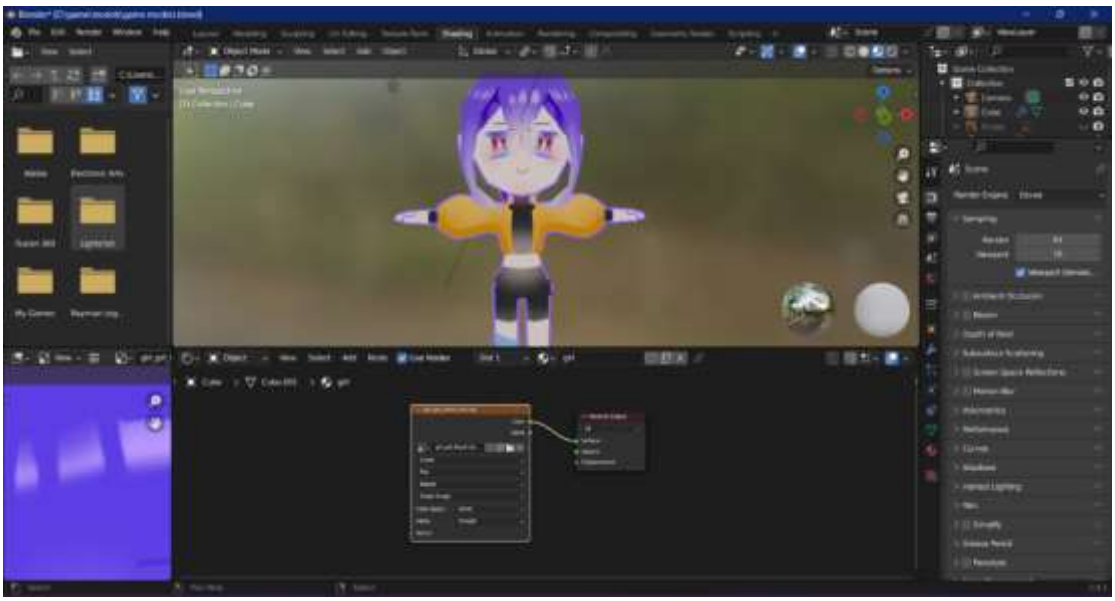

Рисунок 3.4 – Під’єднання текстур до моделі

Далі треба створити кістки для моделі. Процес створення кісток називається ріггінг (рис 3.5). Обираємо 3D-модель у режимі об'єкта і переходимо до режиму "Object Data Properties" [29]. Тут можна створити нову групу вершин для кожної частини моделі, яку плануємо анімувати.

У режимі об'єкта додаємо новий об'єкт - "Armature". Це буде основою системи кісток для анімації. Відкриваємо режим редагування "Armature", додаємо кістки, які будуть відповідати структурі моделі. Використовуємо різні типи кісток для різних частин тіла (наприклад, таз, стегна, гомілка тощо).

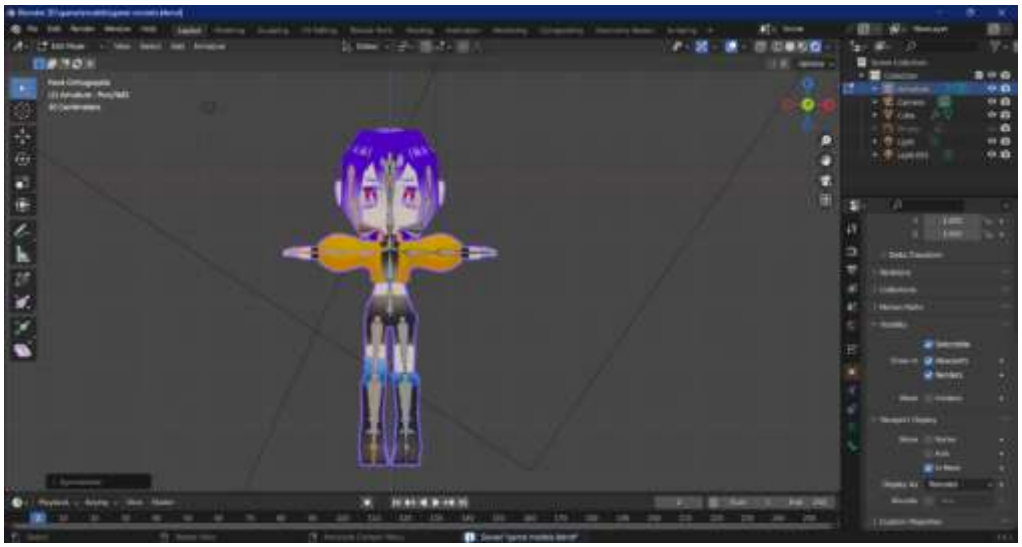

Рисунок 3.5 – Створення кісток для моделі

Після створення кісток повертаємось до режиму об'єкта моделі, обираємо модель, а потім "Armature". Користуємось комбінацію клавіш Ctrl + P для встановлення батьківства “Parenting” між моделлю та арматурою [30]. Обираємо опцію "With Automatic Weights", щоб Blender автоматично призначив ваги вершинам на основі їх близькості до кісток.

Далі необхідно протестувати анімацію, для цього переходимо до режиму анімації, створюємо кілька ключових кадрів та перевіряємо, чи правильно працює система кісток. Можна використовувати режим "Pose" для редагування позицій кісток та перевірки анімації [31].

Після завершення процесу ріггінгу можна переходити до створення анімацій, використовуючи арматуру для керування рухами моделі. Можна створювати ключові кадри, переходи між позами та інші анімаційні ефекти.

Створення анімації (рис 3.6). Для моделі у Blender - це процес, який дозволяє нам надати 3D-об'єктам життя, руху та взаємодії.

Переходимо до режиму "Animation" у нижній частині вікна Blender. Далі необхідно розмістити курсор часу (timeline cursor) на початок тимчасової лінії (timeline) [32]. Змінюємо положення моделі або кісток арматури, встановлюємо ключовий кадр (натискаємо "I" на клавіатурі і обираємо тип ключового кадра, наприклад, "Location" або "Rotation").

Створюємо анімовані послідовності, для цього переміщаємо курсор часу до іншого моменту на тимчасовій лінії. Змінюємо положення або форму моделі та встановлюємо ще один ключовий кадр [33]. Повторюємо цей процес для різних елементів анімації, створюючи плавні переходи між рухами.

Також використовуємо графік руху для детального керування анімаційними кривими, редакції ключових кадрів та створення плавних переходів між анімаційними елементами.

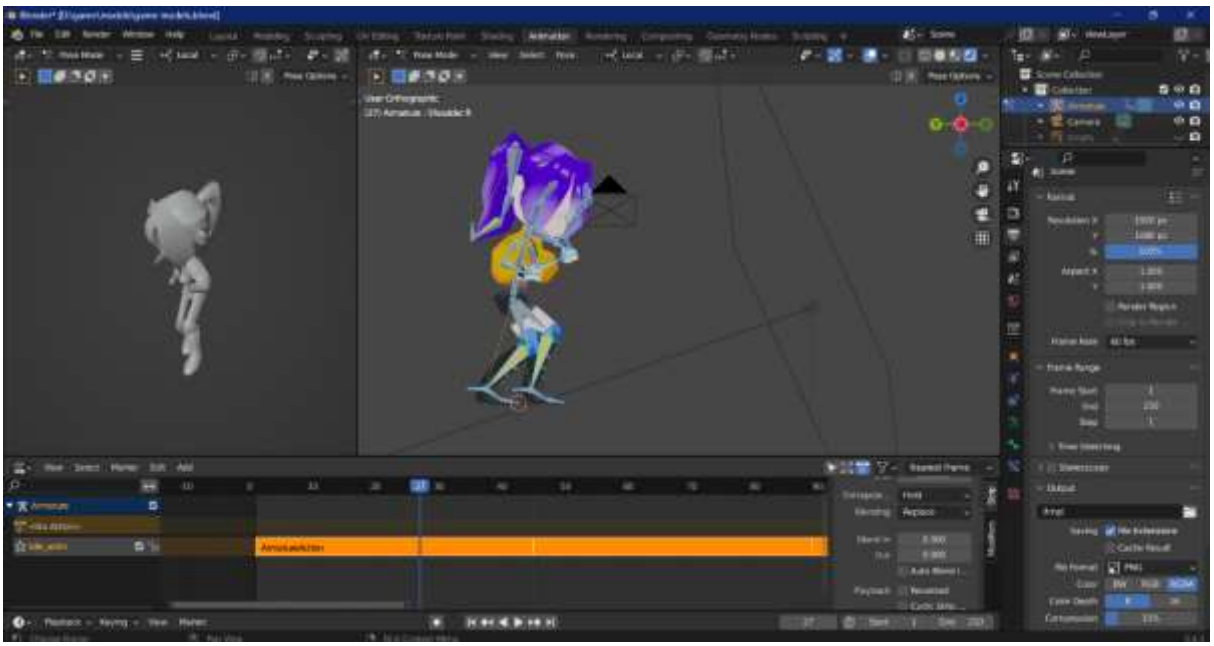

Рисунок 3.6 – Створення анімації

Таким чином створено готову модель головного героя, з текстурами та кістками. Також створено анімації, які знадобляться у подальшій розробці гри.

3.2 Створення оточення

Створення оточення для гри є важливим комплексним процесом, що вимагає детального підходу до дизайну та оптимізації .

Перш ніж розпочати створення оточення, розробимо концепт. Дотримуючись візуальної стилістики, можна створити захоплююче оточення для гри, яке зацікавить та захопить гравців. Також необхідно визначити вимоги та очікування від оточення [34].

При створенні зброї беремо референси з компонентів електричних кіл, а саме резистор, діод, катушка індуктивності (рис 3.7).

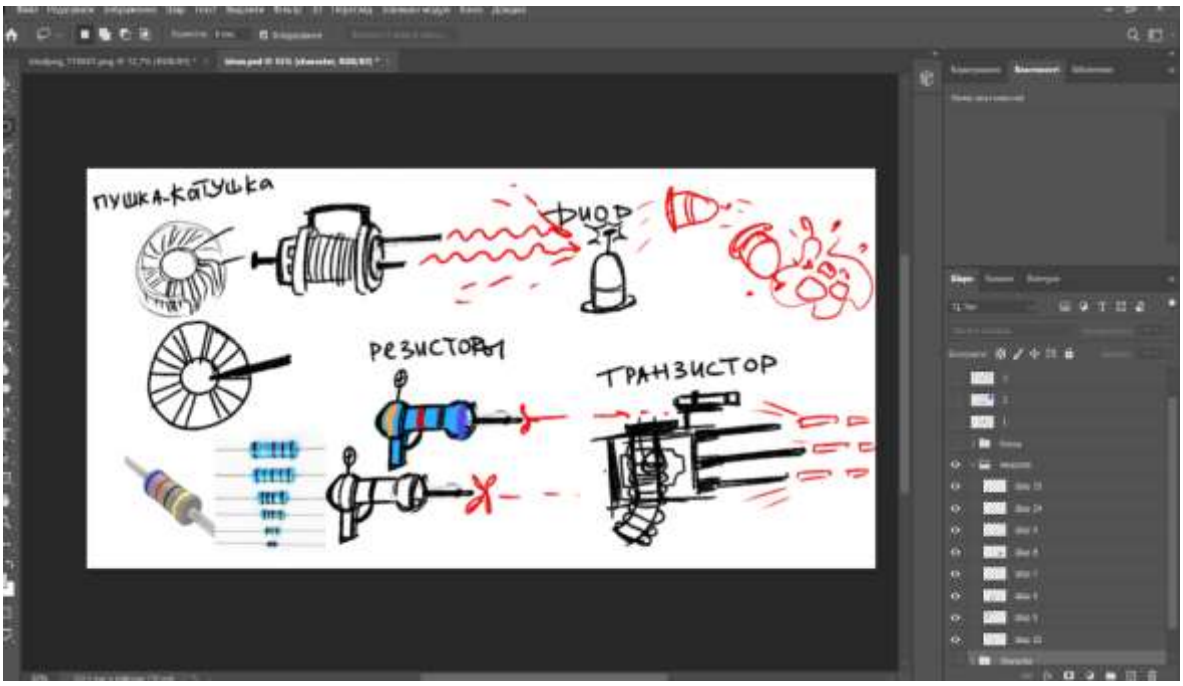

Рисунок 3.7 – Концепт зброї головного героя

У якості головного ворога було обрано осцилограф, та намальовано його концепт малюнок (рис 3.8).

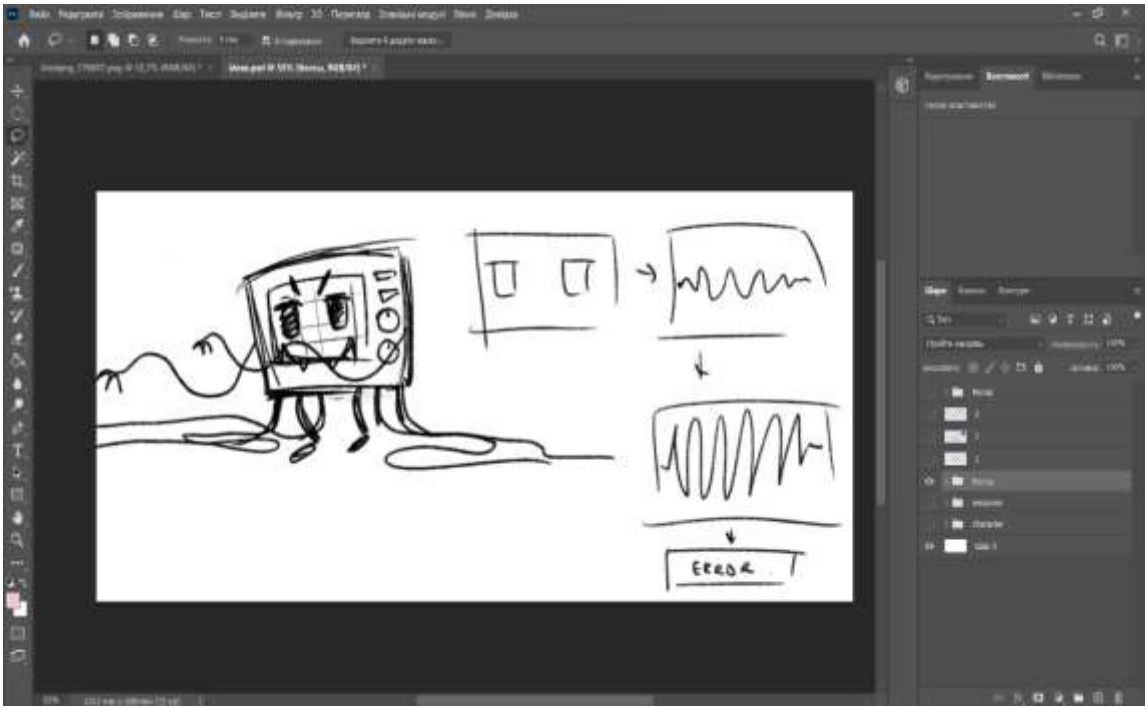

Рисунок 3.8 – Концепт осцилографа

Далі створюємо елементи оточення та задній фон гри. За основу беремо типову аудиторію ХНУРЕ, та починаємо малювати предмети. По перше намалюємо підлогу, стіни на стелю (рис 3.9).

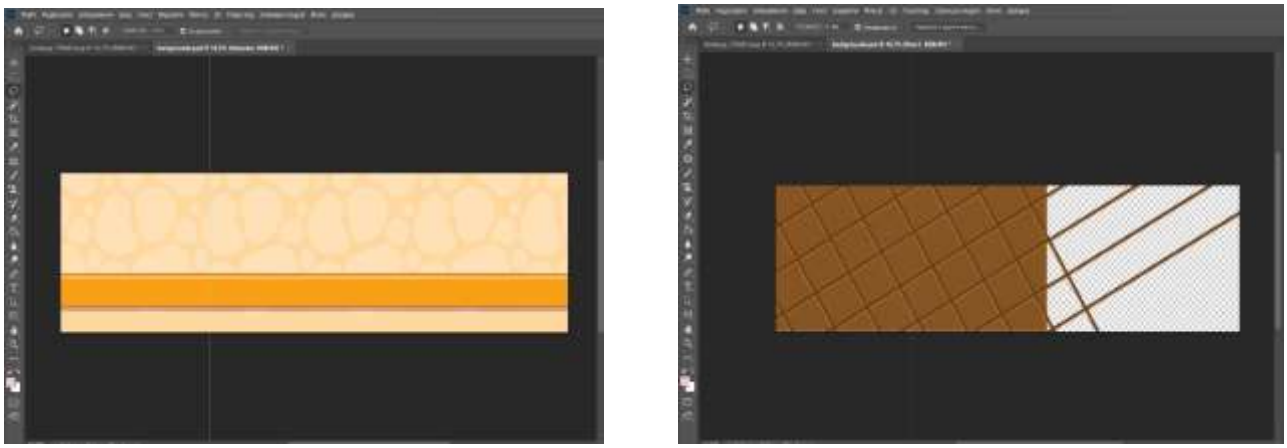

Рисунок 3.9 – Елементи оточення стіна та підлога

Тепер створюємо наповнення для кімнати малюємо дошку, яка додає нам більшу схожість на аудиторію (рис 3.10).

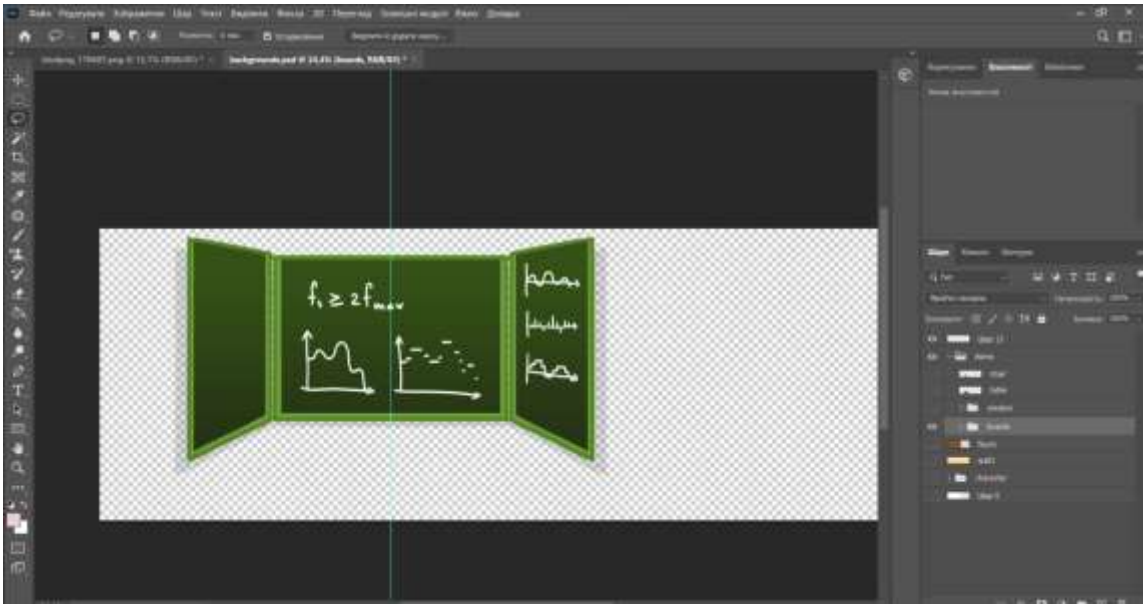

Рисунок 3.10 – Елемент оточення дошка

Далі намалюємо інші елементи оточення: стільці та парти. З ними гра буде виглядати біль насиченою деталями (рис 3.11).

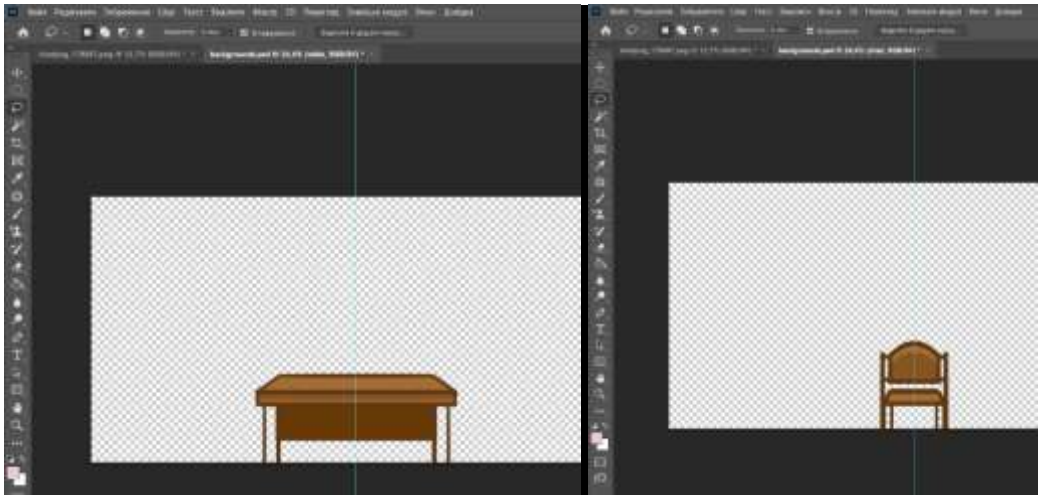

Рисунок 3.11 – Елементи оточення парта та стілець

Тепер ми можемо розставити парти та стільці на нашому ігровому рівні, таким чином, копіюючи і вставляючи оді і ті самі елементи оточення можна створити вигляд аудиторії, яка наповнена меблями.

Тепер малюємо вікно (рис 3.12).

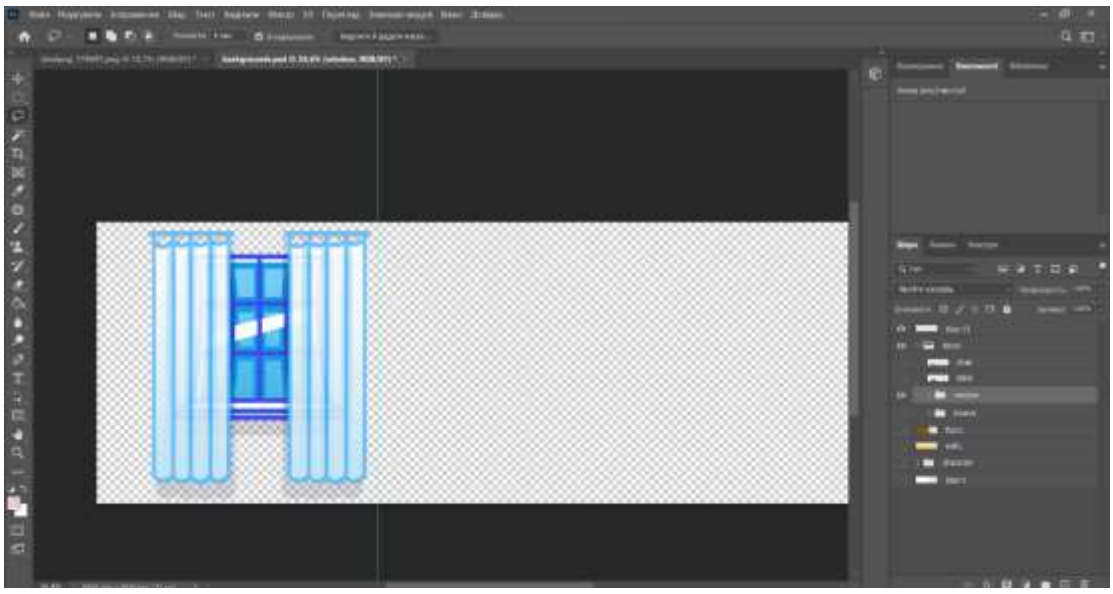

Рисунок 3.12 – Елемент оточення вікно

Тепер необхідно зробити 3D-модель зброї на осцилографа, відповідно до нашого концепту [35]. Інші об’єкти будуть у 2D – графіці і не підлягають переносу в трьох вимірну графіку.

Подібно до створення головного героя, створимо зброю (рис 3.13). Почнемо зі створення полігональної моделі пістолета.

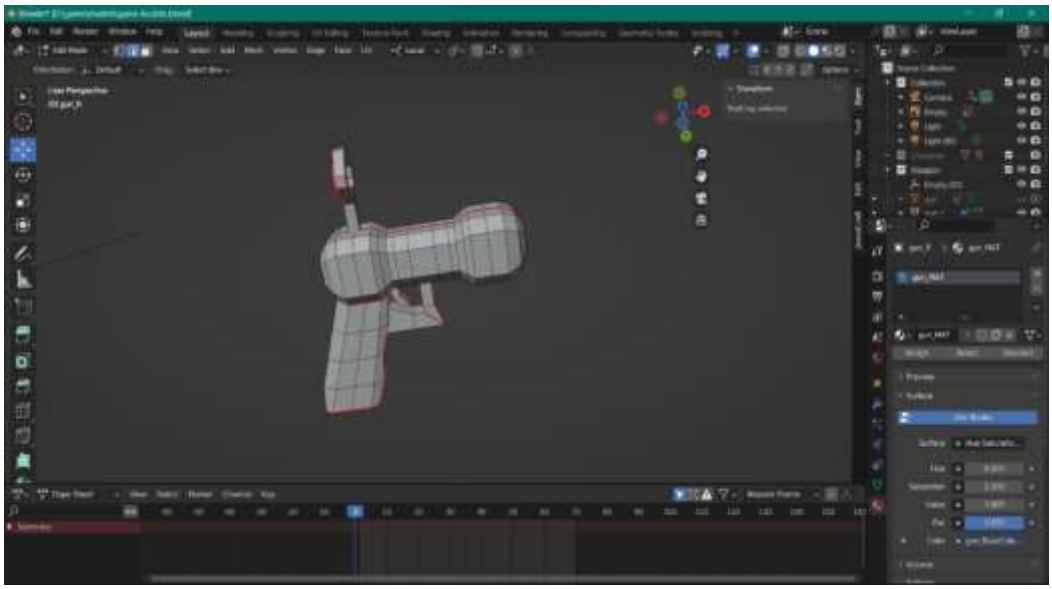

Рисунок 3.13 – 3D-модель пістолета

Тепер зробимо його розгортку, та розфарбуємо відповідно до маркування резистора (рис. 3.14).

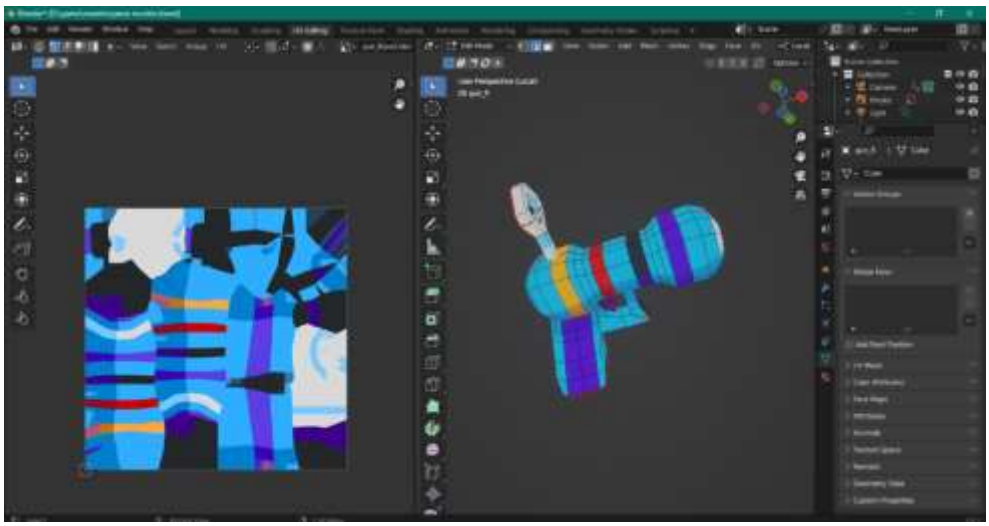

Рисунок 3.13 – Розгортка пістолета

Далі потрібно зробити 3D-модель осцилографа (рис 3.13). Також подібно до моделі головного героя. Зразу додаємо йому кістки.

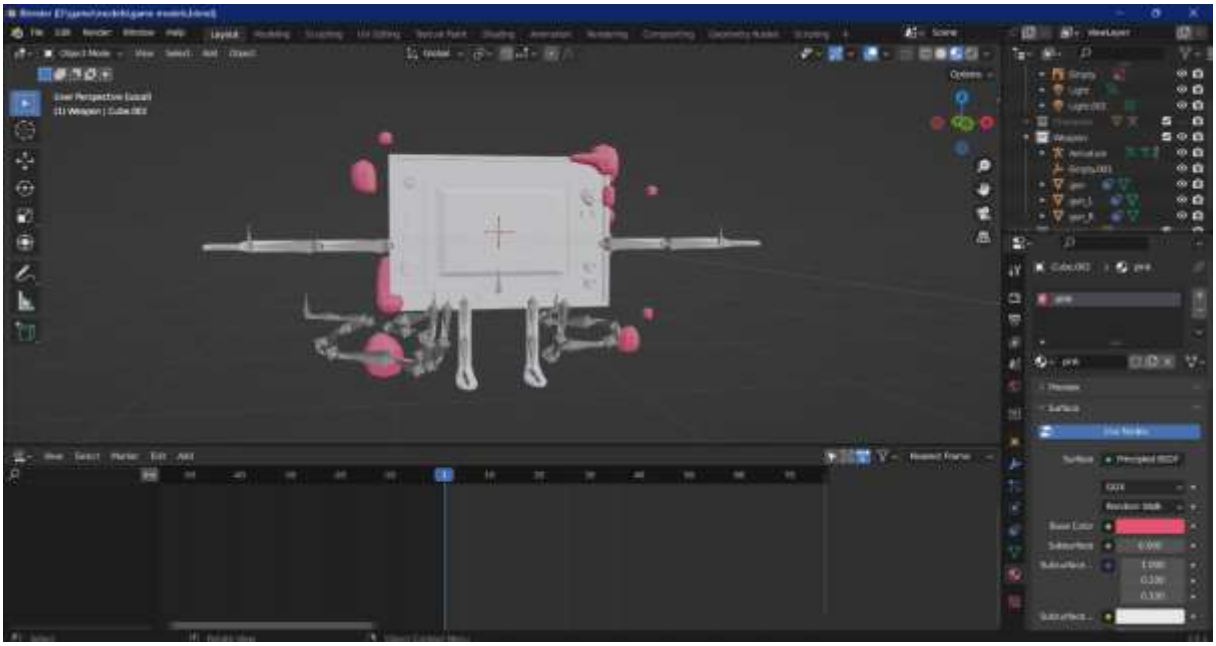

Рисунок 3.14 – 3D-модель осцилографа

Тепер зробимо розгортку осцилографа, розфарбуємо її та додаємо матеріалів для більшої деталізації (рис 3.15).

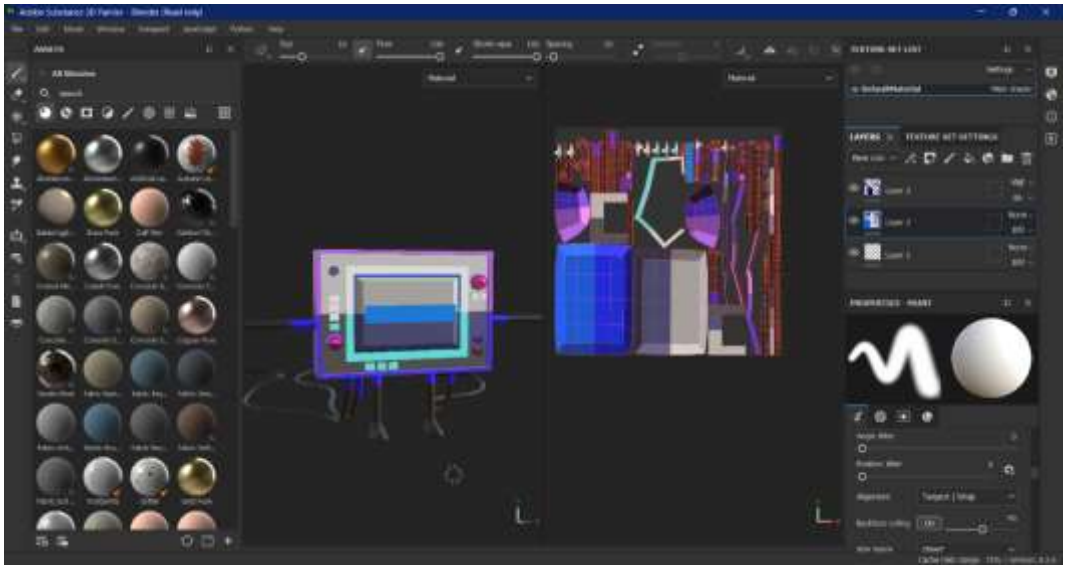

Рисунок 3.15 – Розгортка осцилографа

Тепер створимо елементи інтерфейсу. За потрібно намалювати кнопку, яка буде змінюватись при натисканні (рис 3.16).

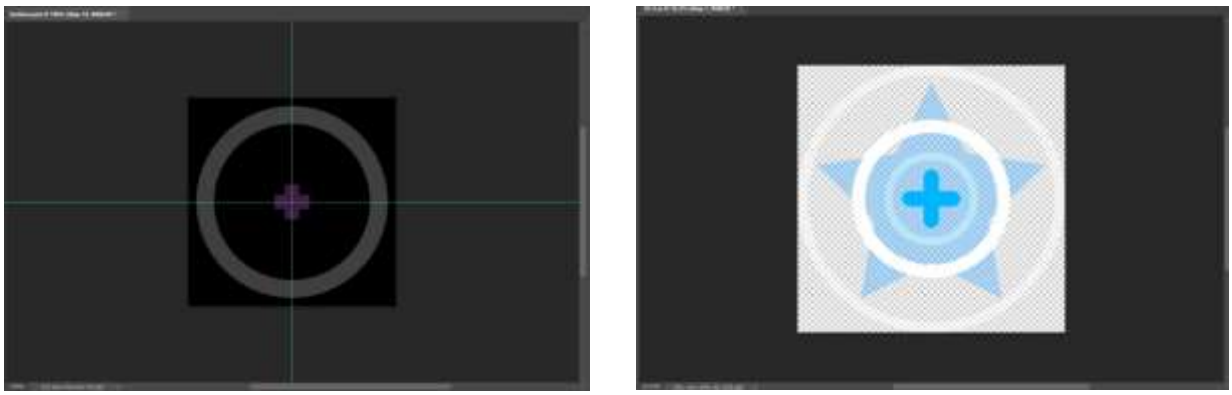

Рисунок 3.16 – Розгортка осцилографа

Далі створимо інтерфейс головного меню (рис 3.17). Та намалюємо картинку на фон (рис 3.18), щоб додати стилю грі.

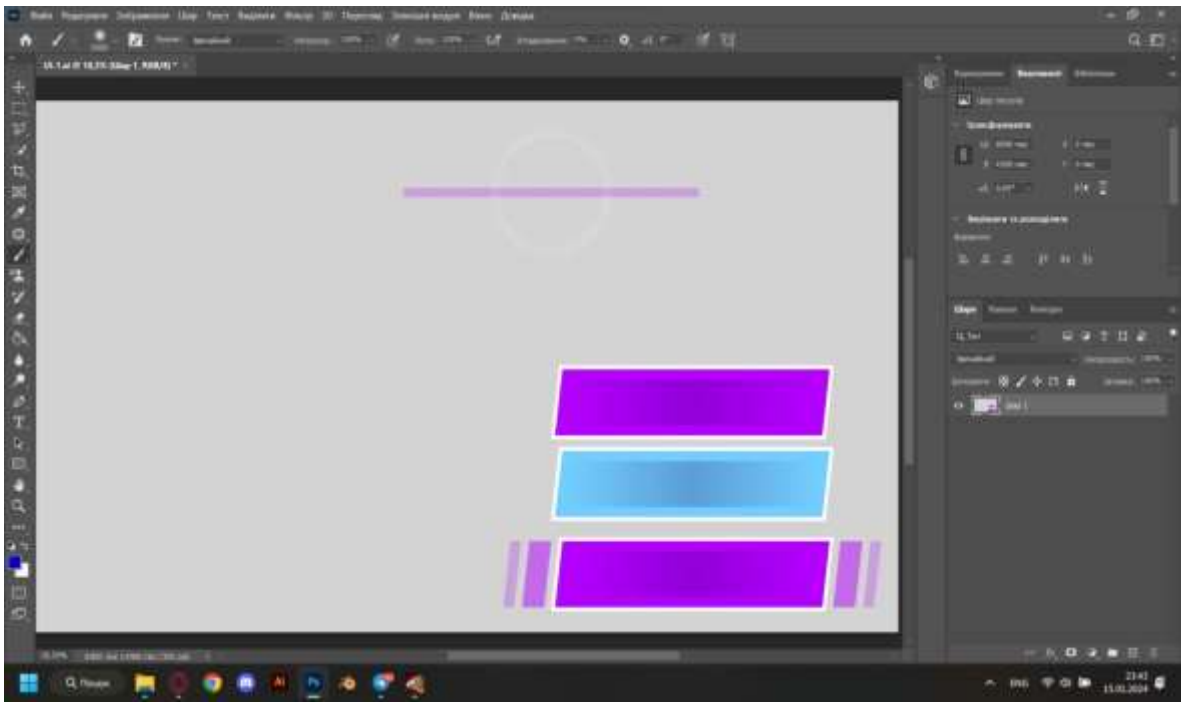

Рисунок 3.17 – Інтерфейс головного меню

На фоні зображуємо головного героя нашої гри.

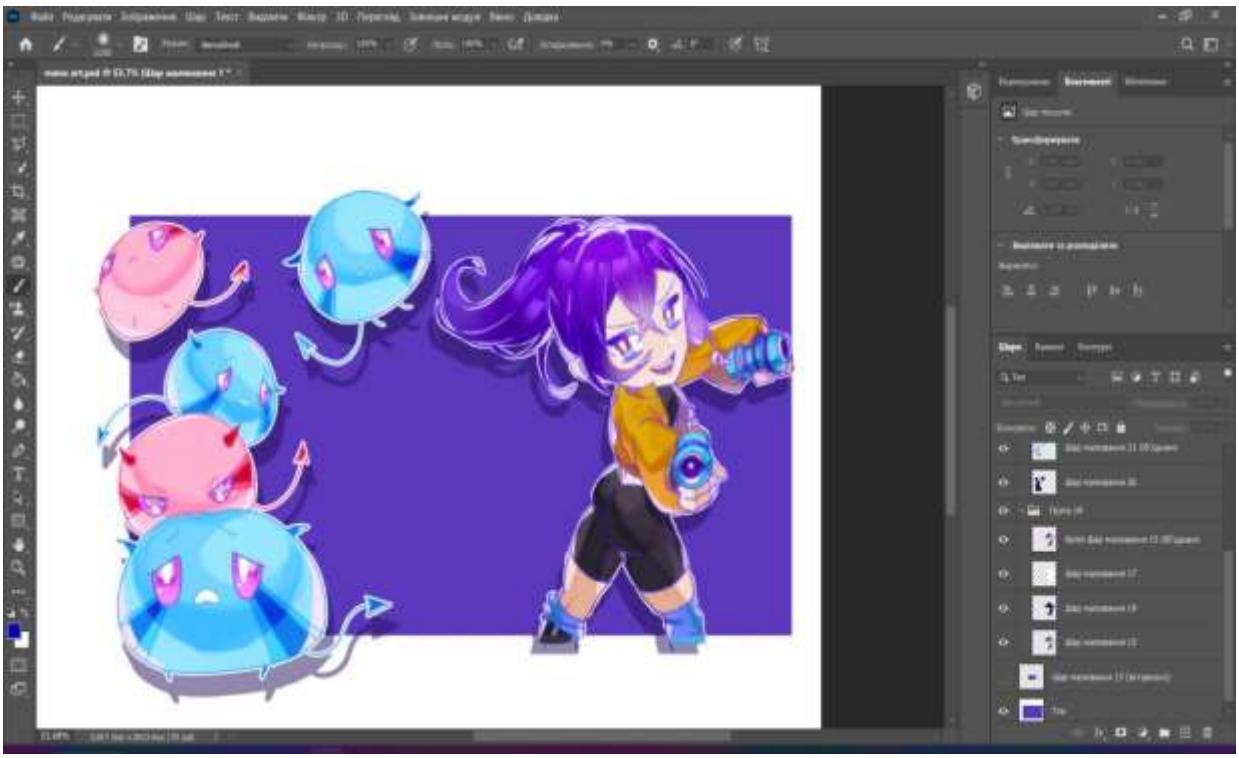

Рисунок 3.18 – Фон головного меню

Тепер намалюємо ворогів для ритм гри, з якими ми будемо взаємодіяти під час гри [36]. Вони мають бути простими, щоб не відволікати нас, але в той же час добре помітними (рис 3.19).

Рисунок 3.19 – Вороги ноти

У цьому підрозділі було створено елементи оточення та інтерфейси гри, які будуть перенесені на ігровий рушій.

3.3 Створення музичного наповнення

Створення музики для ритм-гри це творчий процес, який об'єднує музичні елементи та геймплей. Потрібно розуміти ритм, темп та структури рівня щоб синхронізувати музику з геймплеєм.

Для створення музики запускаємо програму FL Studio. Потрібно створити дві музикальні теми, які будуть відрізнятись одна від одної за рівнем складності.

Перший трек буде простішим, у нього буде менший BPM (beats per minute).

Почнемо зі створення семплів, які будуть використовуватись у подальшому створенні трека. Для першого семплу обираємо синтезатор (рис 3.20).

Рисунок 3.20 – Семпл синтезатору

Тепер створюємо семпл басу у вкладці “Bass” (рис 3.21).

Рисунок 3.21 – Семпл басу

Тепер можна створити перший трек, використовуючи створенні раніше семпли. Для цього розміщаємо семпли на звуковій доріжці (рис 3.22).

Рисунок 3.22 – Звукова доріжка першого треку

Тепер накладаємо на різні канали звуку ефекти, для цього відкриваємо мікшер (рис 3.23).

Рисунок 3.22 – Мікшер першого треку

Зберігаємо проект.

Далі створюємо другий трек, у якого буде більший БПМ, а відповідно у грі це буде вищий рівень складності.

Розставляємо семпли на звуковій доріжці, також додаємо семпли із бібліотеки FL Studio для більшого різноманіття (рис 3.23).

Рисунок 3.23 – Звукова доріжка другого треку

Тепер відкриємо мікшер для того щоб накласти різні єфекти на канали звуку, чим покращити звучання треку (рис 3.24).

Рисунок 3.24 – Мікшер другого треку

Зберігаємо трек. Пілся чого треба зберегти його у форматі mp3 для сумісності і рушієм. Для цього відкриваємо меню, натискаємо зберегти як, от обираємо відповідні параметри (рис 3.25).

Рисунок 3.24 – збереження треку у форматі mp3

Після збереження треків у відповідному форматі можна додати їм обкладинки, щоб гравці могли обрати музику орієнтуючись на картинки.

Для цього можна скористуватись штучним інтелектом, який створить обкладинки для треків. Застосуємо ШІ Midjourney. Для цього заходимо на відповідний сервер у додатку Discord. Та робимо запит на картинку на англійській мові (рис 3.25).

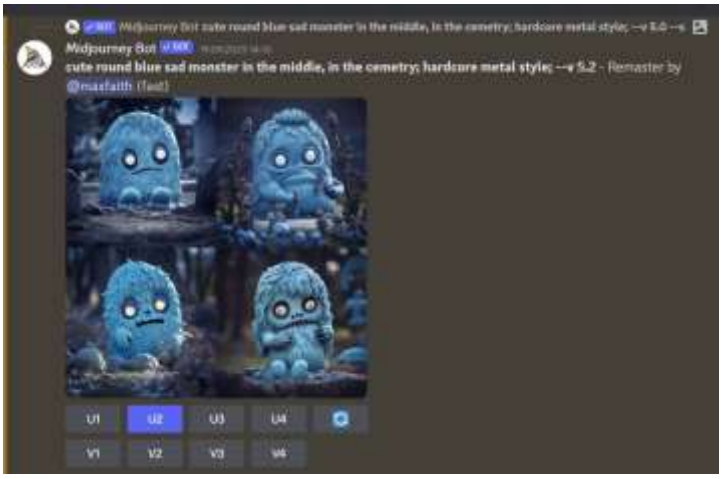

Рисунок 3.25 – Генерація окладники першого треку за допомогою ШІ

Обираємо один із чотирьох варіантів, які нам запропонував Ші, на свій смак.

Тепер зробимо запит на обкладинку для нашого другого треку (рис 3.26).

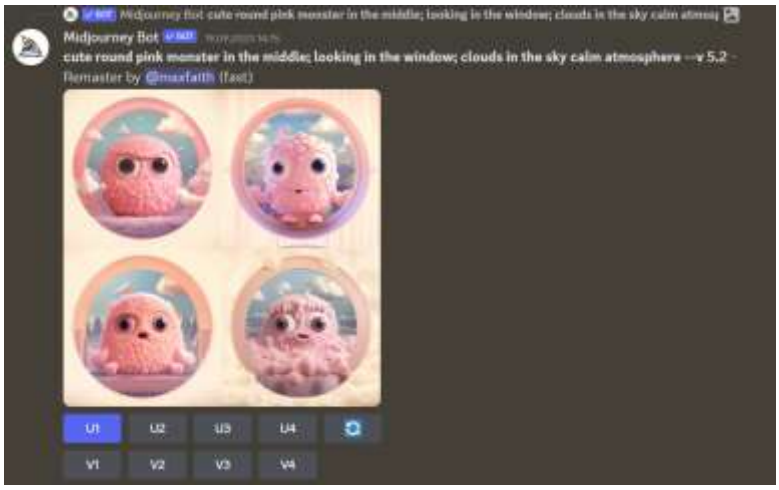

Рисунок 3.26 – Генерація окладники другого треку за допомогою ШІ

Таким чином було створено два треки, які будуть використовуватись під час гри.

3.4 Створення гри

Запускаємо Unreal Engine, обираємо створити новий проект гру, далі створюємо гру від третьої особи (рис 3.27).

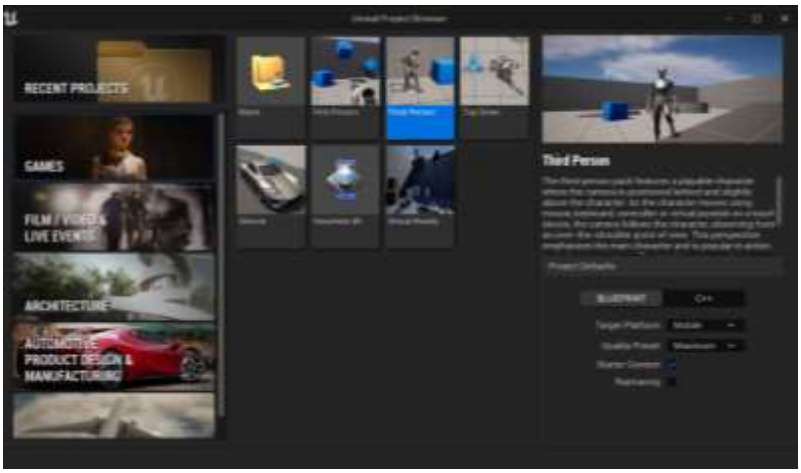

Рисунок 3.27 – Створення нового проекту в UE

У створений проект необхідно завантажити 3D-об’єкти, що були створені раніше. Для персонажу - спочатку завантажуємо саму модель без анімацій, але зі скелетом (input mesh відмічаємо) [37].

Потім завантажуємо окремо експортовані анімації у форматі FBX, знімаємо галочку з input mesh, а у пункты Skeleton обираємо модель персонажа (рис 3.28).

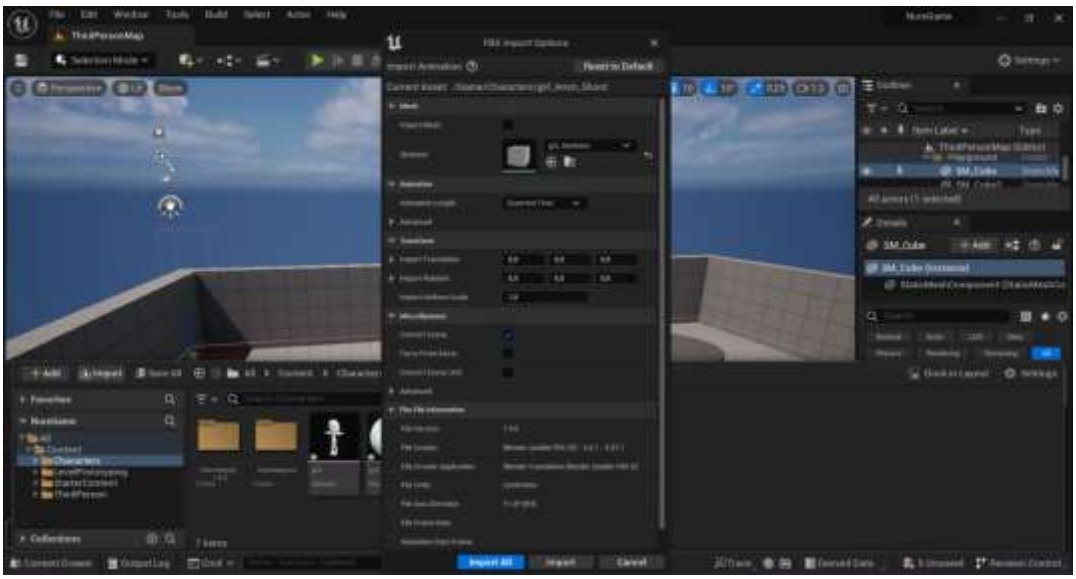

Рисунок 3.27 – завантаження 3D об'єкту

Тепер створюємо матеріал нашого персонажу (рис 3.28).

Рисунок 3.28 – матеріал ігрового персонажа

Через те, що персонаж буде мати плоске освітлення (Cell shading), потрібно налаштувати особливий матеріал, який буде обмежувати усе світло і тіні до двох значень [38]. Тобто якщо значення світла більше, за заданий коефіцієнт, то значення яскравості текстури буде 1, а якщо менше чи дорівнює - 0.5 (рис 3.29).

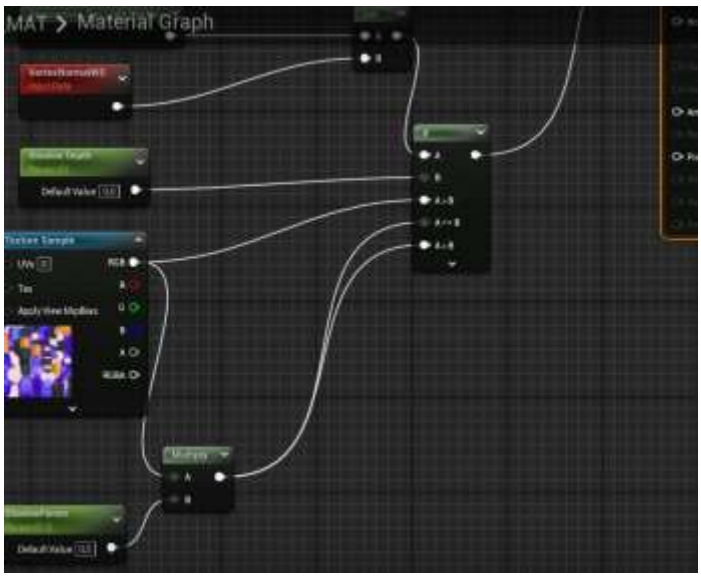

Рисунок 3.29 – Blueprint налаштування матеріалу

Потрібно зробити контур для персонажа, щоб він виділявся на екрані. Для цього необхідно створити матеріал контуру (рис 3.30).

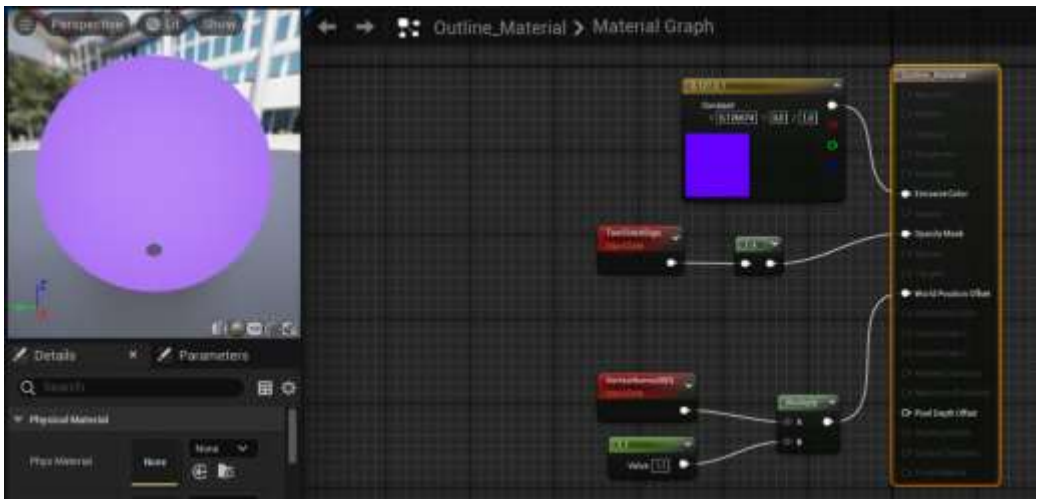

Рисунок 3.30 – Blueprint матеріалу контуру для персонажа

Відкриваємо налаштування моделі персонажа, у вкладці “overlay” додаємо створений матеріал (рис 3.31).

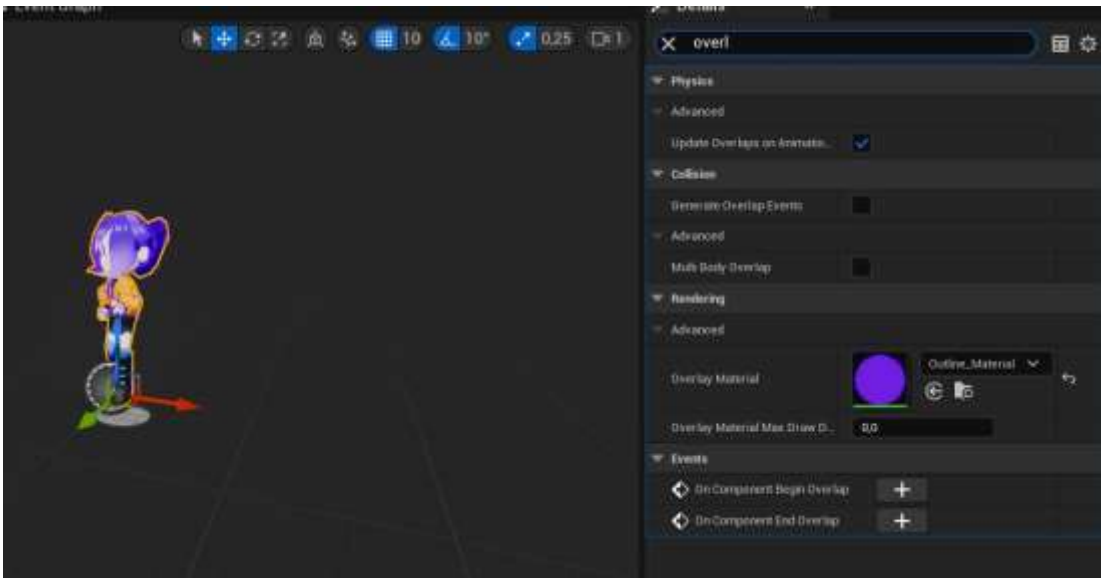

Рисунок 3.31 – Додання матеріалу для контуру персонажа

Для того щоб завантажити 2D елементи оточення потрібно спочатку застосувати налаштування Paper2D Texture Settings. Так буде коректно відображатися освітлення і прозорість текстури. Потім з такої текстури створюємо окремий спрайт, який вже можна розмістити у полі гри (рис 3.32).

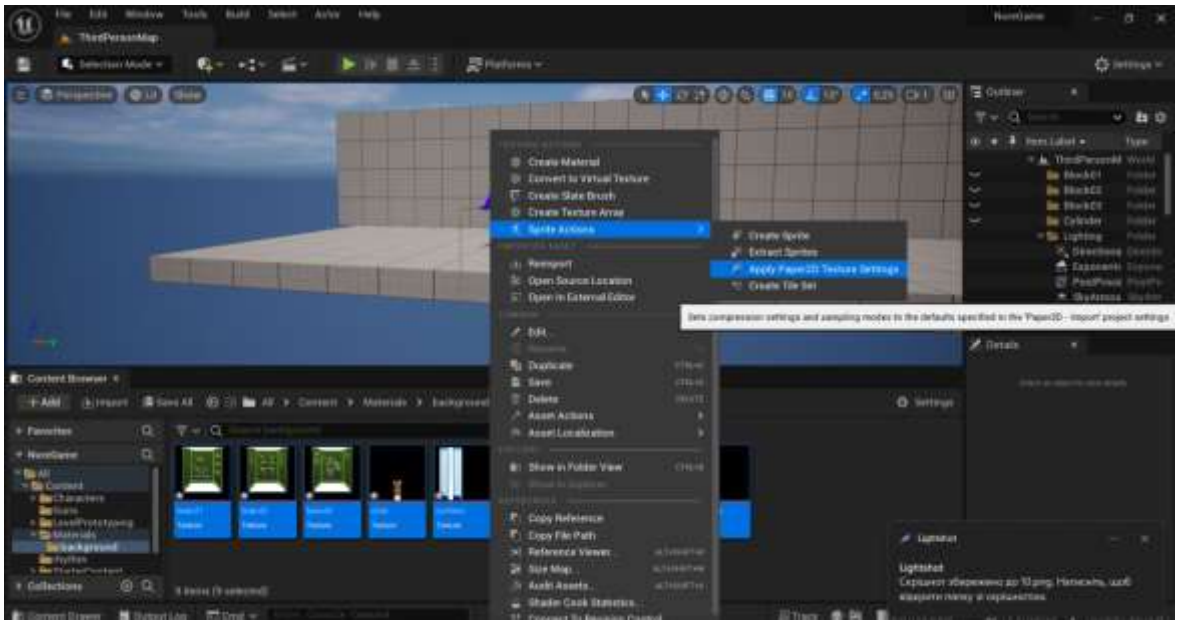

Рисунок 3.32 – Налаштування 2D елементів оточення

Налаштуємо анімації персонажа. Залишаємо всі стандарті Blueprint які використовуються для рухів персонажа у просторі та додаймо нові [39]. Додаємо

команду щоб при зупинці руху персонажа починала програватись анімація бездії (рис 3.33).

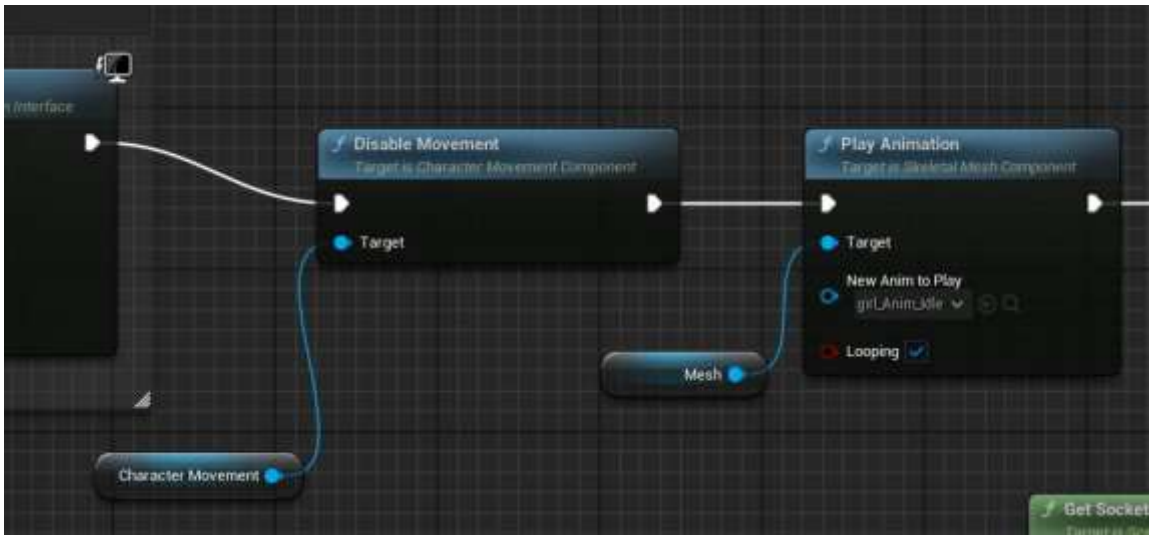

Рисунок 3.32 – Blueprint анімації бездії

Тепер створюємо новий об'єкт лівого пістолета і додаємо його у заготовлений відділ в руці персонажа. Так само повторюємо для правого (рис 3.34).

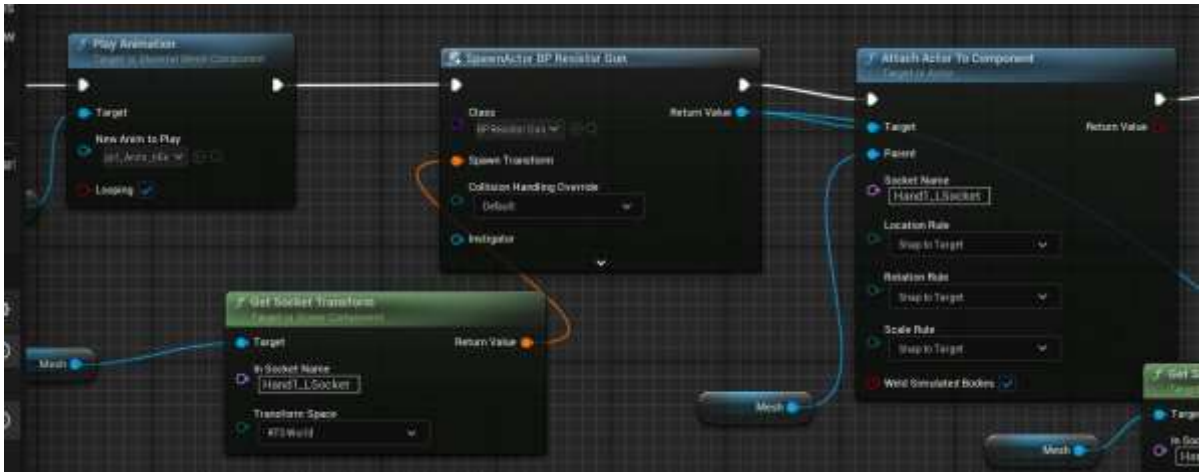

Рисунок 3.34 – Blueprint додавання предмету персонажу у обидві руки

Створюємо ворогів - ноти, які будуть вилітати. Додаємо спрайт ворога і сплайн - пряму, по якій він буде рухатись (рис 3.35).

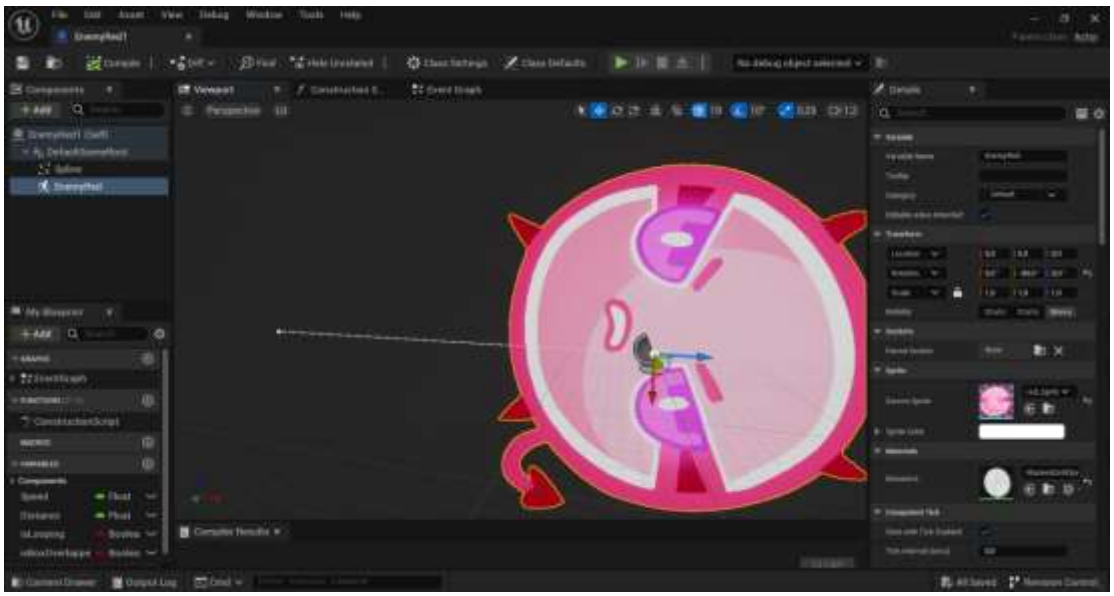

Рисунок 3.35 – Спрайт ворога і сплайн

Налаштовуємо рух ворога. Визначаємо змінні “Speed” та “Distance” і створюємо функцію плавного руху за допомогою додавання відстані за кожний проміжок часу (Delta Seconds) [40]. Встановлюємо розмір та положення ворога на рівні за допомогою “Set Relative Transform” (рис 3.36).

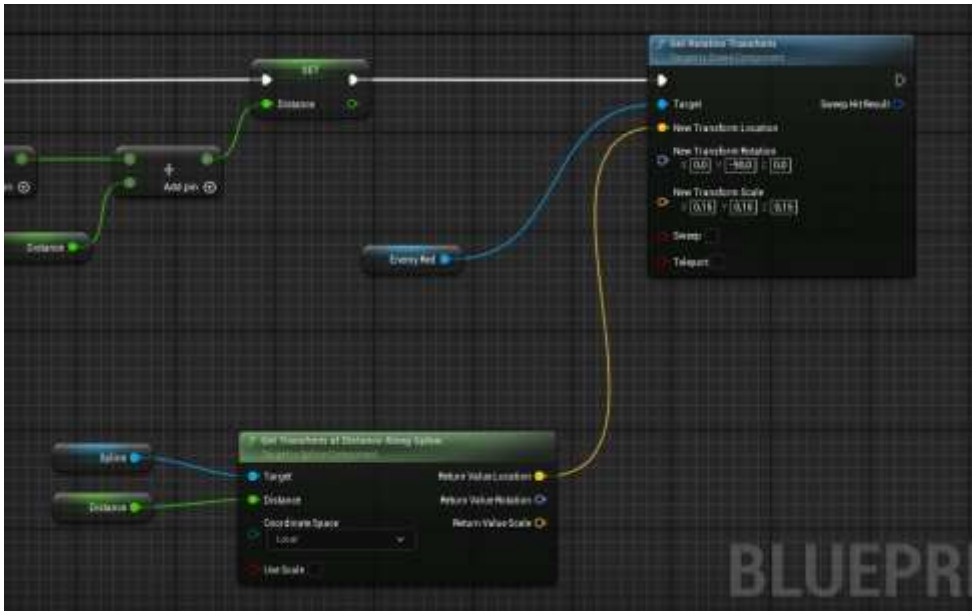

Рисунок 3.36 – Blueprint рух ворога

Теж саме повторюємо для другого ворога (рис 3.37).

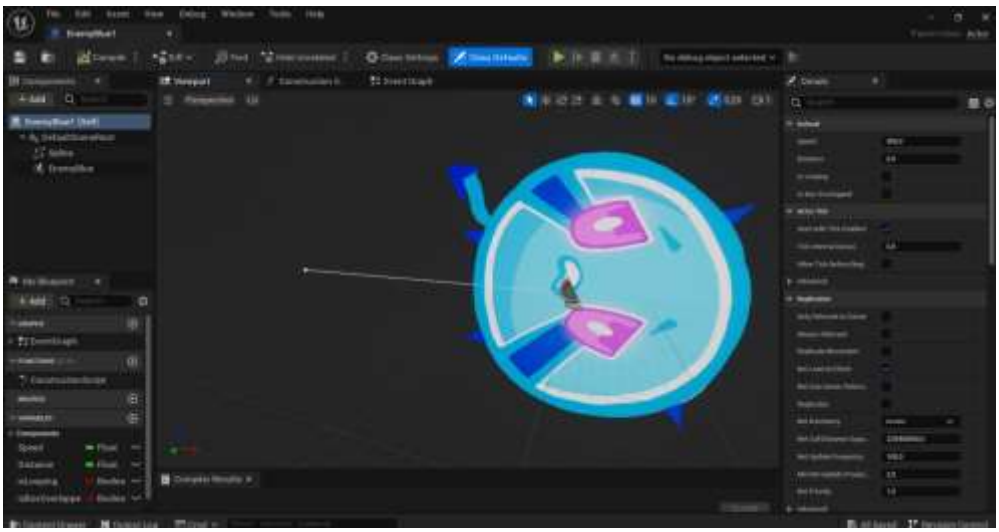

Рисунок 3.37 – Спрайт другого ворога і сплайн

Тепер завантажимо музику. Будемо використовувати “Sequencer”. Він дозволить нам створювати події в залежності від того, де ми їх розмістимо [41]. Створюємо “Level Sequence” і завантажуємо туди наш перший трек (рис 3.38).

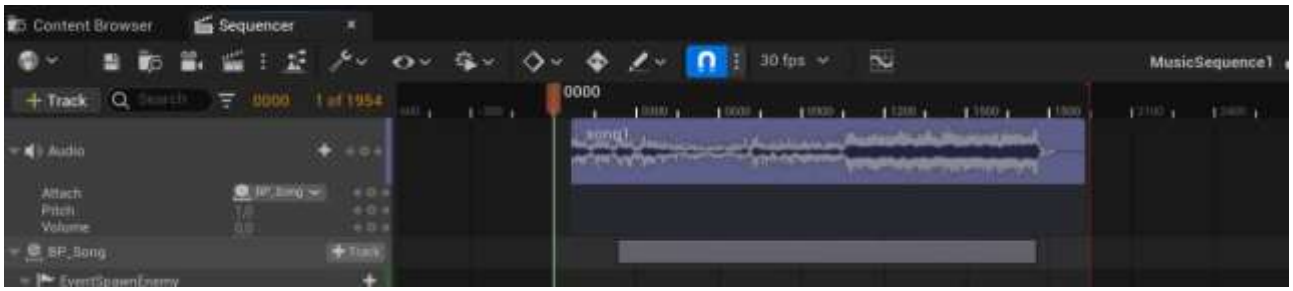

Рисунок 3.38 – Перший трек у “Level Sequence”

В залежності від ритму будуть з’являтися рожеві та блакитні ноти, тож нам треба створити 2 різні події – “EventSpawnEnemyRed” та “EventSpawnEnemyBlue” (рис 3.39).

Рисунок 3.39 – Blueprint подія поява нот

Далі підбираємо ноти і розміщуємо ці події на “Sequencer” (рис 3.40).

Рисунок 3.40 –Ноти розміщені у “Sequence”

Також потрібно, щоб протягом пісні змінювалися емоції ворога. Створимо Blueprint ворога (рис 3.41).

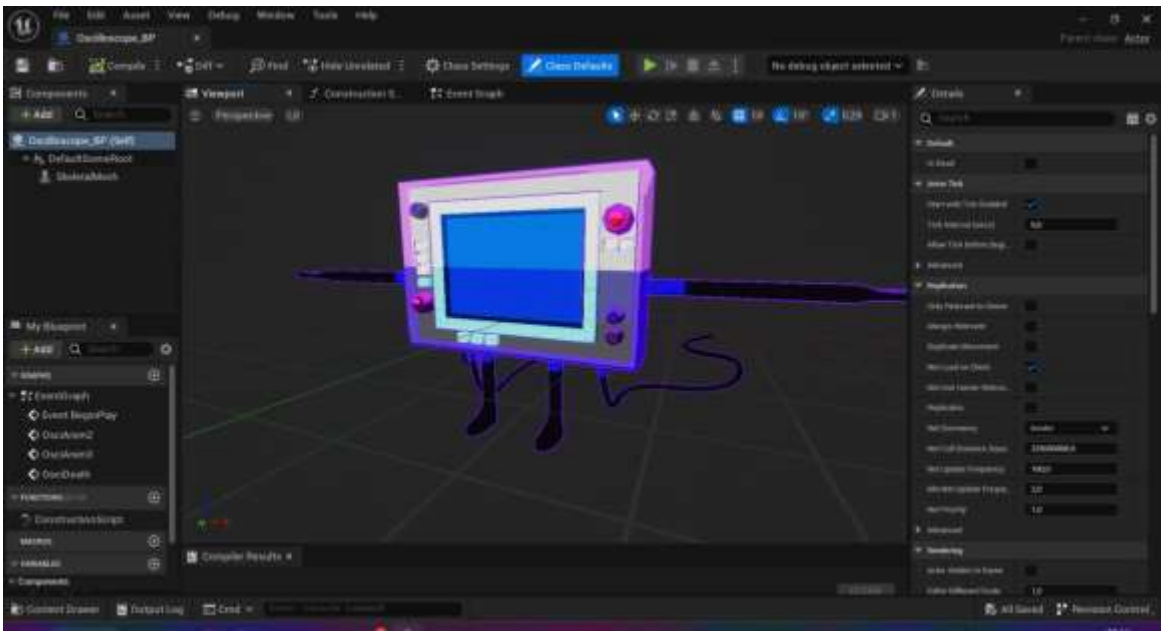

Рисунок 3.41 – Модель осцилографа

Створимо анімації для обличчя осцилографа із заготовлених текстур (рис 3.42).

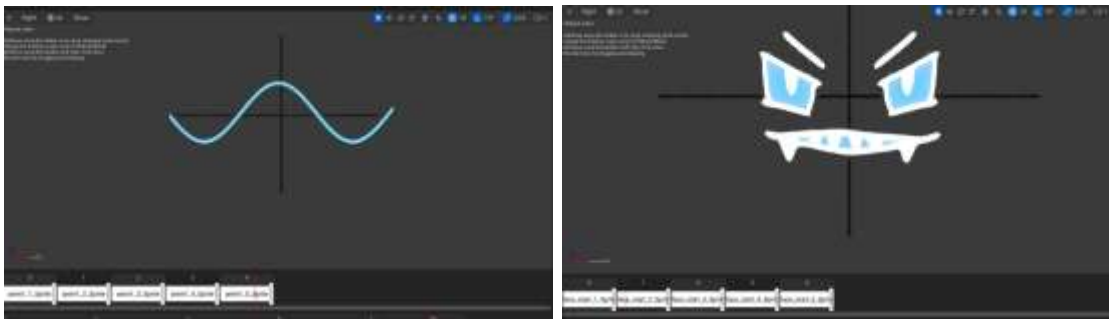

Рисунок 3.41 – Анімація обличчя осцилографа

Зробимо Blueprint для першої анімації обличчя осцилографа (рис 3.42). Та анімацію відпочинку (рис 3.43).

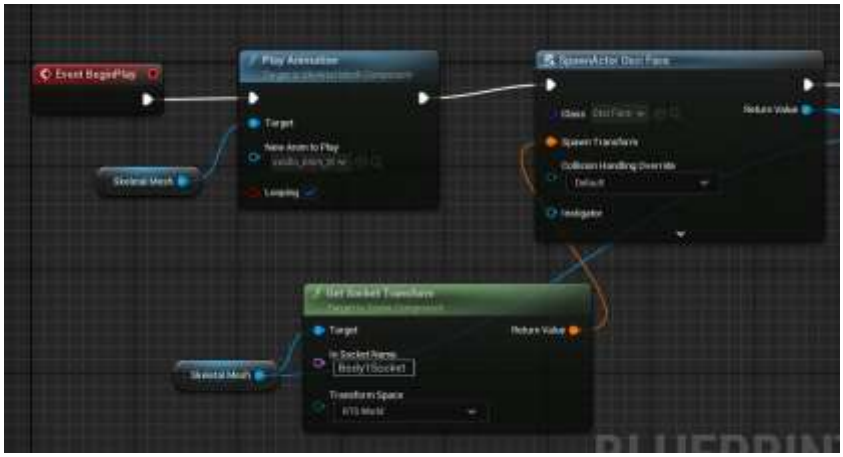

Рисунок 3.42 – Blueprint анімація першого обличчя осцилографа

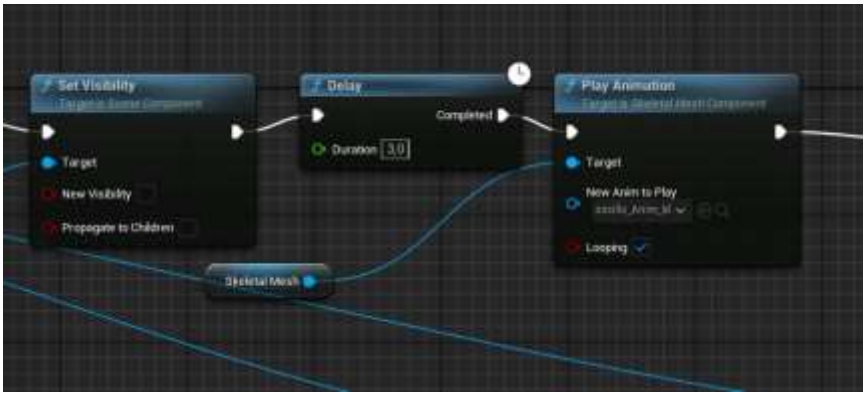

Рисунок 3.43 – Blueprint анімація відпочинку осцилографа

Створюємо подію, яка буде змінювати перше обличчя на друге [42]. Тут по запуску цієї події ми просто ховаємо старе обличчя і відкриваємо нове, якщо ворог ще живий за допомогою “Set Visibility” (рис 3.44).

Рисунок 3.44 – Blueprint анімація другого обличчя осцилографа

Так само повторюємо для 3 і останньої анімації смерті (рис 3.45).

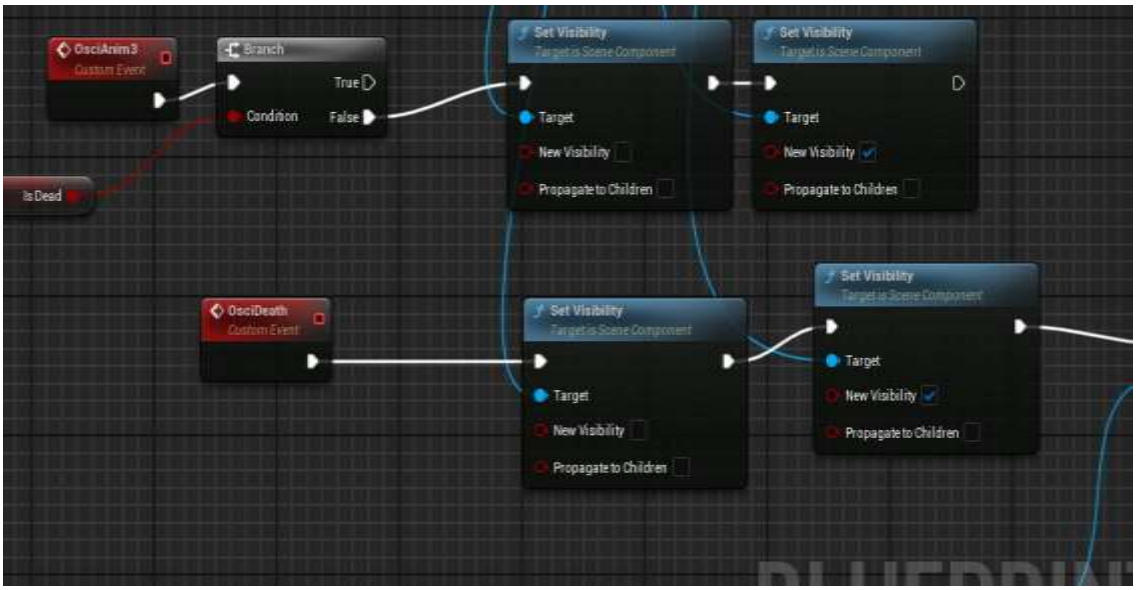

Рисунок 3.45 – Blueprint анімація третього обличчя осцилографа

Після смерті відмічаємо змінну “isDead” (рис 3.46).

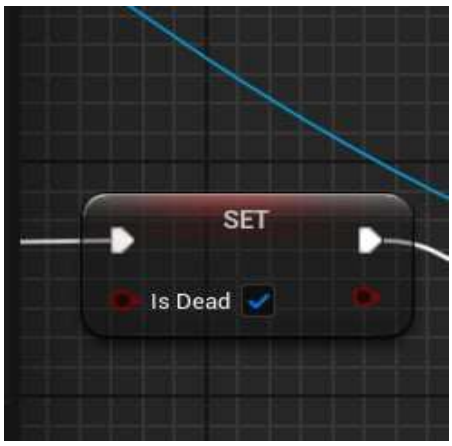

Рисунок 3.46 – Blueprint змінна “isDead”

Також нам потрібно буде зберегти рахунок. Поки що ми не набираємо очки, а лише будемо переносити значення між рівнями [43]. Для цього створимо “GameInstance ScoreVar”, в якому будуть зберігатися різні потрібні значення. Додаємо туди змінні, які нам знадобляться (рис 3.47).

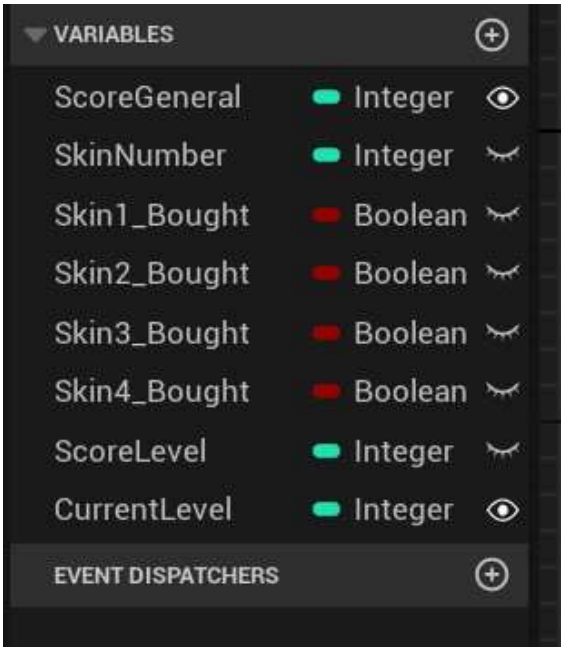

Рисунок 3.46 – Панель “GameInstance ScoreVar”

Отже, рахуємо очки, що герой назбирав на рівні “Score Level” та додаємо їх до загального значення “Score General”. Після підрахунку відкриваємо рівень з результатом “LevelFinishedMap”.

Не забуваємо додати події для осцилографа на “Sequencer” (рис 3.47).

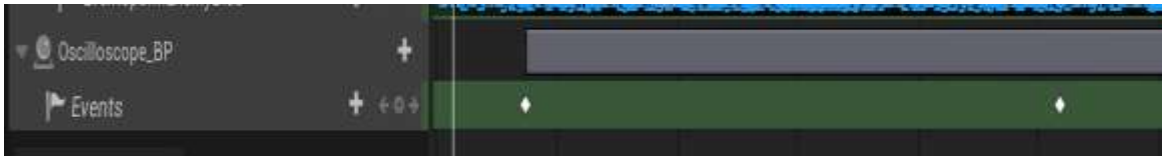

Рисунок 3.47 – Події для осцилографа на “Sequencer”

Тепер створимо Blueprint для того, щоб підраховувати очки. Назвемо його BP_Box_Overlap, бо тут буде взаємодія з тріггер-боксом, тобто, зоною, яка відстежує усі зміни [44].

Додаємо в Blueprint 4 бокси - 2, які будуть відстежувати зону входження ноти в потрібний проміжок, а 2, які будуть видаляти ноти. Також додаємо елемент інтерфейсу, який підкаже, де саме треба натискати на кнопки (рис 3.48).

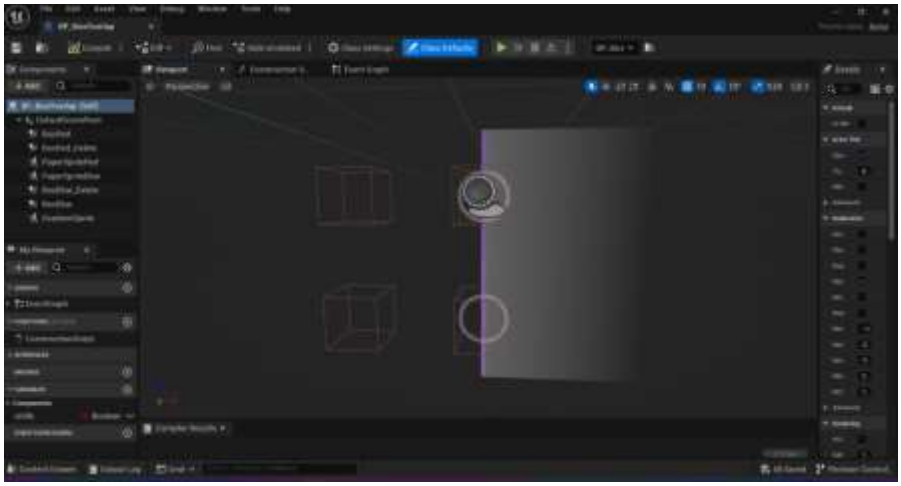

Рисунок 3.48 – Взаємодія з тріггер-боксом

По запуску події, створюємо інтерфейс з кнопками, які повинен натиснути користувач. Поки що робимо заготовку, потім налаштуємо його візуально.

Для відстеження натискання використовуємо елемент “Gate” [45]. Він видає позитивний сигнал на виході, якщо обидва активні входи “Enter” та “Open”. При чому “Open” при першому сигналі на ньому залишається відкритим, а “Enter“ появляється та одразу пропадає [46]. Таким чином ми відкриваємо “Gate”, коли нота входить в зону боксу та не даємо сигнал далі допоки користувач не натисне відповідну кнопку інтерфейсу.

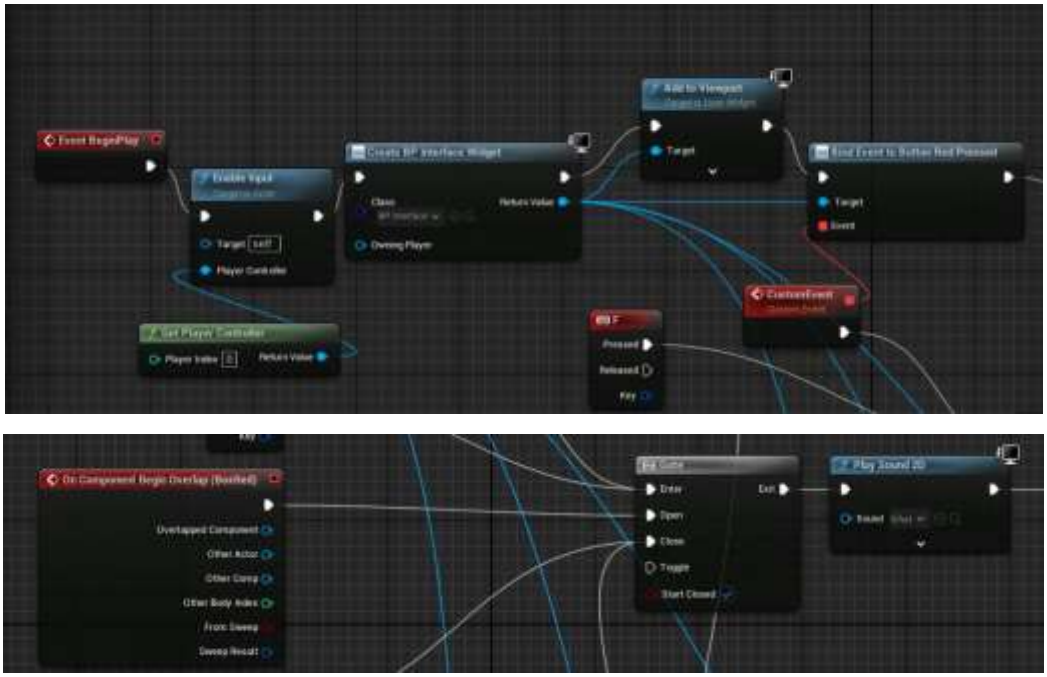

Рисунок 3.49 – Blueprint кнопок

Після успішного натискання видаляємо ноту (рис 3.50).

Рисунок 3.50 – Blueprint видалення ноти

Також додаємо до створеної змінної “ScoreVar” 10 очків, запускаємо анімацію пострілу. Використовуємо “Retriggerable Delay” для того, щоб можна було не чекати на завершення анімації [47].

По виходу ноти із зони боксу закриваємо гейт, тобто, користувач вже не зможе набрати очки за цю ноту (рис 3.51).

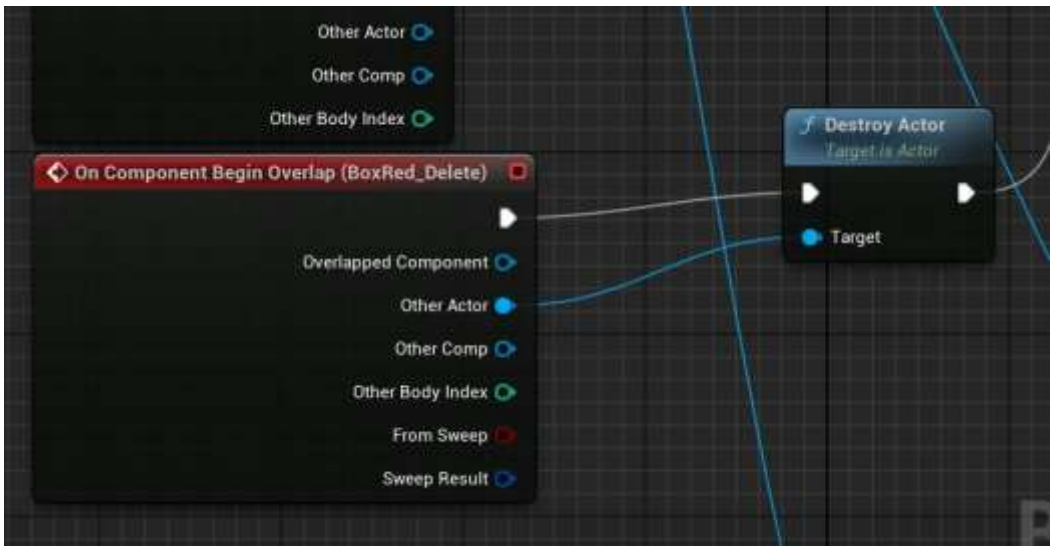

Рисунок 3.51 – Blueprint закриття гейту

Тепер, коли ця нота, що залишилась досягне другого боксу для видалення нот, знищуємо її за допомогою “Destroy Actor” (рис 3.52).

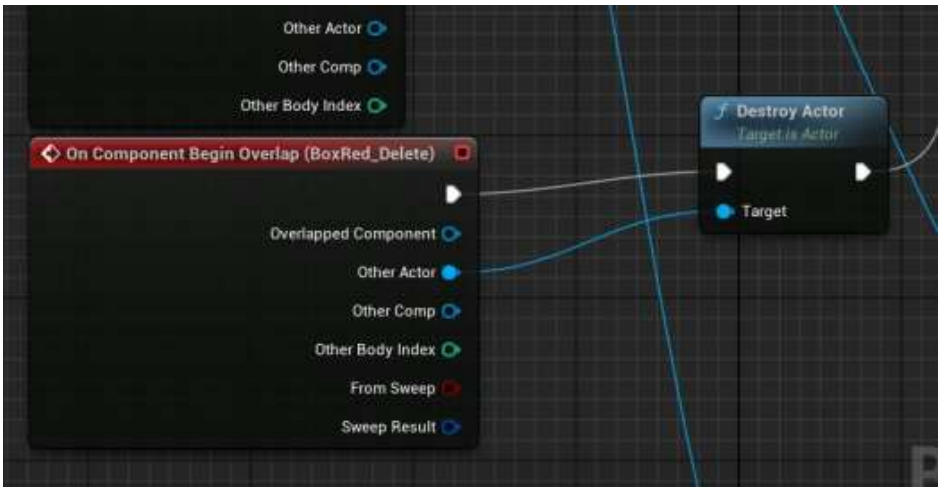

Рисунок 3.52 – Blueprint знищеня ноти за допомогою “Destroy Actor”

Також одразу налаштовуємо текст на початку рівня, який буде відраховувати час (рис 3.53).

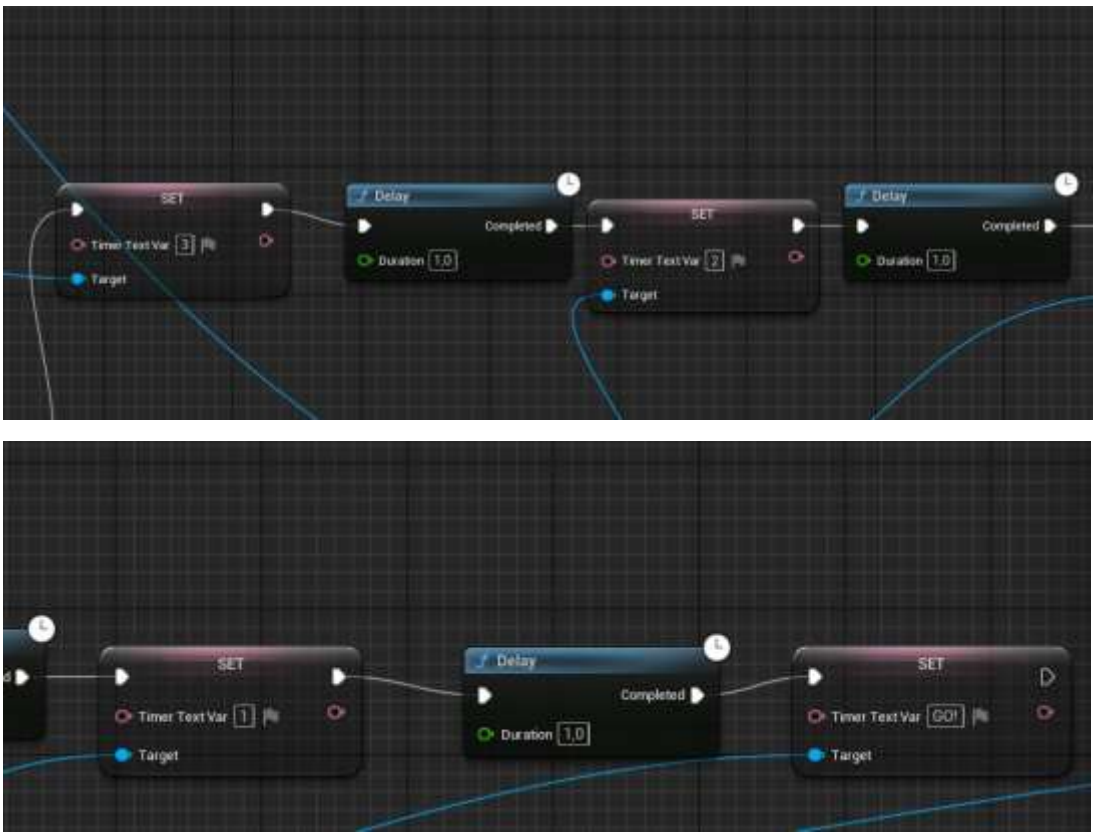

Рисунок 3.53 – Blueprint відрахування часу

Повертаємо до Blueprint інтерфейсу. Додаємо до нього усі потрібні кнопки (рис 3.54).

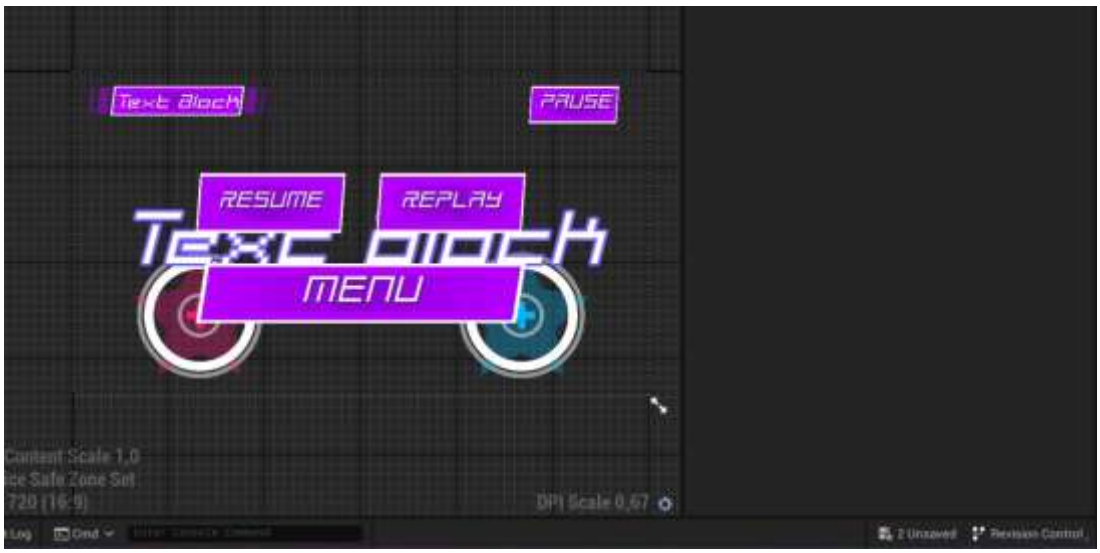

Рисунок 3.54 – Інтерфейс гри

Додаємо події для рожевої і блакитної кнопки, які будуть подаватися на “Gate” для підрахунку очків (рис 3.55).

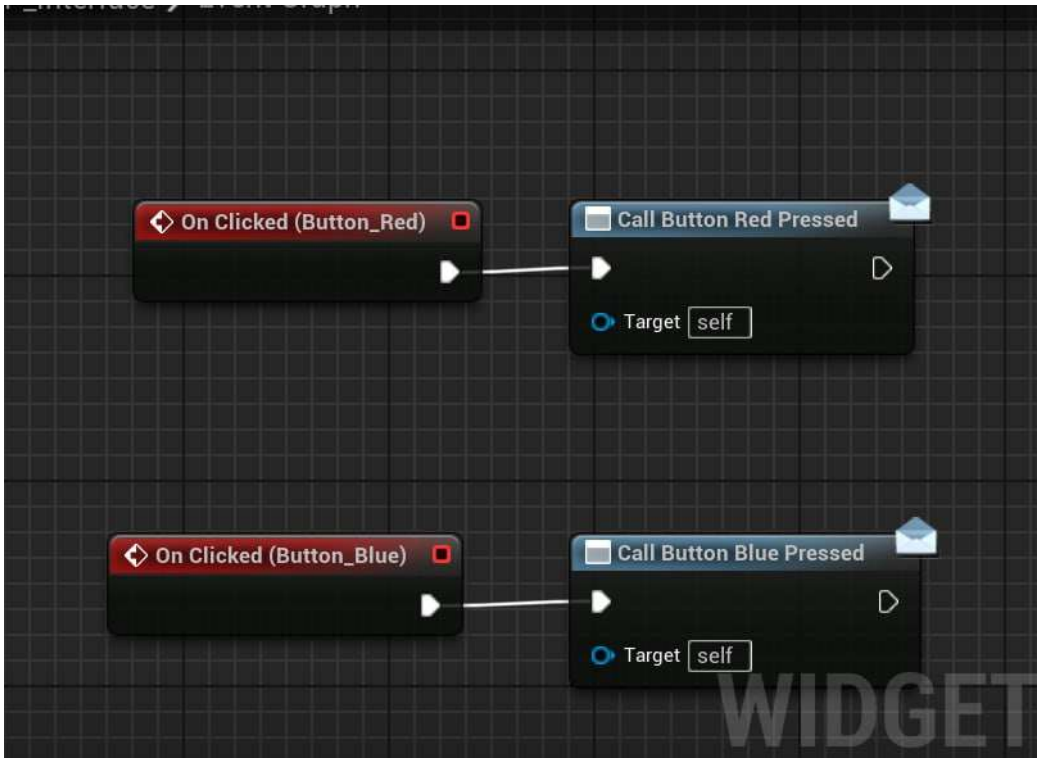

Рисунок 3.55 – Blueprint підрахунок очків

Після створення інтерфейсу нам потрібно перевірити, який саме рівень запустили, бо усі персонажі і вороги залишаються однаковими, змінюється лише звук. Із нашого “Game Instance” беремо значення рівня і запускаємо відповідний звук (рис 3.56).

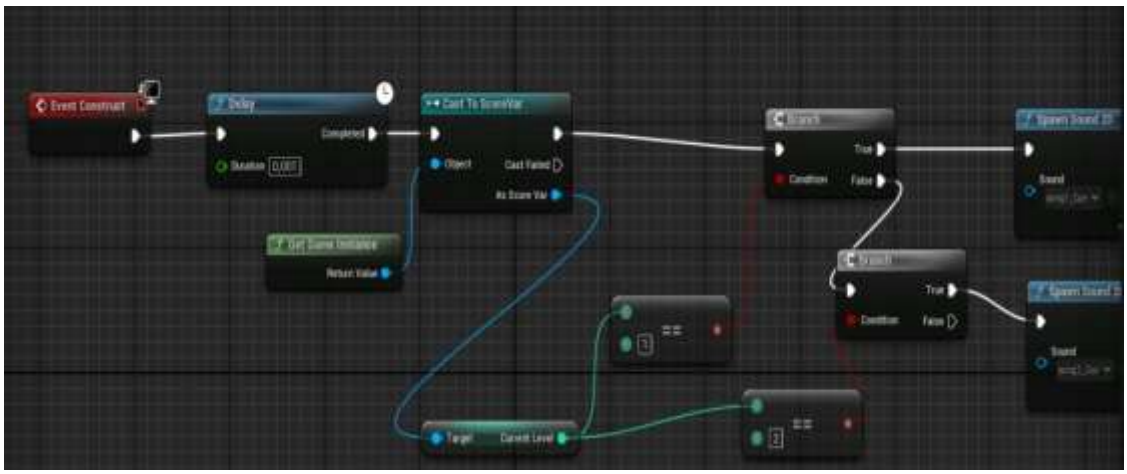

Рисунок 3.56 – Blueprint рівня

Далі по натисканню на кнопку Паузи відкриваємо меню, де будуть кнопки “Resume”, “Replay”, “Menu” (рис 3.57).

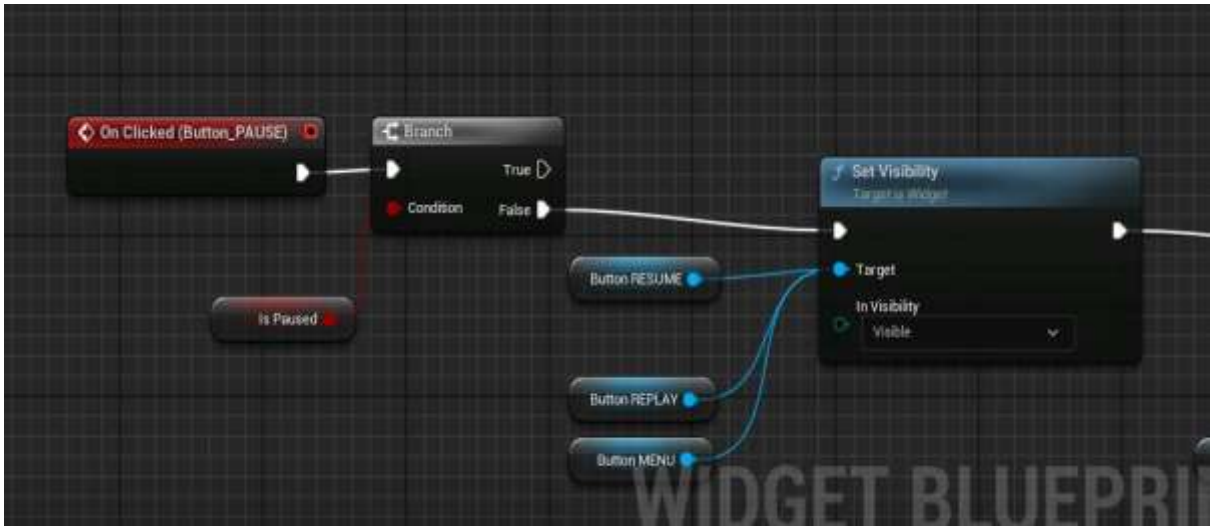

Рисунок 3.57 – Blueprint кнопок меню

Ставимо звук, гру та змінну на паузу (рис 3.58).

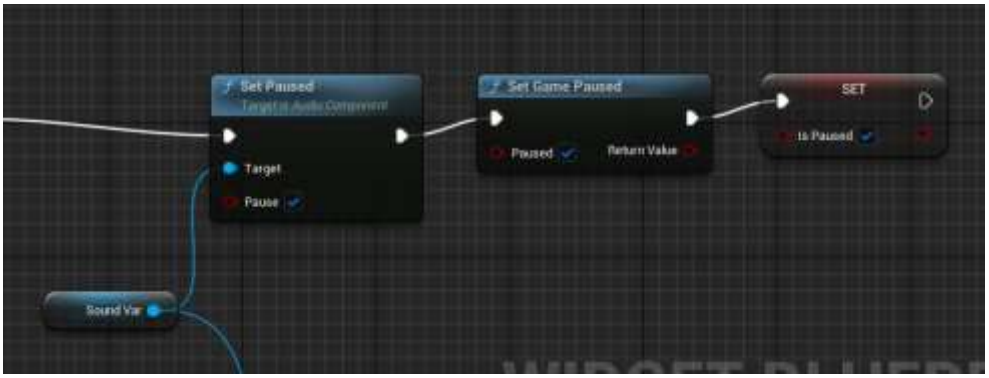

Рисунок 3.58 – Blueprint паузи

Якщо гравець натисне “Resume”, гра продовжиться. Ховаємо меню паузи і знімаємо звук, змінну і саму гру з паузи (рис 3.59).

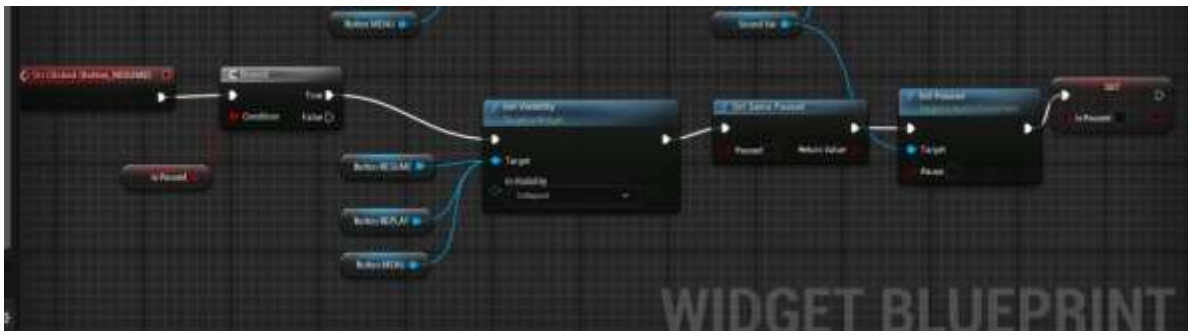

Рисунок 3.59 – Blueprint “Resume”

Якщо гравець натисне “Replay”, то рівень запускається спочатку (рис 3.60).

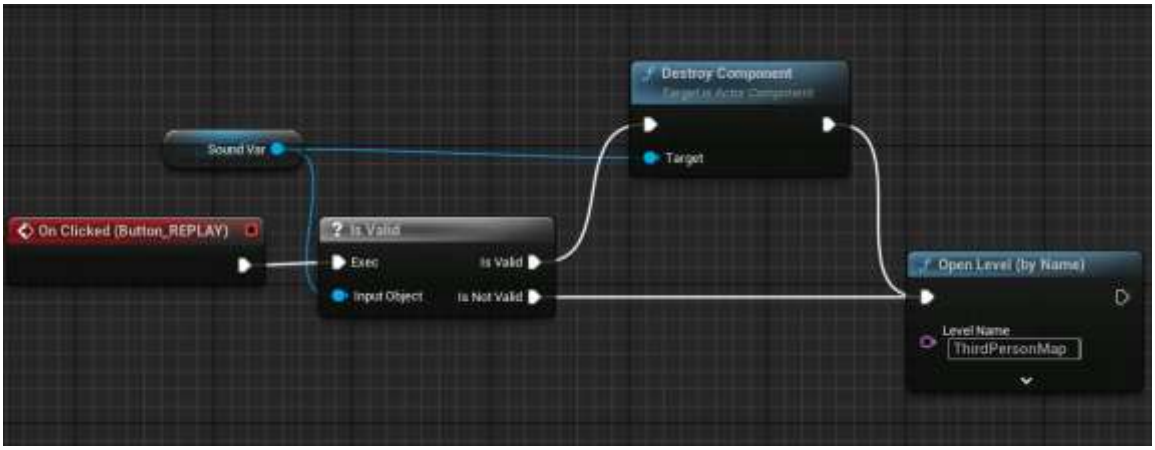

Рисунок 3.60 – Blueprint “Replay”

Якщо гравець натисне “Menu”, то переходимо на окремий рівень з меню (рис 3.61).

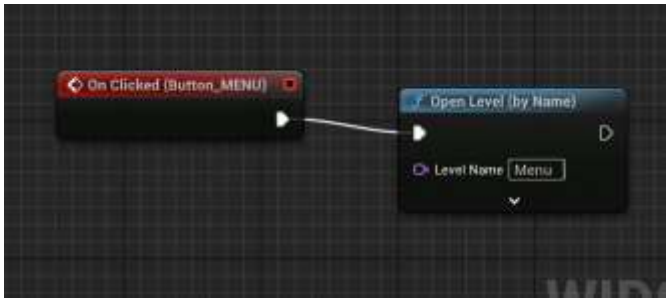

Рисунок 3.61 – Blueprint “Menu”

У Blueprint самого рівня також прописуємо функції.

По запуску спочатку знову дізнаємося, який саме рівень відкрили через “Game Instance” (він буде обиратися на окремому меню). Запускаємо відповідний звукову послідовність [48]. Треба відмітити, що в інтерфейсі і тут ми запускаємо різні звуки, бо “Sound” - це просто звук, який буде чути гравець. А “Sequence” - це та послідовність подій, яку ми налаштували раніше, вона запускається трохи раніше, щоб ноти встигли долетіти до заданої відмітки (рис 3.62).

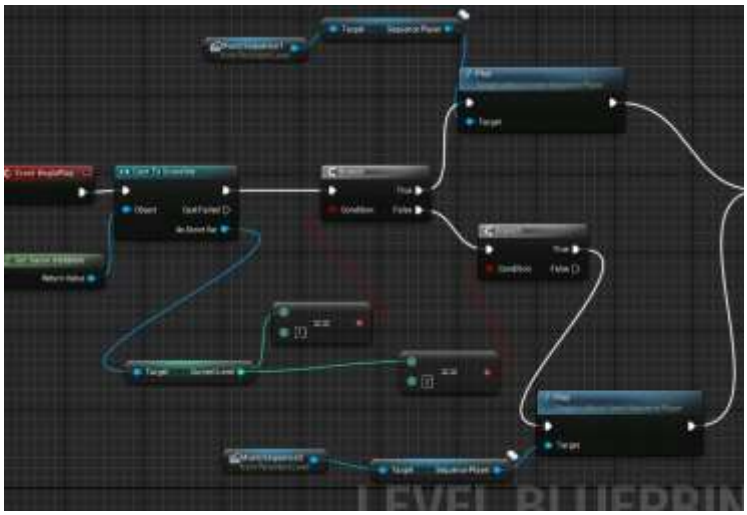

Рисунок 3.62 – Blueprint рівня

Перевіряємо, який скін обрано (рис 3.63).

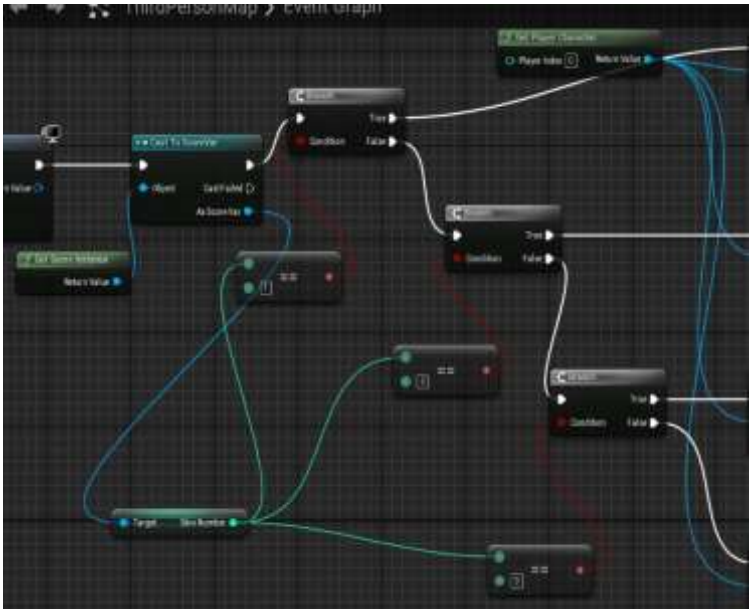

Рисунок 3.63 – Blueprint перевірка скіна

В залежності від номеру скіна встановлюємо його на персонажа, змінюючи матеріал і контур (рис 3.64).

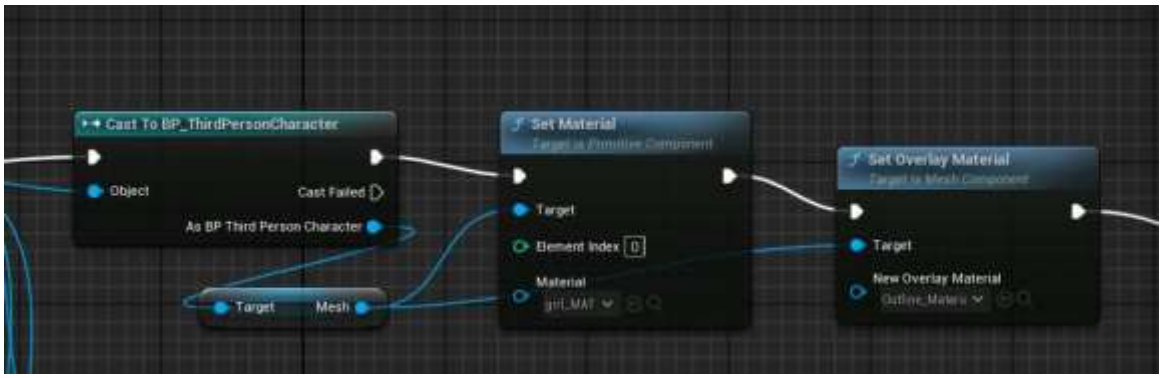

Рисунок 3.64 – Blueprint встановлення скіна

Встановлюємо вид із камери (рис 3.65).

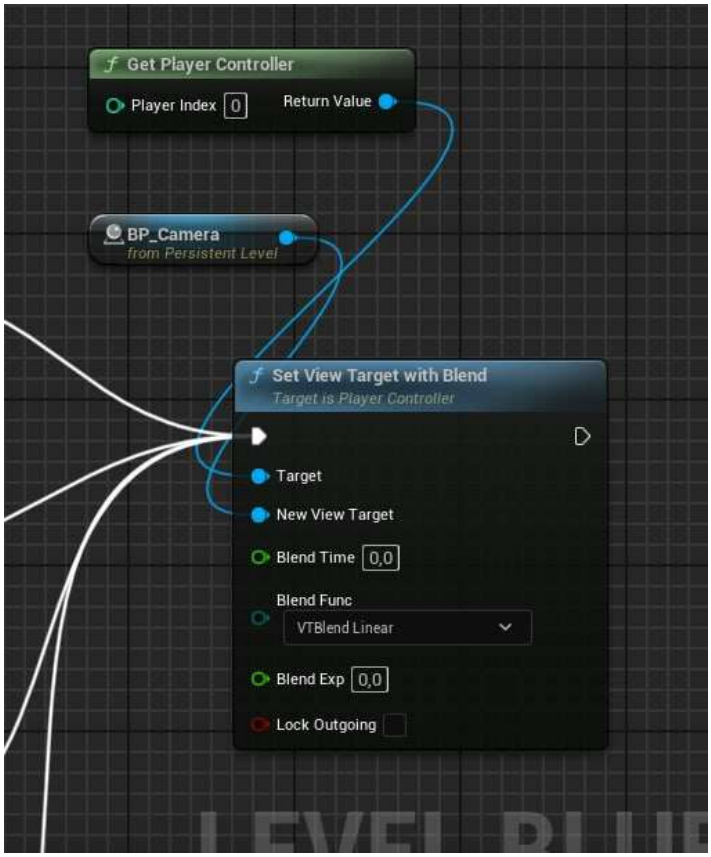

Рисунок 3.65 – Blueprint встановлення виду з камери

Створюємо окремий рівень початкового меню [48]. В блупрінтах рівня створюємо лише інтерфейс меню (рис 3.66).

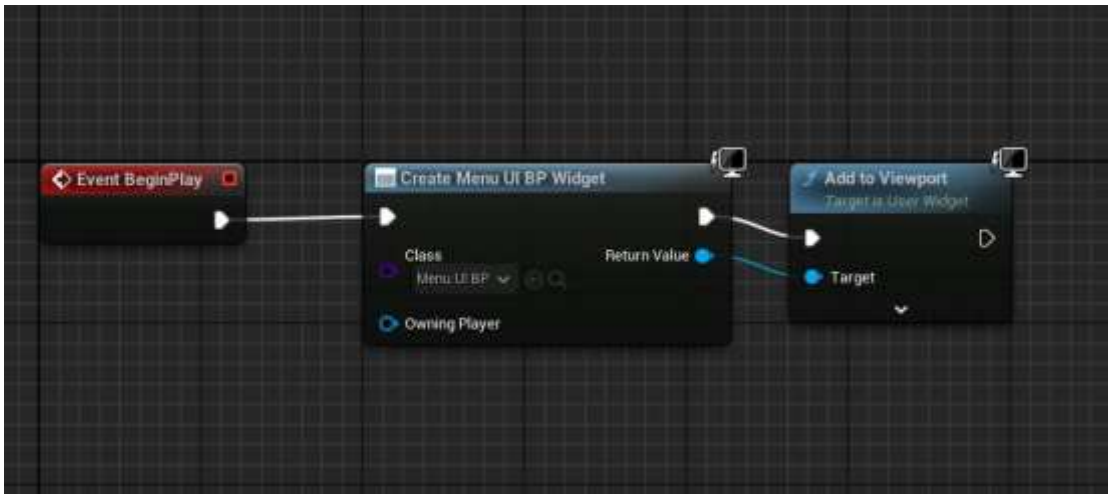

Рисунок 3.66 – Blueprint окремого меню

На цей інтерфейс меню розміщуємо намальований фон та кнопки (рис 3.67).

Рисунок 3.67 – Розміщення інтерфейсу

По натисканню на “PLAY” буде запускатися рівень, де обирається пісня (рис 3.68).

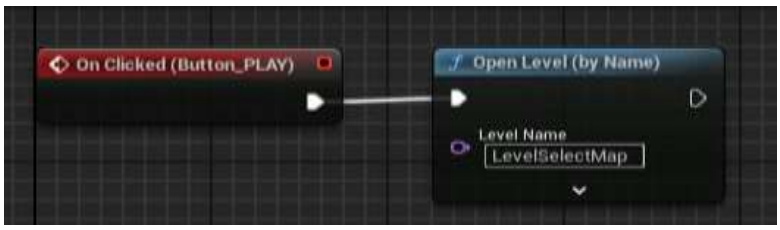

Рисунок 3.68 – Blueprint кнопки “PLAY”

На “EXIT” користувач виходить з гри (рис 3.69).

Рисунок 3.69 – Blueprint кнопки “EXIT”

На “SKIN OPTIONS” - відкриваються покупка та вибір скіна (рис 3.70).

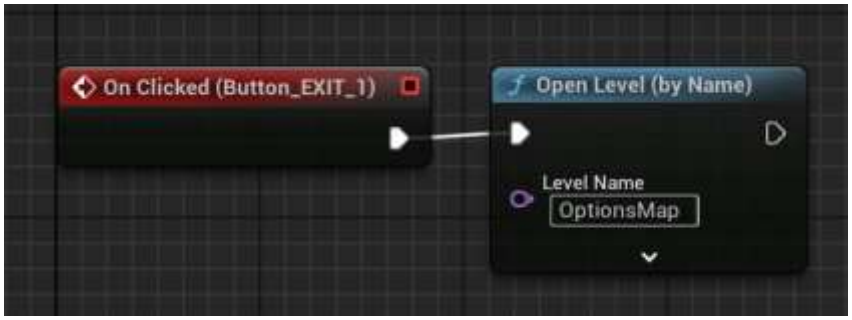

Рисунок 3.70 – Blueprint кнопки вибору скіна

Налаштовуємо рівень вибору пісні. У Blueprint рівня теж створюємо тільки інтерфейс (рис 3.71).

Рисунок 3.71 – Blueprint вибору пісні

Додаємо інтерфейс вибору пісні (рис 3.72).

Рисунок 3.72 – Інтерфейс вибору пісні

При натисканні на лівий “PLAY” - в “Game Instance” буде записуватися значення рівня 1, а на правий - 2. Мапа рівня запускається одна [49]. На кнопку “BACK” відкриваємо меню (рис 3.73).

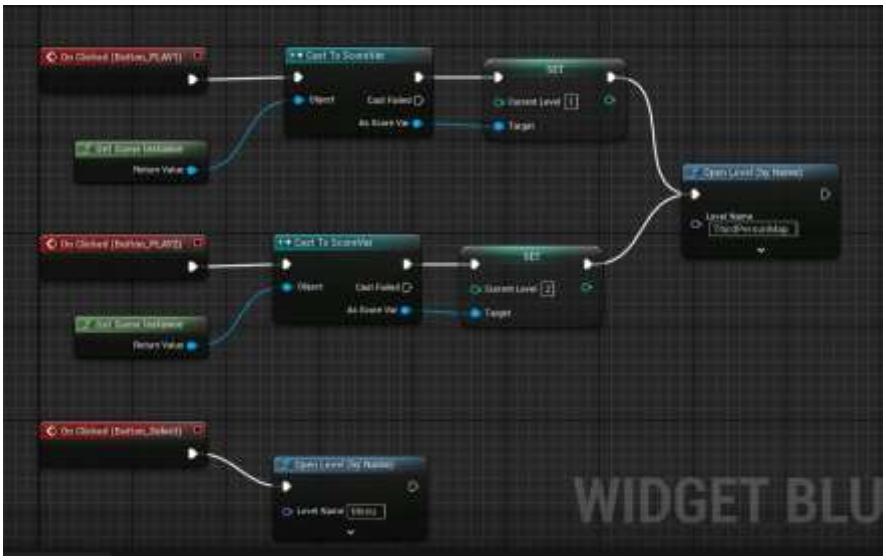

Рисунок 3.73 – Blueprint запуску рівня

Налаштовуємо рівень покупки та вибору скіна [50]. Створюємо інтерфейс в блупрінтах рівня (рис 3.74).

Рисунок 3.74 – Інтерфейс вибору скіна

Щоб змінювався текст, його потрібно прив'язати до відповідної функції [51]. Кількість очків прив'язуємо до GetScore (рис 3.75).

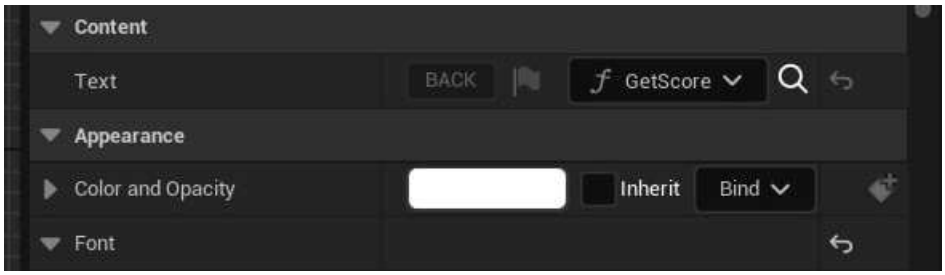

Рисунок 3.75 – Налаштування тексту

Функція “GetScore” відображає загальний рахунок із “Game Instance” і перетворює його в звичайний текст (рис 3.76).

Рисунок 3.76 – Blueprint функції “GetScore”

Текст під зображенням скіна прив'язуємо до “GetText” [52]. Якщо у гравця цей скін вже є і він зараз у нього активний - текст буде “Selected”. Якщо не активний, то “Select” (рис 3.77).

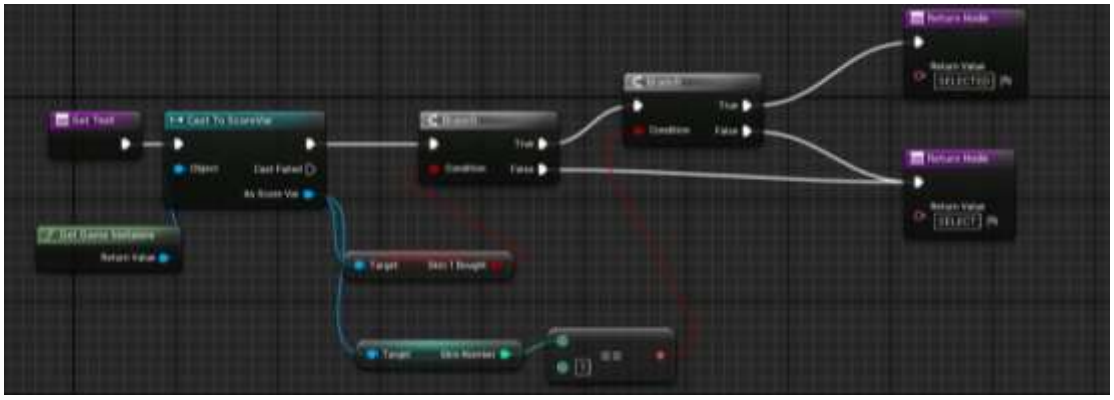

Рисунок 3.77 – Blueprint вибору скіна

Оскільки перший скін безкоштовний, то його вже не можна купити і у нього немає ціни [53]. А ось інші скіни мають задану вартість і тоді якщо у гравця вже є скін, то кнопки будуть такі самі як і для першого [54]. А якщо він ще його не купив, то буде додатковий текст із ціною скіна (рис 3.78).

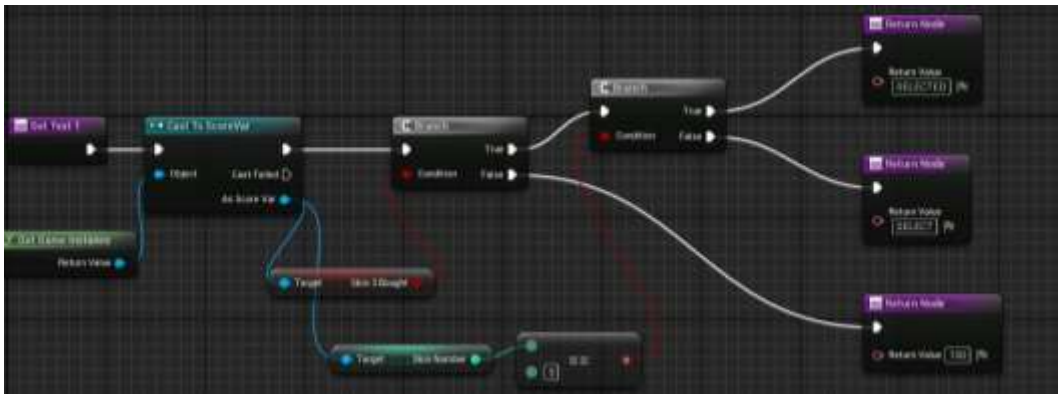

Рисунок 3.77 – Blueprint інших скінів

Підраховуємо тепер очки при покупці. При натисканні на кнопку скіна спочатку перевіряємо, чи купили його раніше [55]. Якщо так, то перевіряємо чи стоїть він зараз. Якщо знову так, то нічого не міняжмо, значить він вже активний [56]. Якщо ж ні, то в “Game Instance” прописуємо обраний номер скіна (рис 3.78).

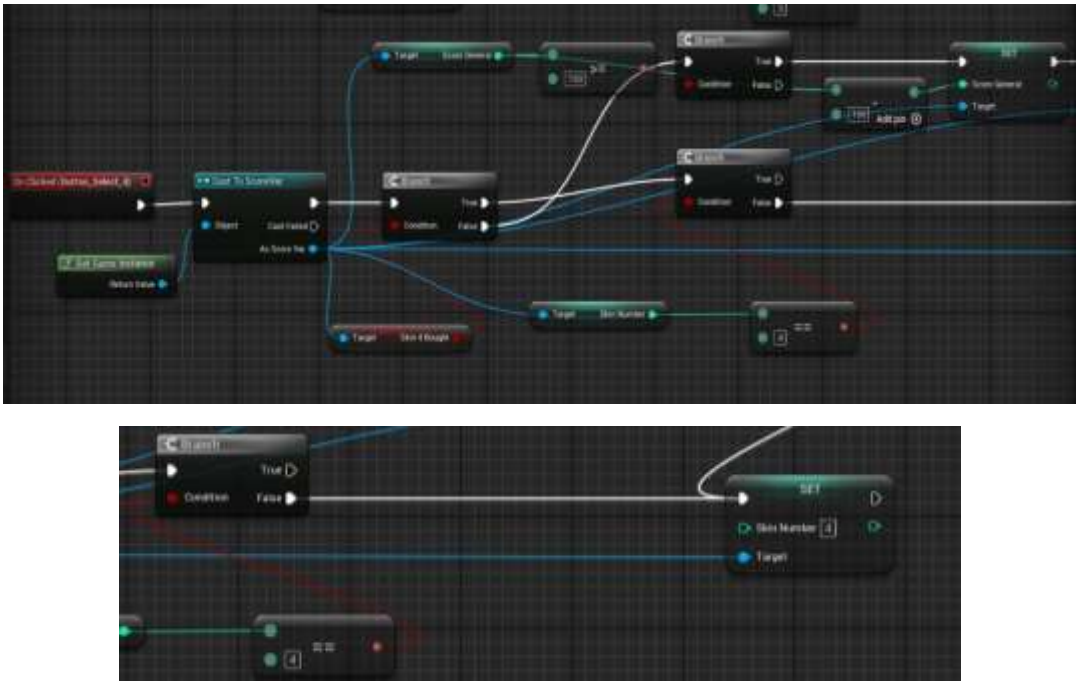

Рисунок 3.78 – Blueprint покупки скінів

Якщо ж скін у гравця не куплений, то перевіряємо, чи більше в нього очків, ніж стільки, скільки коштує скін [57]. Якщо менше - нічого не відбувається, гравець не покупає скін. Якщо більше, то від загального рахунку віднімаємо задану вартість і відмічаємо, що скін купили [58]. Встановлюємо його одразу (рис 3.78).

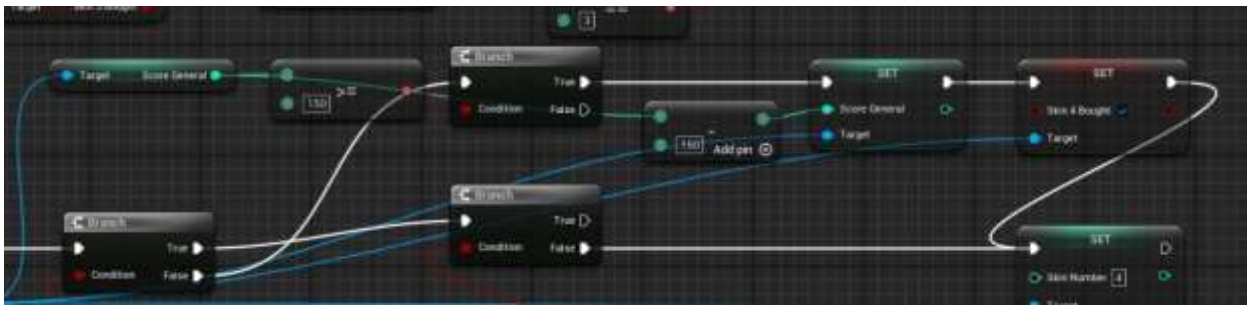

Рисунок 3.78 – Blueprint перевірка очків гравця

Так само повторюємо з усіма скінами.

Останній рівень - результати гри. Так само як і в інших, створюємо інтерфейс в Blueprint рівня (рис 3.79).

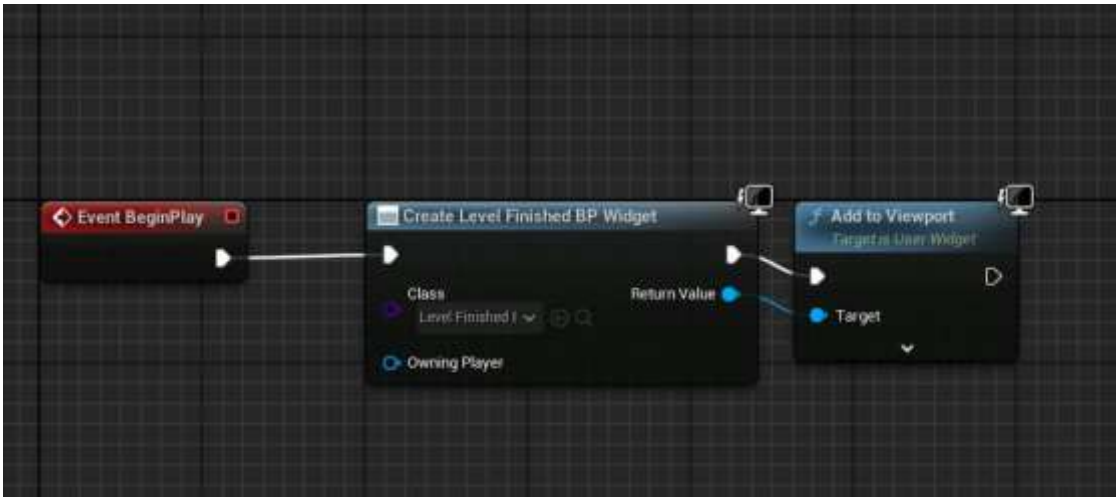

Рисунок 3.79 – Blueprint інтерфейс рівня

Створюємо інтерфейс результату рівня. Тут буде змінюватися текст після “Score”, буде відображатися, скільки очків набрав гравець (рис 3.80).

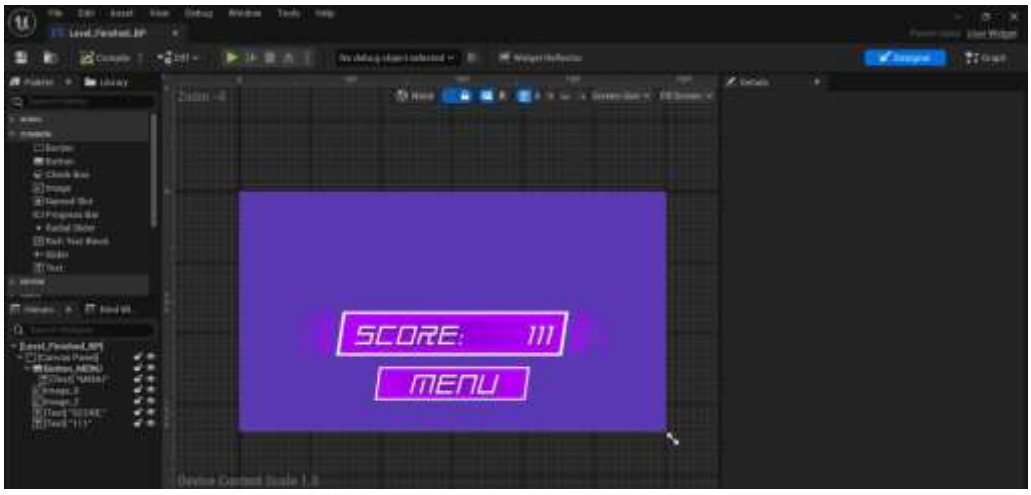

Рисунок 3.80 – Інтерфейс рівня

Підв'язуємо текст до функції “GetText” (рис 3.81).

Рисунок 3.81 – Підв’язка функції “GetText”

Ця функція бере значення рівню, але не загальне, а тимчасове, яке гравець набрав за останнє своє проходження пісні (рис 3.82).

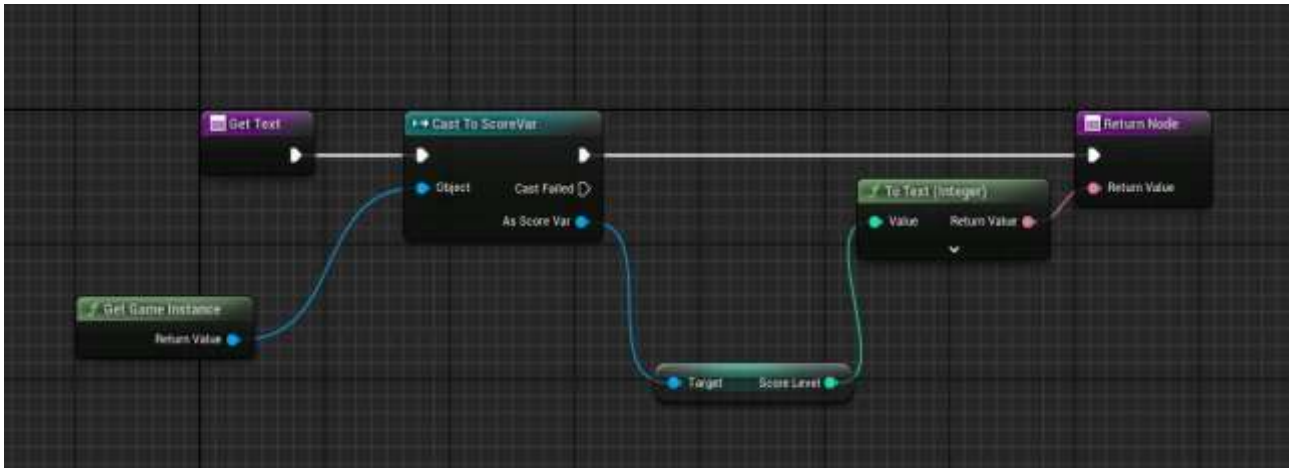

Рисунок 3.82 – Blueprint рівня

Щоб зібрати гру для Android, треба експортувати її як APK файл. Завантажуємо SDK, NDK файли та Java, необхідні для коректної компіляції [59]. Вказуємо завантажені версії та шляхи до них у налаштуваннях проекту (рис 3.83).

Рисунок 3.83 – Експорт гри на Android

Також необхідно створити індивідуальний ключ. Він створюється за допомогою завантаження коду для командної строки, де ми вказуємо назву

проекту, компанію та інші данні для дистрибуції гри [60]. Без нього гра на Android не зкомпілюється (рис 3.84).

Рисунок 3.84 – Створення ключа для експорту

Вписуємо назву гри (рис 3.85).

Рисунок 3.85 – Вказання назви гри

Обираємо оптимальні налаштування пакування (рис 3.86).

Рисунок 3.86 – Налаштування пакування

Завантажуємо іконки ігри (рис 3.87).

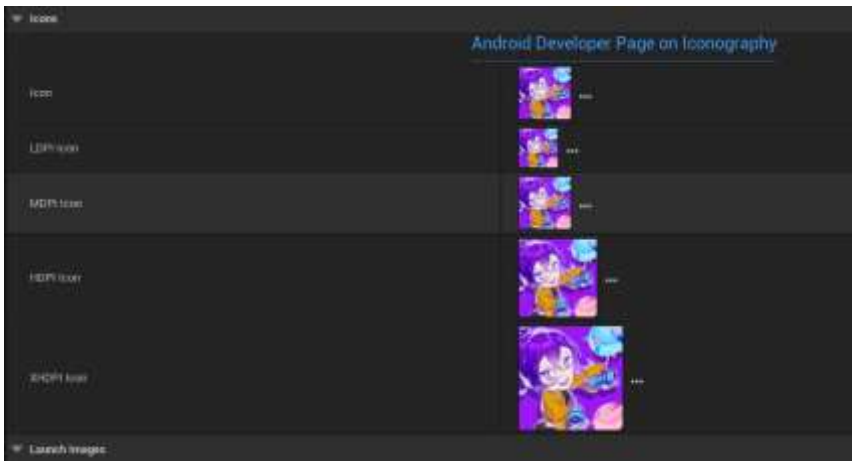

Рисунок 3.87 – Завантаження іконки гри

Після правильних параметрів пакування для Android побачимо вгорі зелене повідомлення, що файли можна записати (рис 3.88).

Рисунок 3.88 – Повідомлення про дозвіл на запис файлів

Переходимо до налаштувань самого проекта. В налаштуваннях “Maps & Modes” обираємо стартову карту меню (рис 3.89).

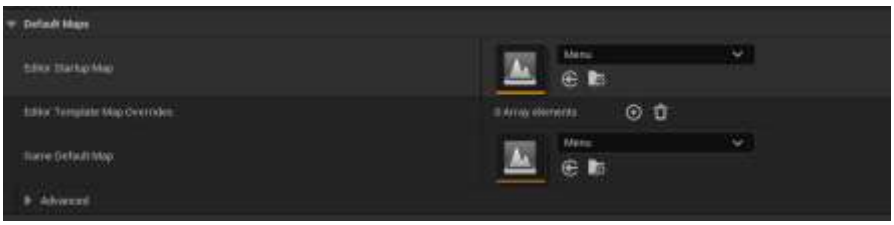

Рисунок 3.89 – Обрання стартової карти меню

Перевіряємо, щоб наша “Game Instance”, де зберігаються усі змінні була у списку (рис 3.90).

Рисунок 3.90 – Перевірка “Game Instance”

У налаштуваннях “Packaging” обираємо “Build Configuration” – “Shipping” [61]. Так гра зкомпілюється як для звичайного гравця без тестового режиму і режиму розробника (рис 3.91).

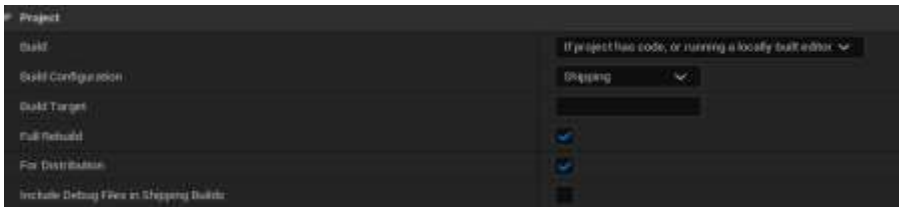

Рисунок 3.91 – Налаштування компіляції гри

Компілюємо і зберігаємо проект (рис 3.92).

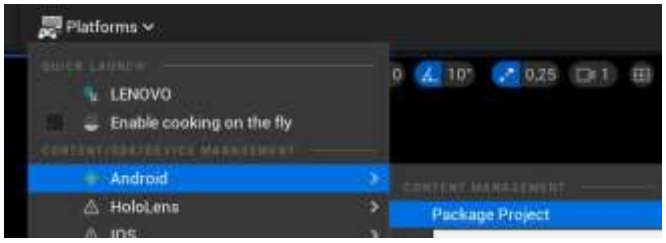

Рисунок 3.92 – Компіляція та проекту

3.5 Висновок до розділу

У цьому розділі було створено графічний та звуковий контент. А саме були розроблені 3D-моделі головного героя та ворога осцилографа, намальовані 2D об’єкти оточення та інтерфейсу меню. Було створено два музичні трека. Створений контент був додан до ігрового рушію. Таким чином було створено ритм гру на рушії Unreal Engine 5.

ВИСНОВКИ

У кваліфікаційній роботі була проведена розробка ігрового додатку "Ритм" на базі ігрового рушія Unreal Engine 5. Обране завдання вимагало вивчення сучасних технологій створенні ігор, основ функціонування ігрових рушіїв та їхніх особливостей. А також знання програм для роботи з 2D та 3D графікою і обробкою звука.

Для досягнення поставленої мети було проведено аналіз можливостей Unreal Engine 5, визначено його переваги та недоліки, і в результаті вибрано оптимальні рішення для розгри. Процес розробки включав в себе створення моделей, інтерфейсу користувача, програмування ігрової логіки та інтеграцію музичних елементів у гру.

В результаті роботи був отриманий функціональний прототип ігрового додатку "Ритм", який успішно демонструє взаємодію гравця з музичним середовищем. Використання Unreal Engine 5 надало можливість створити приємні візуальні та звукові ефекти, що підвищують поринання у гру.

Отже, робота дозволила реалізувати поставлену мету та продемонструвати потенціал ігрового рушія Unreal Engine 5 у створенні захоплюючих ігор. Дана розробка може послужити основою для подальших досліджень у галузі ігрової розробки та розширення можливостей розробки ігор на основі новітніх технологій.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. PaRappa the Rapper [Електронний ресурс].
   URL https://en.wikipedia.org/wiki/PaRappa_the_Rapper(дата звернення 30.10.2023)
2. Guitar Hero [Електронний ресурс].
   URL: https://en.wikipedia.org/wiki/Guitar_Hero(дата звернення 30.10.2023)
3. Osu! [Електронний ресурс].
   URL: https://osu.ppy.sh/wiki/uk/People/osu%21_team (дата звернення 30.10.2023)
4. Osu! Tournaments [Електронний ресурс].
   URL: https://en.wikipedia.org/wiki/Osu!#Tournaments (дата звернення 30.10.2023)
5. Rhythm Heaven [Електронний ресурс].
   URL: https://en.wikipedia.org/wiki/Rhythm_Heaven (дата звернення 30.10.2023)
6. Muse Dash [Електронний ресурс].
   URL: https://en.wikipedia.org/wiki/Muse_Dash (дата звернення 30.10.2023)
7. Beat Saber [Електронний ресурс].
   URL: https://beatsaber.com/ (дата звернення 30.10.2023)
8. Friday night funkin [Електронний ресурс].
   URL: https://en.wikipedia.org/wiki/Friday_Night_Funkin%27 (дата звернення 30.10.2023)
9. Hatsune Miku: Colorful Stage! [Електронний ресурс].
   URL: https://en.wikipedia.org/wiki/Hatsune_Miku:_Colorful_Stage! (дата звернення 30.10.2023)
10. Cytus [Електронний ресурс].
    URL: https://en.wikipedia.org/wiki/Cytus (дата звернення 30.10.2023)

11. BPM: Bullets Per Minute [Електронний ресурс].
URL: https://en.wikipedia.org/wiki/BPM:_Bullets_Per_Minute (дата звернення 30.10.2023)
12.Rhythm Doctor [Електронний ресурс].
https://rhythmdr.com/ (дата звернення 30.10.2023)
13.Unity [Електронний ресурс].
URL: https://unity.com/products/unity-engine(дата звернення 03.11.2023)
14.Blueprint [Електронний ресурс].
URL: https://en.wikipedia.org/wiki/Unreal_Engine#UnrealScript(дата звернення 05.11.2023)
15.Unreal Engine [Електронний ресурс].
URL: https://www.unrealengine.com/en-US/unreal-engine-5 (дата звернення 03.11.2023)
16. GameMaker [Електронний ресурс].
URL: https://en.wikipedia.org/wiki/GameMaker (дата звернення 03.11.2023)
17. Blender [Електронний ресурс].
https://www.blender.org/ (дата звернення 03.11.2023)
18. AutodeskMaya [Електронний ресурс].
https://www-autodesk-com.translate.goog/products/maya/overview?_x_tr_sl=en&_x_tr_tl=uk&_x_tr_hl=uk&_x_tr_pto=sc&term=1&tab=subscription&plc=MAYA (дата звернення 03.11.2023)
19. Adobe Photoshop [Електронний ресурс].
https://www.adobe.com/ua/ (дата звернення 07.11.2023)
20. Krita [Електронний ресурс].
https://krita.org/en/ (дата звернення 07.11.2023)
21.FL Studio [Електронний ресурс].
https://www.image-line.com/ (дата звернення 07.11.2023)
22. Ableton Live [Електронний ресурс].
https://www.ableton.com/ (дата звернення 07.11.2023)

23. V. M. Kartashov, V. N. Oleynikov, S. A. Sheyko, I. V. Koryttsev, S. I. Babkin, O. V. Zubkov, "Peculiarities of small unmanned aerial vehicles detection and recognition," Telecommunications and Radio Engineering, 2019, V. 78, Iss. 9, pp. 771–781. DOI: 10.1615/TelecomRadEng.v78.i9.30.
24. V. N. Oleynikov, O. V. Zubkov, V. M. Kartashov, I. V. Korytsev, S. I. Babkin, S.A. Sheiko, "Investigation of detection and recognition efficiency of small unmanned aerial vehicles on their acoustic radiation," Telecommunications and Radio Engineering, 2019, V. 78, Iss. 9, pp. 759–770. DOI: 10.1615/TelecomRadEng.v78.i9.20.
25. Kartashov, V.M., Sidorov, G.I., Sheiko, S.A., Kolendovska, M.M., Sergienko O.Yu. Principles of Construction and Assessment of technical Characteristics of multi-Frequency atmospheric Sodar in the Humidity Measurement Mode / Telecommunications and Radio Engineering.- New York. - 2020.- Vol. 79, №4.- P.323-333. (стаття). DOI: 10.1615/TelecomRadEng.v79.i4.50.
26. Kartashov, V.M., Oleynikov V.N, Zubkov, O.V., Korytsev I.V., Babkin, S. I., Sheiko, S.A.,. Kolendovskaya, M.M. Spatial-temporal Processing of acoustic Signals of Unmanned Aerial Vehicles; Telecommunications and Radio Engineering, 2020. Vol. 79, Iss, 9, pp.769-780.
27. V.M. Semenets, V.M. Kartashov, V.I. Leonidov. Features of Acoustic Noise of Small Unmanned Aerial Vehicles // Telecommunications and Radio Engineering.- New York. - 2020.- Vol. 79, №11.- P. 985-995. DOI: 10.1615/TelecomRadEng.v79.i11.80 (стаття).
28. Oleynikov V.N., Kartashov, V.M., Babkin, S. I., Zubkov, O.V., Korytsev I.V., Sheiko, S.A.,. Seleznov I.S. Structure and Parameter Unmanned Aerial Vehicles Sound Fields/ Telecommunications and Radio Engineering.- New York. - 2020.- Vol. 79, №17.- P.1539-1550. DOI: 10.1615/TelecomRadEng.v79.i17.50 (стаття).
29. В.А. Тихонов, В.М. Карташов, В.М. Олейніков, В.І. Леонідів, Л.П. Тимошенко, І.С. Селезнєов, Н.В. Рибніков. Виявлення-розпізнавання безпілотних літальних апаратів з використанням складової моделі

авторегресії їхнього акустичного випромінювання// Вісник НТУУ «КПІ». Радіотехніка. Радіоапаратобудування. – 2020. – Віп. №81. - С.38-46. (Web of Science) Карташов В.М., Коритцев І.В., Олійников В.М., Зубков О.В., Шейко С.А., Бабкін С.І., Левський Н.А., Селезньов І.С. Алгоритми пеленгації безпілотних літальних апаратів з їхнього акустичного випромінювання// Радіотехніка. (Харків). - 2019. - Вип. 196. - С. 22-31.Карташов В.М., Харченко О.І., Чумаков В.І. Використання ефекту стохастичного резонансу аналізу спектрів акустичного випромінювання малих безпілотних літальних апаратів // Радіотехніка. (Харьков). — 2019. — Вып. 197. — С. 100-106.

30. Карташов В.М., Сидоров Г.І., Толстих Є.Г., Шаповалов С.В. Акустичний вимірювач швидкості вітру в атмосферному прикордонному шарі// Радиотехника. (Харьков). — 2019. — Вып. 199. – С. 54-58.
31. A Comparative Example Between The Use Of Pca And Mds For Image Classification / Hernandez, W., Mendez, A., Flor-Unda, O., Camejo, I.M., Kolendovska, M.// IEEE International Symposium on Industrial Electronics, 29th IEEE International Symposium on Industrial Electronics, ISIE 2020; Delft; Netherlands; 17 June 2020 до 19 June 2020; Volume 2020-June, June 2020, № 9152565, Pages 1353-1358
32. Algorithm For Generating Refined Frequency Estimates In Atmospheric Radio Sounding Systems / Kartashov V., Hernandez W., Hernandez-Balbuena D., M. Kolendovska, Konovalenko O., Melnyk V.// IEEE International Symposium on Industrial Electronics, 29th IEEE International Symposium on Industrial Electronics, ISIE 2020; Delft; Netherlands; 17 June 2020 до 19 June 2020; Volume 2020-June, June 2020, № 9152562, Pages 79-82
33. Application of Fast Frequency Shift Measurement Method for INS in Navigation of Drones / D. Avalos-Gonzalez, D.H. Balbuena, V. Tyrsa, V.M. Kartashov, M. Kolendovska, S. Sheiko, O. Sergiyenko, V. Melnyk, F.N. Murrieta-Rico // IECON 2018 – 44th Annual Conference of the IEEE Industrial Electronics Society. – P. 3159–3164.

34. Avalos-Gonzalez, D., Sergiyenko, O., Hernandez-Balbuena, D., Tyrsa,V., Kartashov V.M., V.,Rivas-Lopes, M., Murrieta-Rico, F.N. Constraints definition and application optimization based on geometric analysis of the frequency measurement method by pulse coincidence// Measurement: Journal of the International Measurement Confederation (USA). 2018, V.126. P. 184-193.

35. Book “Control and Signal Processing Applications for Mobile and Aerial Robotic Systems”, Hardback - Advances in Computational Intelligence and Robotics English. Edited by Oleg Sergiyenko, Moises Rivas-Lopez, Wendy Flores-Fuentes, Julio Cesar Rodríguez-Quiñonez, Lars Lindner. Editorial IGI Global, Hershey, United States, January 2020, 340 páginas. ISBN10 152259924X, ISBN13 9781522599241

36. Cesar Sepulveda-Valdez ; Oleg Sergiyenko ; Vera Tyrsa ; Wendy Flores-Fuentes ; Julio César Rodríguez-Quiñonez ; Fabian Natanael Murrienta-Rico ; Jesús Elías Miranda-Vega ; Paolo Mercorelli ; Marina Kolendovska. "Geometric analysis of a laser scanner functioning based on dynamic triangulation," 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, Netherlands, 17-19 of June 2020, pp. 1398-1403, doi: 10.1109/ISIE45063.2020.9152268.

https://ieeexplore.ieee.org/abstract/document/9152268

37. Cuauhtémoc Mariscal-García; Wendy Flores-Fuentes; Daniel Hernández-Balbuena; Julio C. Rodríguez-Quiñonez ; Oleg Sergiyenko. "Classification of Vehicle Images through Deep Neural Networks for Camera View Position Selection," 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, Netherlands, 17-19 of June 2020, pp. 1376-1380, doi: 10.1109/ISIE45063.2020.9152440.

https://ieeexplore.ieee.org/abstract/document/9152440

38. Developing and Applying Optoelectronics in Machine Vision/ O. Sergiyenko, J.C. Rodriguez-Quiñonez, IGI Global, 2016; 341p.

39. Experimental estimation of direction finding to unmanned air vehicles algorithms efficiency by their acoustic emission, /Oleynikov, V., Zubkov, O., Kartashov, V., ...Sheiko, S., Babkin, S.//2019 IEEE International Scientific-Practical Conference: Problems of Infocommunications Science and Technology, PIC S and T 2019 - Proceedings, 2019, стр. 175-178, 9061337
40. Features of acoustic noise of small unmanned aerial vehicles / Semenets, V.V., Kartashov, V.M., Leonidov, V.I. //Telecommunications and Radio Engineering (English translation of Elektrosvyaz and Radiotekhnika), 2020, 79(11), стр. 985-995
41. Geometric Analysis Of A Laser Scanner Functioning Based On Dynamic Triangulation /Sepulveda-Valdez, C., Sergiyenko, O., Tyrsa, V, Mercorelli, P., Kolendovska, M.// IEEE International Symposium on Industrial Electronics, 29th IEEE International Symposium on Industrial Electronics, ISIE 2020; Delft; Netherlands; 17 June 2020 до 19 June 2020; Volume 2020-June, June 2020, № 9152268, Pages 1398-1403
42. I. Y. A. Corpus, L.Lindner, O.Sergiyenko. "Transimpedance Amplifier for Laser Scanning System Range Extension," 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, Netherlands, 17-19 of June 2020, pp. 1421-1426, doi: 10.1109/ISIE45063.2020.9152487. https://ieeexplore.ieee.org/abstract/document/9152487
43. Ivanov, M., Sergiyenko, O., Mercorelli, P., Hernandez, W.c, Rodriguez Quinonez, J.C.d, Katashov V., Kolendovska, M., Iryna, T. Effective informational entropy reduction in multi-robot systems based on real-time TVS. IEEE International Symposium on Industrial Electronics, 2019-June,8781209, c. 1162-1167.
44. Jonathan J. Sanchez-Castro ; Julio C. Rodríguez-Quiñonez ; Luis R. Ramírez-Hernández ; Guillermo Galaviz ; Daniel Hernández-Balbuena ; Gabriel Trujillo-Hernández ; Wendy Flores-Fuentes ; Paolo Mercorelli ; Wilmar Hernández-Perdomo ; Oleg Sergiyenko ; Félix Fernando González-Navarro. "A Lean Convolutional Neural Network for Vehicle Classification," 2020 IEEE 29th

International Symposium on Industrial Electronics (ISIE), Delft, Netherlands, 17-19 of June 2020, pp. 1365-1369, doi: 10.1109/ISIE45063.2020.9152274. https://ieeexplore.ieee.org/abstract/document/9152274

45. Lindner, L., Sergiyenko, O., Rivas-López, M., (...), Gurko, A., Kartashov, V.M. Machine vision system for UAV navigation; IEEE, 2016 International Conference on Electrical Systems for Aircraft, Railway, Ship Propulsion and Road Vehicles and International Transportation Electrification Conference, ESARS-ITEC, 2016; pp.1–6. DOI: 10.1109/ESARS-ITEC.2016.7841356.
46. M. Ivanov, O. Sergiyenko, V. Tyrsa, P. Mercorelli, V. Kartashov, W. Hernandez, S. Sheiko, M. Kolendovska. Individual scans fusion in virtual knowledge base for navigation of mobile robotic group with 3D TVS // Proceedings of 44th Annual Conference of IEEE Industrial Electronics Society (IECON).. -2018. – Washington DC, USA. -S. 3187-3192. . ISBN 978-1-5090-6683-4/18/.
47. Murrieta-Rico, F.N., Petranovskii, V., Galvan, D.H., Sergiyenko, O., Yocupicio-Gaxiola, R.I., De Dios Sanchez-Lopez, J. Phase effect in frequency measurements of a quartz crystal using the pulse coincidence principle. 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, Netherlands, 17-19 of June 2020, pp. 185-190, 9152255, DOI: 10.1109/ISIE45063.2020.9152255
48. Oleksandr Sotnikov, Vladimir Kartashov, Oleksandr Tymochko, Oleg Sergiyenko, Vera Tyrsa, Paolo Mercorelli, Wendy Flores-Fuentes. Methods for Ensuring the Accuracy of Radiometric and Optoelectronic Navigation Systems of Flying Robots in a Developed Infrastructure. Chapter 16// Machine Vision and Navigation; Springer, Cham. pp.537–578. Editors: Sergiyenko, Oleg, Flores-Fuentes, Wendy, Mercorelli, Paolo. DOI: 10.1007/978-3-030-22587-2_16.
49. Optical detection of unmanned air vehicles on a video stream in a real-time/Kartashov, V., Oleynikov, V., Zubkov, O., Sheiko, S.// 2019 International

Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019 - Proceedings, 2019, 9165362/

50. Principles Of Construction And Assessment Of Technical Characteristics Of Multi-Frequency Atmospheric Sodar In The Humidity Measurement Mode / Kartashov, V.M., Sidorov, G.I., Sheiko, S.A., Kolendovskaya, M.M., Sergienko, O.Yu. // Telecommunications And Radio Engineering (English Translation Of Elektrosvyaz And Radiotekhnika), 2020, ISSN Print: 0040-2508, ISSN Online: 1943-6009, DOI: 10.1615/TelecomRadEng.v79.i4.50, p. 323-333/

51. Research Of The Uncertainty Of Measurement Frequencies And Definitions Of The Frequency Signal In The Waveguide With Respect To Power / Semenets, V.Zakharov, I. Serhiienko, M., Kartashov, V.M,, Kolendovska, M., Hernandez, W., Hipolito, J.I.N.,, Tyrsa, V.// 45th Annual Conference of the IEEE Industrial Electronics Society, IECON 2019; Lisbon Congress CenterLisbon; Portugal; 14 October 2019 до 17 October 2019; CFP19IEC-ART; Код 155980, Volume 2019-October, October 2019, № 8927203, Pages 4674-4679

52. Spatial-Temporal Processing Of Acoustic Signals Of Unmanned Aerial Vehicles /Kartashov V.M., Oleinikov V.N., Zubkov O.V., Sheiko S.A., Kolendovska M.M.// Telecommunications And Radio Engineering (English Translation Of Elektrosvyaz And Radiotekhnika), 2020, ISSN Print: 0040-2508, ISSN Online: 1943-6009, DOI: 10.1615/Telecomradeng.v79.i9.40, p. 769-780

53. Stereoscopic Vision Systems In Machine Vision, Models, And Applications (Book Chapter)/ Ramírez-Hernández, L.R., Rodríguez-Quiñonez, J.C., Castro-Toscano, M.J., Kolendovska, M., Murrieta-Rico, F.N.// Machine Vision And Navigation, 2019 Machine Vision and Navigation30 September 2019, Pages 241-265

54. StrelkovaT., KartashovV., Lytyuga A., Strelkov A. Theoretical Methods of Images Processing in Optoelectronic Systems. Chapter 16. // Biometrics: Concepts, Methodologies, Tools, and Applications; Oleg Sergiyenko and Julio

C. Rodriguez-Quiñonez. (341p.), IGI Global, 2017; pp. 361-381. DOI: 10.4018/978-1-5225-0983-7.ch016.

55. StrelkovaT., KartashovV., Lytyuga A.,StrelkovA. Theoretical Methods of Images Processing in Optoelectronic Systems. Chapter 6// Developing and Applying Optoelectronics in Machine Vision; Oleg Sergiyenko and Julio C. Rodriguez-Quiñonez. (341p.) – USA, Herhey, IGI Global, 2016; pp.180-205.
56. Sytnik O., KartashovV. Methods and Algorithms for Technical Visionin Radar Introscopy. Chapter 13// Optoelectronics in Machine Vision-Based Theories and Applications. IGI Global, 2019; pp. 373-391.
57. The Use of Factorization and Multimode Parametric Spectra in Estimating Frequency and Spectral Parameters of Signal/Semenets, V., Kartashov, V., Sergiyenko, O., ...Rodriguez-Quinonez, J.C., Flores-Fuentes, W.//IEEE International Symposium on Industrial Electronics, 2020, 2020-June, p. 215-219
58. Unda, O.F., Hernandez, W., Vargas, O., Mendez, A., Sergiyenko, O., Tyrsa, V. Construction of a robotic platform of differential type for first-year students of electronic engineering, 2020 International Symposium on Power Electronics, Electrical Drives, Automation and Motion, SPEEDAM 2020, 24-26 de junio de 2020, Sorrento, Italia, pp. 538-543, 9161870, DOI: 10.1109/SPEEDAM48782.2020.9161870
59. Use of Acoustic Signature for Detection, Recognition and Direction Finding of Small Unmanned Aerial Vehicles/Kartashov, V., Oleynikov, V., Koryttsev, I., ...Babkin, S., Selieznov, I.//Proceedings - 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering, TCSET 2020, 2020, p. 377-380/
60. V. Semenets; Vladimir Kartashov ; Oleg Sergiyenko; Vyacheslav Tikhonov ; Paolo Mercorelli ; Sergiy Sheiko ; Nataliya Chmelarova. "The Use of Factorization and Multimode Parametric Spectra in Estimating Frequency and Spectral Parameters of Signal," 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, Netherlands, 17-19 of June 2020, pp. 215-

219, doi: 10.1109/ISIE45063.2020.9152238.
https://ieeexplore.ieee.org/abstract/document/9152238

61. Wilmar Hernandez ; Alfredo Mendez ; Omar Flor-Unda ; Vicente Gonzalez-Posada ; Jose Luis Jimenez ; Oleg Sergiyenko ; Julio C. Rodriguez-Quiñonez ; Mykhailo Ivanov ; Ivan Menes Camejo ; Marina Kolendovska. "A comparative example between the use of PCA and MDS for image classification," 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, Netherlands, 17-19 of June 2020, pp. 1353-1358, doi: 10.1109/ISIE45063.2020.9152565.
https://ieeexplore.ieee.org/abstract/document/9152565