Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Програмної Інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

рівень вищої освіти перший (бакалаврський)
Ігровий програмний застосунок у жанрі 3D Rogue-like шутер від третьої особи.

Левел дизайн, UI/UX, балансування зброї, генерація ігрових об'єктів, наратив

(тема)

Виконав
студент 4 курсу, групи ПЗПІ-20-7
Борисенко А.Е.
(прізвище, ініціали)
Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Програмна інженерія

(повна назва освітньої програми)

Керівник старший викладач Новіков Ю.С.
(посада, прізвище, ініціали)

Допускається до захисту
Зав кафедри _____ Дудар З.В.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерних наук
Кафедра програмної інженерії
Рівень вищої освіти перший (бакалаврський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми Освітньо-професійна
Освітня програма Програмна Інженерія
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Студентові Борисенко Артемій Едуардович
(прізвище, ім’я, по батькові)

1. Тема роботи Ігровий програмний застосунок у жанрі 3D Rogue-like шутер від третьої особи. Левел-дизайн, UI/UX, балансування зброї, генерація ігрових об'єктів, наратив.

Затверджена наказом по університету від 20 травня 2024 р. № 471Ст

2. Термін подання студентом роботи до екзаменаційної комісії 06.06.2024

3. Вихідні дані до роботи Розробити ігровий програмний застосунок в жанрі 3D Rogue-like шутер від третьої особи, а саме левел-дизайн, UI/UX, балансування зброї, генерація ігрових об'єктів, наратив, за допомогою ігрового рушію Unreal Engine 5 та мови програмування Blueprints.

4. Перелік питань, що потрібно опрацювати в роботі

Вступ, аналіз предметної галузі, формування вимог до програмної системи, архітектура та проектування програмного забезпечення, опис прийнятих програмних рішень, тестування розробленого програмного забезпечення, висновки, додатки.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	08.04.2024	*виконано*
2	Створення специфікації ПЗ	11.04.2024	*виконано*
3	Проектування ПЗ	15.04.202	*виконано*
4	Розробка ПЗ	15.05.2024	*виконано*
5	Тестування ПЗ	17.05.2024	*виконано*
6	Оформлення пояснювальної записки	20.05.2024	*виконано*
7	Підготовка презентації та доповіді	21.05.2024	*виконано*
8	Попередній захист	22.05.2024	*виконано*
9	Нормоконтроль, рецензування	25.05.2024	*виконано*
10	Здача роботи у електронний архів	01.06.202	*виконано*
11	Допуск до захисту у зав. кафедри	03.06.2024	*виконано*

Дата видачі завдання 8 квітня 2024 р.

Студент (ка)_____ Борисенко А.Е.
(підпис)

Керівник роботи _____ старший викладач Новіков Ю.С.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка до кваліфікаційної роботи бакалавра, стор.105, рис.96, джерел.14, табл. 5.

ШУТЕР, ІГРОВИЙ ПРОЕКТ, 3D, ROGUE-LIKE, UNREAL ENGINE 5, ГРА ВІД ТРЕТЬОЇ ОСОБИ, КІБЕРПАНК

Об’єктом розробки - Ігровий програмний застосунок у жанрі 3D Rogue-like шутер від третьої особи в стилі кіберпанк, з фокусом на левел-дизайн, UI/UX, балансування зброї, генерацію ігрових об'єктів та труб, наратив.

Метою розробки - створення ігрового програмного застосунку з цікавою та різноманітною бойовою системою шутера, а також увагою до левел-дизайну, UI/UX, балансування зброї, генерації ігрових об'єктів та труб, наративу та освітлення. Гра повинна бути виконана у стилі кіберпанк.

Методом рішення - використання середовища розробки та ігрового рушія Unreal Engine 5, а також мови програмування Blueprint.

В результаті розробки був створений ігровий програмний застосунок, який включає в себе левел-дизайн, UI/UX, балансування зброї, генерацію ігрових об'єктів та труб, наратив, створення освітлення та збереження налаштувань гравця.

SHOOTER, GAME PROJECT, 3D, ROGUE-LIKE, UNREAL ENGINE 5, THIRD-PERSON GAMEPLAY, CYBERPUNK

The o bject of development is a game software application in the 3D Rogue-like shooter genre with third-person gameplay in a cyberpunk style, focusing on level design, UI/UX, weapon balancing, generation of game objects and pipes, and narrative.

The development goal is to create a game software application with an engaging and diverse shooter combat system, as well as attention to level design, UI/UX, weapon balancing, generation of game objects and pipes, narrative, and lighting. The game should be executed in a cyberpunk style.

The solution method is to use the development environment and game engine Unreal Engine 5, as well as the Blueprint programming language.

As a result of the development, a game software application was created, which includes level design, UI/UX, weapon balancing, generation of game objects and pipes, narrative, lighting creation, and player settings saving.

Я, Борисенко Артемій Едуардович, студент гр. ПЗПІ-20-7, здобувач вищої освіти на першому (бакалаврському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Ігровий програмний застосунок у жанрі 3D Rogue-like шутер від третьої особи», що буде представлена до екзаменаційної комісії для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElAr KhNURE. Усі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови до допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ВСТУП

У сучасному світі ігрова індустрія постійно розвивається, надаючи гравцям все більш складний та захоплюючий ігрові досвід. Одним із напрямів цього розвитку є кіберпанк, жанр, що поєднує в собі футуристичні технології, соціальні проблеми та атмосферу кібернетичного міста.

Rogue-like — це піджанр рольових ігор, у яких кожне рішення має вирішальне значення [1], вони мають дуже великий попит. Такі ігри характеризуються процедурно згенерованими світами, незворотною смертю та випадковими предметами, створюючи захопливий та повторюваний досвід.

Рухлива природа випадкових подій, незворотність смерті та постійно мінливі предмети роблять кожне проходження унікальним, розпалюючи азарт та мотивацію гравців. Це змушує їх постійно адаптуватися, що й становить головну ціль жанру Rogue-like.

У рамках даного дипломного проекту була розроблена гра в стилі кіберпанк 3D Rogue-like шутер що поєднує в собі елементи кіберпанку та характерні риси Rogue-like ігор. Шутер від третьої особи — це жанр тривимірних відеоігор, в якому камера знаходиться не в очах персонажа [2]. Гра створена на платформі Unreal Engine 5, що дозволяє досягти високої якості графіки та реалістичності геймплею.

Метою цього проекту є створення захоплюючої та непередбачуваної гри, що спонукає гравців до постійного вдосконалення своїх навичок та стратегій. Гра пропонує гравцям унікальний ігровий досвід кіберпанк у всесвіті, що постійно змінюється та адаптується до їхніх дій.

У цьому дипломному проекті буде розглянуто процес розробки гри, від постановки завдання до реалізації ігрового процесу та тестування. Також будуть розглянуті основні принципи гри в стилі Rogue-like та способи їх втілення у кіберпанк атмосфері.

1 АНАЛІЗ ПРЕДМЕТНОЇГАЛУЗІ

1.1 Аналіз предметної галузі

Жанр Rogue-like характеризується процедурно генерованими рівнями, перманентною смертю персонажа і великою випадковістю. Гравець повертається на початок гри при смерті, без можливості продовження зі збереженими досягненнями.

"Enter the Gungeon" — roguelike шутер, в якому гравці б'ються в підземеллі з метою дістатися до самої потужної у світі зброї [3]. Кожен рівень у грі створюється випадковим чином, що робить кожне проходження унікальним. Гравець має збирати різні предмети та зброю, кожен з яких має власні унікальні властивості та впливає на геймплей. Перманентна смерть означає, що після смерті персонажа гравець повертається на початок, втрачаючи всі досягнення.

"DoomRL" — відеогра типу roguelike, розроблена ChaosForge на основі шутерів від першої особи Doom і Doom II [4]. У грі висока випадковість, кожне нове проходження створює унікальні умови. Гравець може покращувати характеристики персонажа та використовувати різні тактичні прийоми для перемоги над монстрами.

"Cyberpunk 2077" — пригодницький рольовий екшн у відкритому світі мегаполісу Найт-Сіті, де у ролі кіберпанкового найманця ви боротиметеся за виживання. [5]. Гра використовує багато ключових елементів кіберпанку, таких як антиутопічне суспільство, низький рівень життя, технологічний прогрес, хакерська субкультура та трансгуманізм. Ці елементи поєднуються, щоб створити атмосферний та захоплюючий досвід для гравців.

1.2 Аналіз конкурентів

Гра «Enter the Gungeon» (див. рис. 1.1).:

Переваги:

– Геймплей: Динамічний геймплей зі стрілянином, використанням

вогнепальної зброї та уникати пасток.

– Багато рівнів: Велика кількість різноманітних рівнів та босів, що робить гру цікавою та непередбачуваною.
– Спільна гра: Можливість грати в кооперативному режимі з друзями, що додає соціальний елемент.

Мінуси:

– Складність: Для деяких гравців гра може бути дуже складною та вимагати багато спроб.
– Монотонність: Після деякого часу гра може стати монотонною через однотипність геймплею.

Рисунок 1.1 – Скріншот з гри Enter the Gungeon

Гра «DoomRL» (див. рис. 1.2):

Переваги:

– Динаміка: Швидкий та динамічний геймплей, що дозволяє швидко отримувати задоволення від гри.
– Відтворюваність: Велика кількість різних персонажів, зброї та рівнів робить кожну гру унікальною.

Мінуси:

– Графіка: Для деяких гравців може виглядати застарілою або нецікавою через свою піксельну графіку.
– Складність: Гра може бути дуже важкою навіть для досвідчених гравців, що може відлякати новачків.

Рисунок 1.2 – Скріншот з гри DoomRL

Гра «Cyberpunk 2077» (див. рис. 1.3):

Рисунок 1.3 – Скріншот з гри Cyberpunk 2077

Переваги:

– Неймовірний світ: Night City вражає деталізацією, неоновим стилем та

глибиною опрацювання. Це місто, де технології тісно переплетені з соціальними проблемами, що створює неповторну атмосферу.

– Графіка нового покоління: Cyberpunk 2077 виглядає приголомшливо. Візуальне оформлення переносить вас у майбутнє, де людина та машина тісно взаємодіють.
– Багатий сюжет з вибором: Рішення, які ви приймаєте, впливають на історію. Це робить кожне проходження гри унікальним.
– Різноманітний бойовий стиль: Вибирайте зброю та тактику, які підходять вашому стилю гри, та знищуйте ворогів на власних умовах.

Недоліки:

– Технічні проблеми: Особливо на старих платформах гра може страждати від падінь продуктивності та інших технічних нюансів.
– Лінійність локацій: Деякі сюжетні локації пропонують обмежену свободу дослідження, оскільки їх проходження часто йде за чітким маршрутом.
– Відсутність TPS елементів: Хоча гра пропонує вид від першої особи, елементів шутера від третьої особи майже немає.

1.3 Виявлення та вирішення проблем

У жанрі Rogue-like ключові аспекти, які потребують уваги розробників, включають баланс гри, випадковість, перманентну смерть та управління ресурсами. Для забезпечення цікавого геймплею важливо збалансувати складність ігрового процесу, зробити випадковість передбачуваною, додати можливості збереження прогресу при смерті та ретельно виважити використання ресурсів гравцем.

1.4 Постановка задачі

У процесі створення гри в жанрі rogue-like, зокрема щодо таких складових як

генерація ігрових об'єктів, левел-дизайн, UI/UX, збереження балансування зброї та наратив, необхідно врахувати наступні аспекти:

а) генерація ігрових об'єктів;

1) створити генерацію труб;
2) створити генерацію предметів

б) левел-дизайн;

1) створення макета стартової локації;
2) створення макета локації зі спавном зброї;
3) створення макета локації боса;
4) створення макета локації №4;
5) створення макета локації №5;
6) створення макета локації №6;
7) створення макета локації №7;
8) створення макета локації №8;
9) створення стартової локації;
10) створення локації зі спавном зброї;
11) створення локації боса;
12) створення локації №4;
13) створення локації №5;
14) створення локації №6;
15) створення локації №7;
16) створення локації №8;
17) створення локації «магазин»
18) створення освітлення на локаціях спавна зброї, стартової локації, локації боса, локації магазин №4-8

в) UI/UX;

1) потрібно розробити інтерфейс головної сторінки;
2) потрібно розробити інтерфейс сторінки обрання персонажів;

3) потрібно розробити інтерфейс сторінки налаштувань відео;
4) потрібно розробити інтерфейс сторінки налаштувань аудіо;
5) потрібно розробити інтерфейс сторінки налаштувань зміни клавіш;
6) потрібно розробити інтерфейс сторінки налаштувань локалізації;
7) потрібно розробити інтерфейс сторінки паузи;
8) потрібно розробити інтерфейс смерті;
9) потрібно розробити інтерфейс сторінки перемоги;
10) потрібно розробити інтерфейс підтвердження виходу з гри;
11) розробити логіку переходу між сторінками головного меню;
12) розробити логіку запуску гри;
13) розробити логіку налаштувань відео, аудіо та зміну клавіш;
14) розробити логіку придбання персонажів;
15) розробити логіку екранів перемоги та смерті;
16) розробити логіку розрахунку коштів;
17) розробити логіку обрання та зміни персонажів;
18) розробити логіку виведення інформації про персонажа, зброю та предмети;
19) створити інтерфейс для відображення предметів та опису їх;
20) створити логіку підтвердження виходу з гри;

г) балансування зброї;
1) збалансувати всі види зброї;
2) збалансувати ШІ;

д) наратив;
1) створити відео з українськими субтитрами;
2) створити відео з англійськими субтитрами;
3) розробити логіку для запуску кат-сцени;

е) збереження:
1) створити збереження параметрів налаштувань аудіо;

2) створити збереження параметрів налаштувань відео;
3) створити збереження параметрів налаштувань локалізації;
4) створити збереження параметрів налаштувань змін клавіш;

Під час розробки та створення цього геймпроекту я буду займатися саме цими завданнями.

1.4.1 Цільова аудиторія

Rogue-like ігри спрямовані на геймерів, які шукають відчуття досягнення через високий рівень виклику та випадковість в геймплеї. Ці ігри також привертають любителів стратегічного мислення, оскільки вони часто вимагають розсудливих рішень та ефективного управління ресурсами.

Перша гра у жанрі Rogue-like, Beneath Apple Manor - це гра, схожа на жуліків, написана Доном Вортом для Apple II і опублікована The Software Factory у 1978 році.[2] (див. рис. 1.4).

Рисунок 1.4 – Скріншот з першої гри у жанрі Rogue-like «Beneath Apple Manor»

«Спеціальні випуски» з вищою роздільною здатністю були випущені в 1982 і 1983 роках через програмне забезпечення якості для 8-розрядних комп’ютерів Apple

II і Atari Це була одна з перших відеоігор, яка використовувала процедурну генерацію[6].

1.4.2 Монетизація

Ми збираємося пропонувати гравцям можливість придбати наш ігровий застосунок за фіксованою ціною, без необхідності здійснювати додаткові покупки усередині гри. Це означає, що гравці можуть спокійно насолоджуватися ігровим процесом, не думаючи про фінансові витрати. Такий підхід підвищує довіру гравців до нашої команди розробників і створює більш позитивне враження від ігрового досвіду.

2 ФОРМУВАННЯ ВИМОГ ДО ПРОГРАМНОЇ СИСТЕМИ

2.1 Функціональні вимоги до ігрового застосунку

Згідно з вимогами, викладеними у попередньому розділі, а також ураховуючи переваги та недоліки розглянутих аналогів, для розроблюваного компоненту ігрового застосунку встановлюються наступні функціональні вимоги:

а) генерація ігрових об'єктів;

1) згенеровані труби повинні мати власну колізію, мати логіне розташування та підходити під стилістику локації;
2) всі згенеровані предмети повинні мати власну колізію, мати логіне розташування та розмір, підходити під стилістику локації;

б) левел-дизайн;

1) задля створення більш якісного планування стартової локації буде створено макет з акторів куб;
2) задля створення більш якісного планування локації зі спавном зброї буде створено макет з акторів куб;
3) задля створення більш якісного планування локації боса буде створено макет з акторів куб;
4) задля створення більш якісного планування локації №4 буде створено макет з акторів куб;
5) задля створення більш якісного планування локації №5 буде створено макет з акторів куб;
6) задля створення більш якісного планування локації №6 буде створено макет з акторів куб;
7) задля створення більш якісного планування локації №7 буде створено макет з акторів куб;
8) задля створення більш якісного планування локації №8 буде створено макет з акторів куб;

9) після розробки детального макету стартової локації буде створена повноцінна локація з стилістикою та детально розробленим положенням генерованих предметів та труб для різноманітних генерацій локацій;
10) після розробки детального макету буде створена повноцінна локація з стилістикою та детально розробленим положенням генерованих предметів та труб для різноманітних генерацій локацій;
11) після розробки детального макету локації боса буде створена повноцінна локація з стилістикою та детально розробленим положенням генерованих предметів та труб для різноманітних генерацій локацій;
12) після розробки детального макету локації №4 буде створена повноцінна локація з стилістикою та детально розробленим положенням генерованих предметів та труб для різноманітних генерацій локацій;
13) після розробки детального макету локації №5 буде створена повноцінна локація з стилістикою та детально розробленим положенням генерованих предметів та труб для різноманітних генерацій локацій;
14) після розробки детального макету локації №6 буде створена повноцінна локація з стилістикою та детально розробленим положенням генерованих предметів та труб для різноманітних генерацій локацій;
15) після розробки детального макету локації №7 буде створена повноцінна локація з стилістикою та детально розробленим положенням генерованих предметів та труб для різноманітних генерацій локацій;
16) після розробки детального макету локації №8 буде створена повноцінна локація з стилістикою та детально розробленим положенням генерованих предметів та труб для різноманітних генерацій локацій;
17) після розробки детального макету локації «магазин» буде створена повноцінна локація з стилістикою та детально розробленим положенням генерованих предметів та труб для різноманітних генерацій локацій;
18) крім детально опрацьованих локацій буде створена генерація освітлення

для зміни складності проходження та візуальної різноманітності локацій;

в) UI/UX;

1) для якісного та зручного розташування клавіш та вікон буде розроблено інтерфейс головної сторінки;
2) для якісного та зручного розташування клавіш та вікон буде розроблено інтерфейс сторінки обрання персонажів;
3) для якісного та зручного розташування клавіш та вікон буде розроблено інтерфейс сторінки налаштувань відео;
4) для якісного та зручного розташування клавіш та вікон буде розроблено інтерфейс сторінки налаштувань аудіо;
5) для якісного та зручного розташування клавіш та вікон буде розроблено інтерфейс сторінки налаштувань зміни клавіш;
6) для якісного та зручного розташування клавіш та вікон буде розроблено інтерфейс сторінки налаштувань локалізації;
7) для якісного та зручного розташування клавіш та вікон буде розроблено інтерфейс сторінки паузи;
8) для якісного та зручного розташування клавіш та вікон буде розроблено інтерфейс смерті;
9) для якісного та зручного розташування клавіш та вікон буде розроблено інтерфейс сторінки перемоги;
10) для якісного та зручного розташування клавіш та вікон буде розроблено інтерфейс підтвердження виходу з гри;
11) для швидкої та якісної обробки дій гравця в меню буде створена логіка переходу між сторінками головного меню;
12) для швидкої та якісної обробки дій гравця в меню буде створена логіка запуску гри;
13) для швидкої та якісної обробки дій гравця в меню буде створена логіка налаштувань відео, аудіо та зміну клавіш;

14) для швидкої та якісної обробки дій гравця в меню буде створена логіка придбання персонажів;
15) для швидкої та якісної обробки дій гравця в меню буде створена логіка екранів перемоги та смерті;
16) для швидкої та якісної обробки дій гравця в меню буде створена логіка розрахунку коштів;
17) для швидкої та якісної обробки дій гравця в меню буде створена логіка обрання та зміни персонажів;
18) для швидкої та якісної обробки дій гравця в меню буде створена логіка виведення інформації про персонажа, зброю та предмети;
19) для якісного та зручного розташування клавіш та вікон буде розроблено інтерфейс для відображення предметів та опису їх;
20) для швидкої та якісної обробки дій гравця в меню буде створена логіка підтвердження виходу з гри;

г) балансування зброї;
1) для зрівняння всіх видів зброї та їх характеристик слід заповнити таблицю;
2) для зрівняння очок життя та усіх видів зброї та їх характеристик слід заповнити таблицю;

д) наратив;
1) для якісної україномовною версії кат-сцени треба розробити відео із матеріалів у стилістиці 80-х та створення аудіо за допомогою ші;
2) для якісної англомовною версії кат-сцени треба розробити відео із матеріалів у стилістиці 80-х та створення аудіо за допомогою ші;
3) після розробки матеріалів для кат-сцени потрібно розробити логіку завантаження обраних матеріалів для обраної локалізації;

е) збереження:
1) створити логіку для збереження обраних характеристик налаштувань

аудіо обраних гравцем;

2) створити логіку для збереження обраних характеристик налаштувань відео обраних гравцем;
3) створити логіку для збереження обраних налаштувань локалізації обраних гравцем;
4) створити логіку для збереження обраних налаштувань змін клавіш обраних гравцем;

Під час розробки та створення цього геймпроекту я буду виконувати ці конкретні завдання.

2.2 Нефункціональні вимоги до ігрового застосунку

Цей ігровий програмний продукт має наступні вимоги:

- інтерфейс головного меню повинен мати якісну кольорову гаму яка не ускладнює вибір гравця та допомагає інтуїтивно розібратися з функціоналом;
- розташування розділів інтерфейсу не повинні буди в анархічному порядку;
- усі дії з грошима повинні відображені для гравця в окремій ділянці;
- завдяки підтвердженню бажання покинути гру, гравець не зможе випадково покинути сесію;
- для відмінності між відкритим та схованими розділами буде виділення іншим кольром;
- для відтворення стилістики кіберпанку треба буде обрати кольори які виглядають якісно та стильно і водночас не заважають гравцю та не створють дискомфорт;
- меню налаштування повинно бути інтуїтивно зрозуміло для гравця;
- локації не повинні мати перевантажену забудову для більш якісної оглядовості та комфортної гри;
- освітлення повинно допомагати гравцеві у розпізнаванні ворогів і забудови;

– усі локації повинні мати свою стилістику але мати схожість між собою;

Тому саме такі показники мають бути в нашому ігровому застосунку для підтримання стилістики та комфортної гри гравцям.

2.3 Вимоги до середовищ розробки

Під час розробки цього ігрового застосунку буде використаний ігровий рушій Unreal Engine 5. Була обрана версія 5.3.2, оскільки вона надає доступ до найновіших функцій, що дозволяє створити більш якісну графіку, використовуючи освітлення за допомогою технології Lumen (Global Illumination) та Virtual Shadow Maps, а також краще оптимізувати продукт.

Глобальне освітлення Lumen вирішує розсіяне непряме освітлення. Наприклад, світло, що розсіюється від поверхні, набирає колір цієї поверхні і відбиває це кольорове світло на інші близькі поверхні, створюючи ефект, який називається кольоровим вициканням. Меші у сцені також блокують непряме освітлення, що також створює непрямі тіні [7].

Virtual Shadow Maps (VSMs) - новий метод картування тіней, який використовується для створення послідовних, високороздільних тіней, що працюють з ресурсами кіноякості та великими, динамічно освітленими відкритими світами, використовуючи функції Nanite Virtualized Geometry, Lumen Global Illumination and Reflections та World Partition у Unreal Engine 5 [8].

В якості мови програмування було використано вбудовану мову програмування Blueprints. Для створення локацій 3D об'єкти будуть взяті з безкоштовних ассетів в Unreal Engine Marketplace.

3 АРХІТЕКТУРА ТА ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 UML проєктування ПЗ

Перед тим, як розпочати розробку ігрового програмного забезпечення у жанрі 3D Rogue-like шутер від третьої особи, були визначені основні дії, які може виконувати гравець. Після докладного аналізу була створена діаграма використання Use-case (див. рис. 3.1).

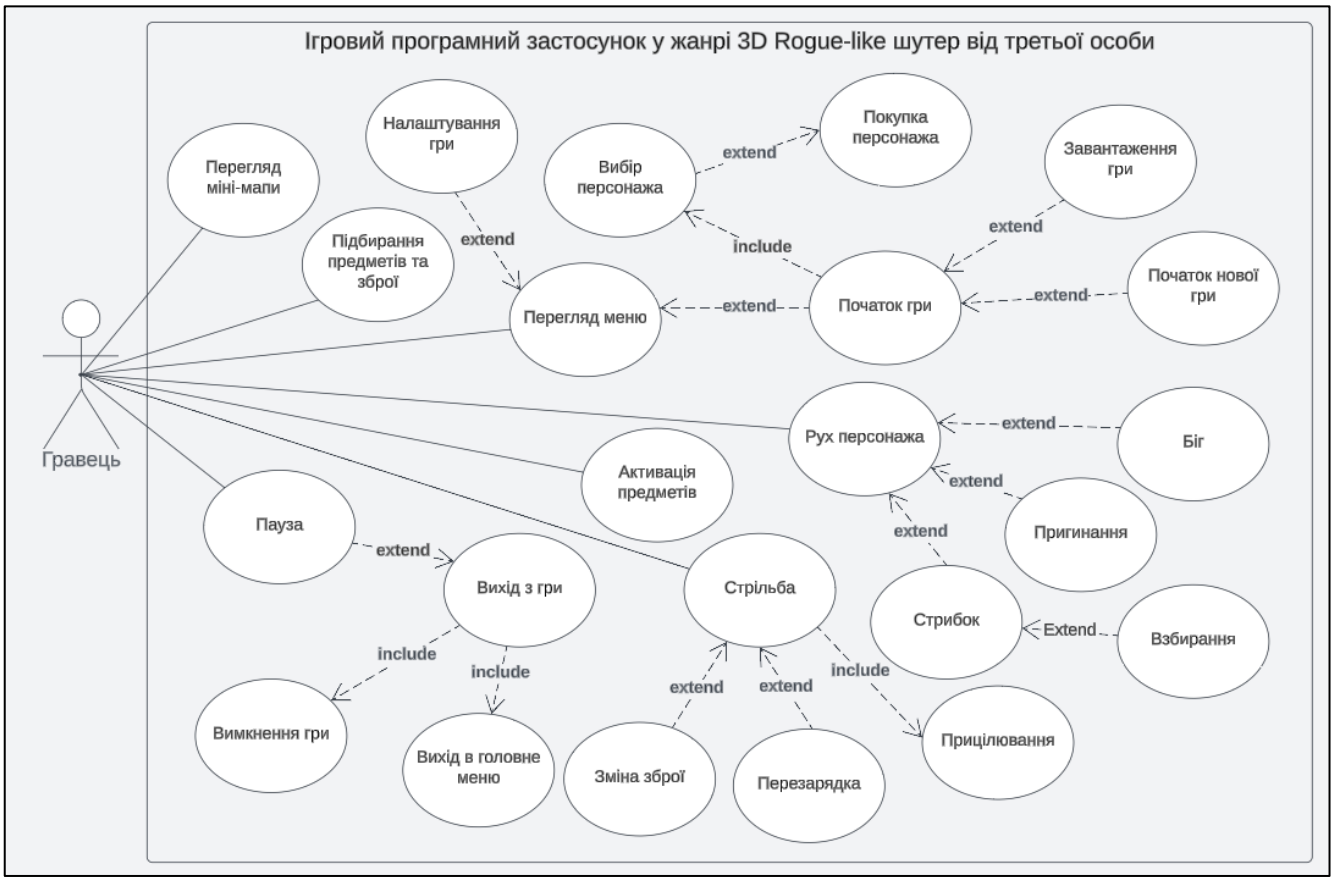


Рисунок 3.1 – Діаграма прецедентів основних функцій зі сторони гравця

Гравець використовує цей ігровий застосунок. У головному меню він може налаштувати гру, купити або вибрати персонажа, розпочати нову гру або завантажити попередню. У грі гравець може керувати персонажем, переміщуючись у різні напрямки, бігаючи, стрибаючи, пригаючи та взбираючись на уступи. Під час гри він

може знаходити нові предмети та зброю, які може підбирати. Персонаж може використовувати підібрані предмети, стріляти, перезаряджати зброю, прицілюватись та змінювати зброю. Крім того, гравець може переглядати міні-мапу для визначення свого положення та прогресу, поставити гру на паузу, повернутися назад у головне меню або повністю вийти з гри. Гравець також може змінювати налаштування відео, аудіо та мапування клавіш у налаштуваннях. Крім цього, він може обирати мову між англійською та українською.

3.2 Вибір архітектури та рушія

При порівнянні Unreal Engine 5 та Unity для розробки 3D шутеру Rogue-like у стилі кіберпанк слід звернути увагу на декілька ключових аспектів, щоб зробити обґрунтований вибір.

По-перше, Unreal Engine 5 відомий своєю потужною графікою та візуальною якістю, що особливо важливо для створення іммерсивного кіберпанк відтворення. Технологія Lumen, що вперше була представлена в UE5, дозволяє динамічно розраховувати освітлення у реальному часі, що робить оточення більш реалістичним та живим. Це дозволяє досягти вражаючих ефектів освітлення та тіней, що додає атмосферності грі.

Крім того, технологія Virtual Shadow Mapping в UE5 дозволяє створювати більш точні тіні з високою деталізацією та реалістичністю, що допомагає поглибити візуальний досвід гравців. Ця технологія підтримується апаратними можливостями сучасних графічних карт, що робить її ефективною для використання у великих ігрових проектах, таких як шутер Rogue-like у стилі кіберпанк.

У порівнянні з Unity, Unreal Engine 5 має більш розвинуті та передові графічні можливості, що робить його більш привабливим вибором для проектів, які акцентуються на високій якості візуального виконання. Таким чином, на підставі аналізу зазначених функцій та можливостей, Unreal Engine 5 є кращим варіантом для

розробки 3D шутеру Rogue-like у стилі кіберпанк.

3.3 Приклади найцікавіших алгоритмів та методів

У межах компоненту комплексної роботи ігрового застосунку 3D шутеру Rogue-like у стилі кіберпанк розглядається алгоритм генерації об’єктів та освітлення (див. рис. 3.2).

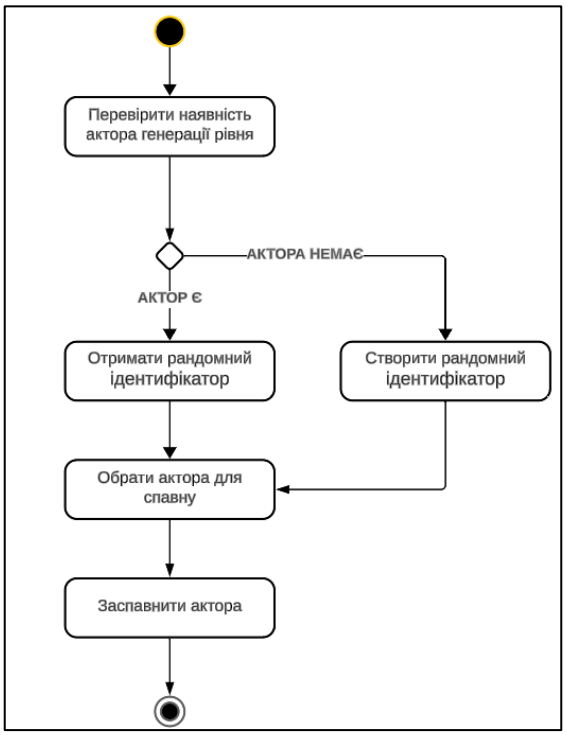


Рисунок 3.2 – Action-діаграма алгоритму генерації об’єктів та освітлення

Після запуску алгоритму відбувається перевірка на існування актора генерації рівня «Level Master». Якщо він існує, то з нього береться випадкове число «Random Stream Seed» дозволяє генерувати повторювані випадкові числа та використовувати їх у Blueprints і Level Blueprints, а також у AnimBlueprints для анімації [9], а якщо ні - то створюється випадкове. Далі за допомогою цього числа з масиву «All Actors» вибирається випадковий об'єкт, який потім спавниться у встановлених координатах

та з визначеним поворотом та масштабом.

3.4 Створення UI/UX

Для створення більшої атмосфери кіберпанку в грі передбачено різноманітні локації з випадково генерованими елементами. Наприклад, локація боса має різні варіанти генерації освітлення та предметів. Крім того, на цих локаціях використовуються генеровані труби, щоб забезпечити гравцю більше задоволення від гри (див. рис. 3.3 та 3.4).

Таким чином, коли гравець проходить локацію багато разів, він повинен адаптуватися до змін місцевості, освітлення та темпу гри.

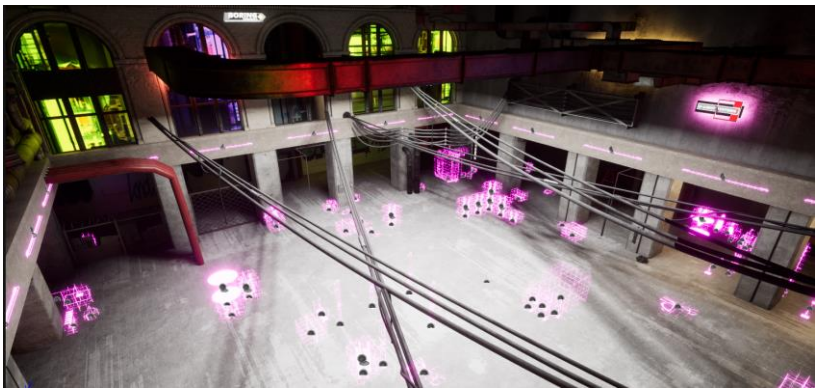


Рисунок 3.3 – Локація боса у демо-рівні

Рисунок 3.4 – Локація боса під час запуску рівня

Після того як персонаж переможе або загине, на екрані з'являться відповідні інтерфейси з привітанням або повідомленням про поразку (див. рис. 3.5 та 3.6). Після цього гравцю пропонується повернутися в головне меню.

Рисунок 3.5 – Інтерфейс перемоги

Рисунок 3.6 – Інтерфейс поразки

У головному меню гравець може обрати продовжувати гру чи почати з самого початку у режимі «CLASSIC», але при релізі самих продуктів гравці не зможуть отримати режими гри «TOWER» та «SURVIVAL». У найближчих оновленнях будуть додаватися самі оновлення та нові режими. Також з головного інтерфейсу у головному меню гравець може перейти до інтерфейсу обрання персонажів, налаштувань та

покинути гру (див. рис. 3.7).

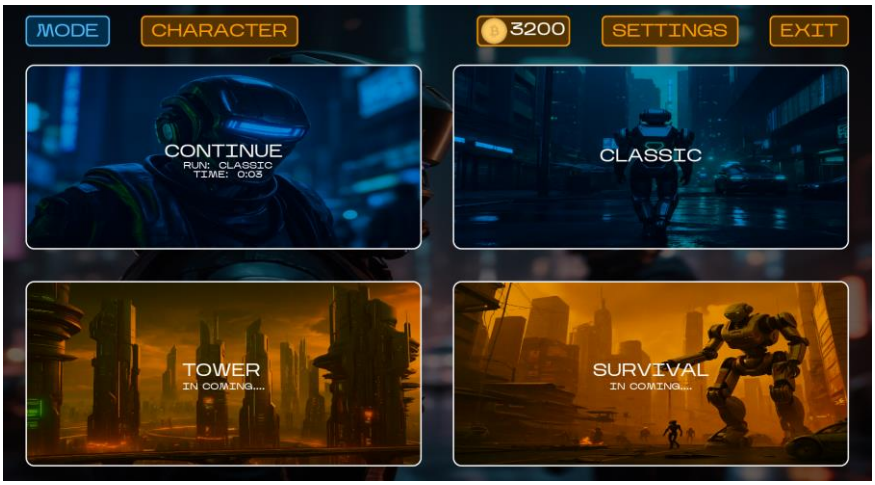


Рисунок 3.7 – Головний інтерфейс меню

У меню обрання чи покупки персонажів гравець може отримати детальну інформацію про бонуси, що надаються самим персонажем та його зброєю. Крім того, гравець може придбати цих персонажів (див. рис. 3.8).

Рисунок 3.8 – Інтерфейс придбання та обрання персонажів

Гравець може налаштовувати параметри для забезпечення більш комфортної гри. Крім того, він може обирати локалізацію, що найбільше підходить йому, і всі текстові поля будуть змінені як у самому меню, так і у грі (див. рис. 3.9-3.11).

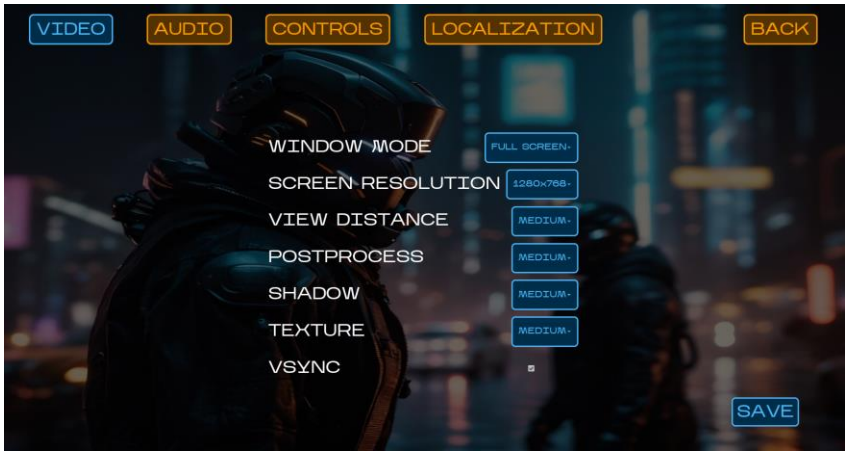


Рисунок 3.9 – Інтерфейс налаштувань графіки

Рисунок 3.10 – Інтерфейс зміни клавіш

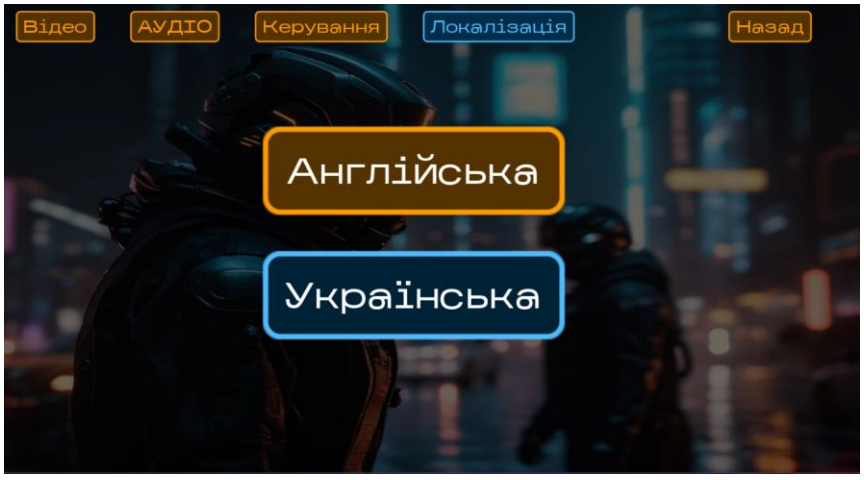


Рисунок 3.11 – Інтерфейс зміни локалізації

Якщо гравець вирішить покинути гру, для попередження перед виходом з'явиться інтерфейс підтвердження дії (див. рис. 3.12).

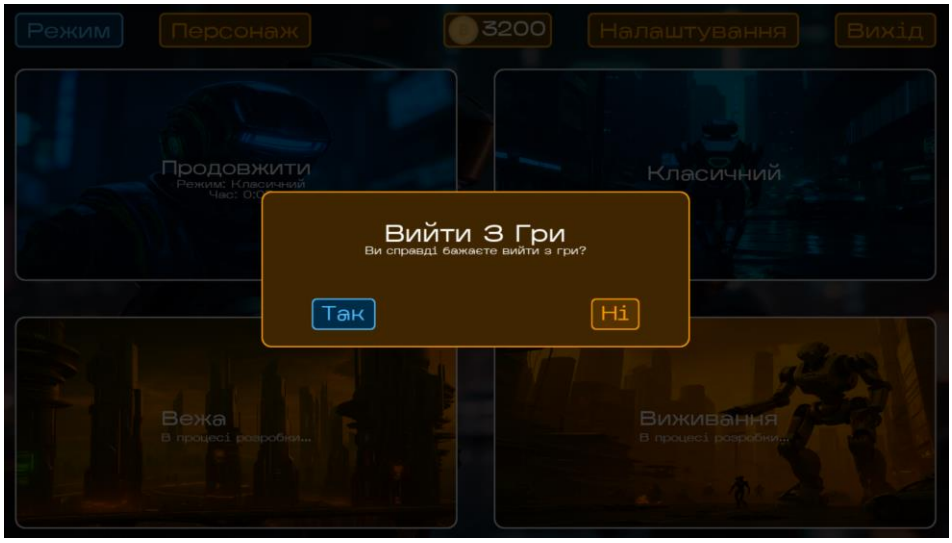


Рисунок 3.12 – Інтерфейс підтвердження дії

При завантаженні та запуску самого продукту гравець повинен зачекати та натиснути будь-яку клавішу для переходу або до інтерфейсу головного меню, або до самої гри (див. рис. 3.13 та 3.14).

Рисунок 3.13 – Інтерфейс запуску гри

Рисунок 3.14 – Інтерфейс завантаження гри

Для забезпечення комфортної гри гравець може відкривати вікно паузи, щоб змінити налаштування, покинути гру та повернутися до головного меню. При цьому весь ігровий процес автоматично ставиться на паузу (див. рис. 3.15).

Рисунок 3.15 – Інтерфейс меню паузи

На погляд нашої команди, саме така стилістики зробить атмосферу гри більш яскравою та комфортною для гравців.

4 ОПИС ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ

4.1 Система генерації об’єктів та світла

У межах компоненту комплексної роботи ігрового застосунку 3D шутеру Rogue-like у стилі кіберпан к розглядатимуться цікаві алгоритми, такі як генерація предметів та світлення, створення труб з використанням технології «Spline», а також створення наративу.

Спершу розглянемо процес генерації предметів та освітлення. Кожна створена папка містить спавнер (дитину актора «ActorsArray»), а також різні варіації предметів або освітлення (дитини актора «DecorInstanceParent») (див. рис. 4.1).

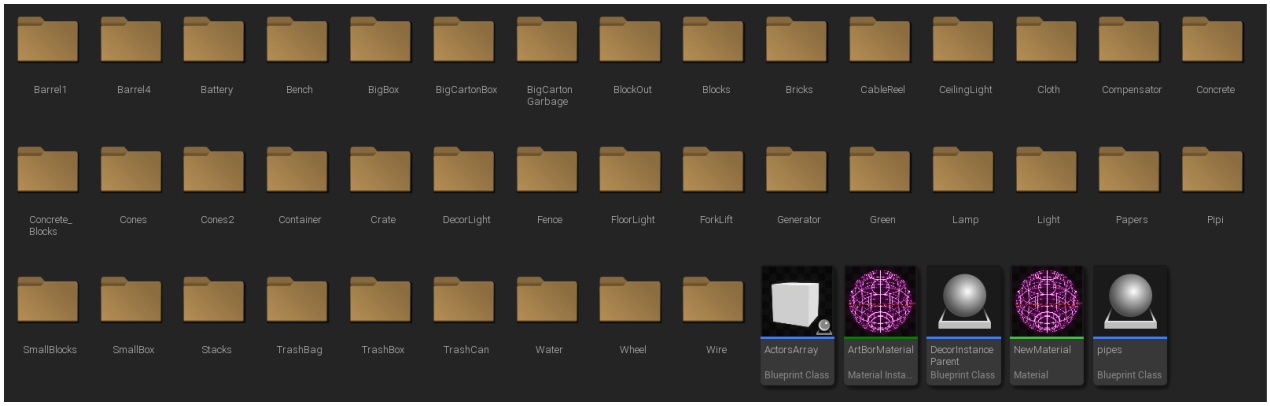


Рисунок 4.1 – Варіації генерацій предметів та освітлення

По-друге, розглянемо дитину актора спавнера, що містить інформацію про всі варіації дитин актора «DecorInstanceParent». Крім того, сам спавнер має загальну форму предмета або освітлення для якісного та детального створення левел-дизайна. На самому акторі спавнера створено окремий матеріал з неоново-рожевої решітки для відокремлення левел-дизайна та геймплея (див. рис. 4.2).

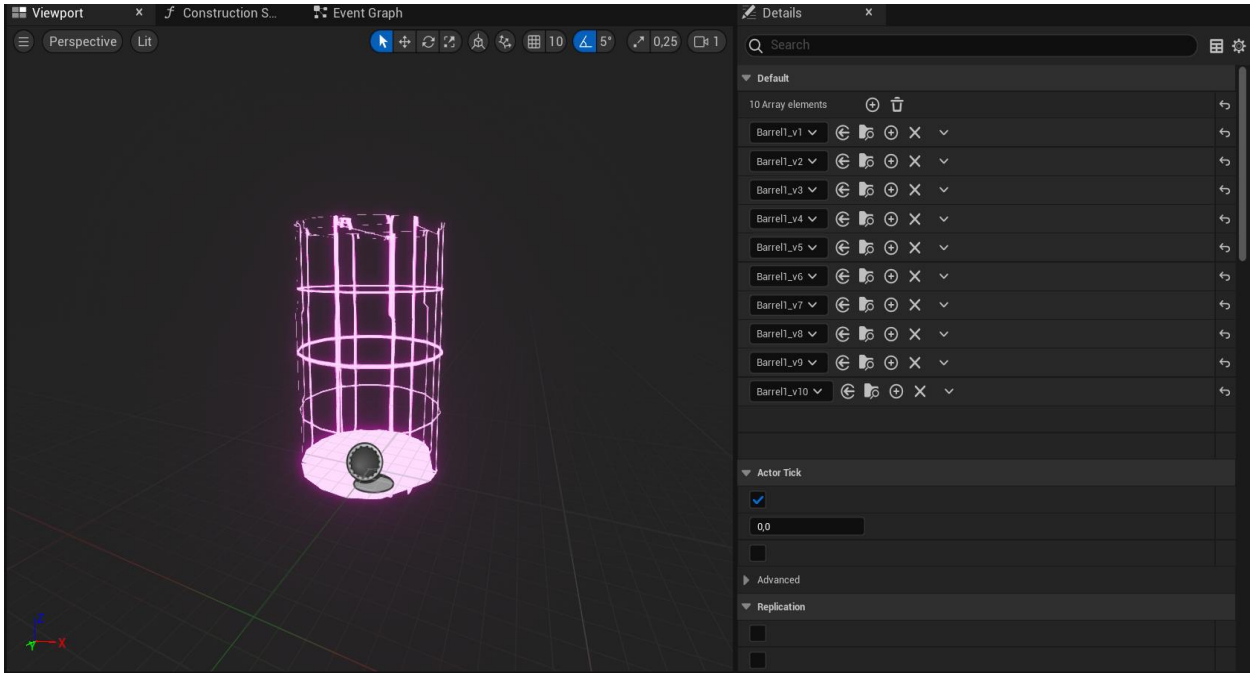


Рисунок 4.2 – Дитина актор спавнера

Далі розглянимо один із варіантів бочки в дитині актора «DecorInstanceParent» (див. рис. 4.3). Кожен із варіантів може мати різний вигляд, колір та форму, але вони усі повинні підходити до базової форми та розміров.

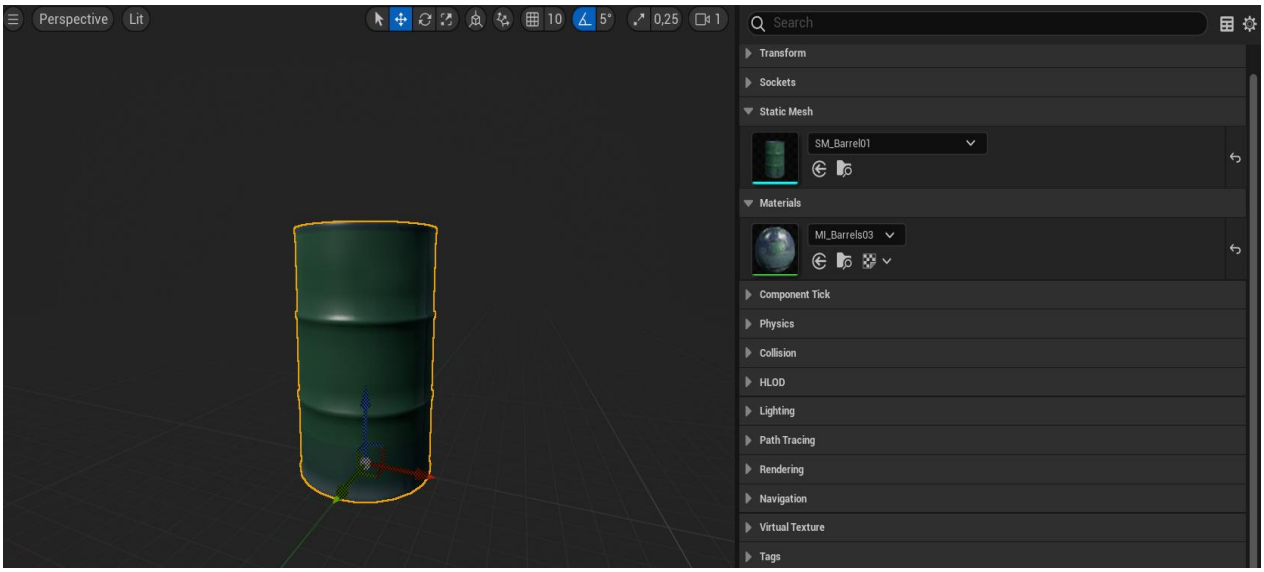


Рисунок 4.3 – Дитина актора «DecorInstanceParent»

Тепер розглянемо процес генерації (див. рис. 4.4).

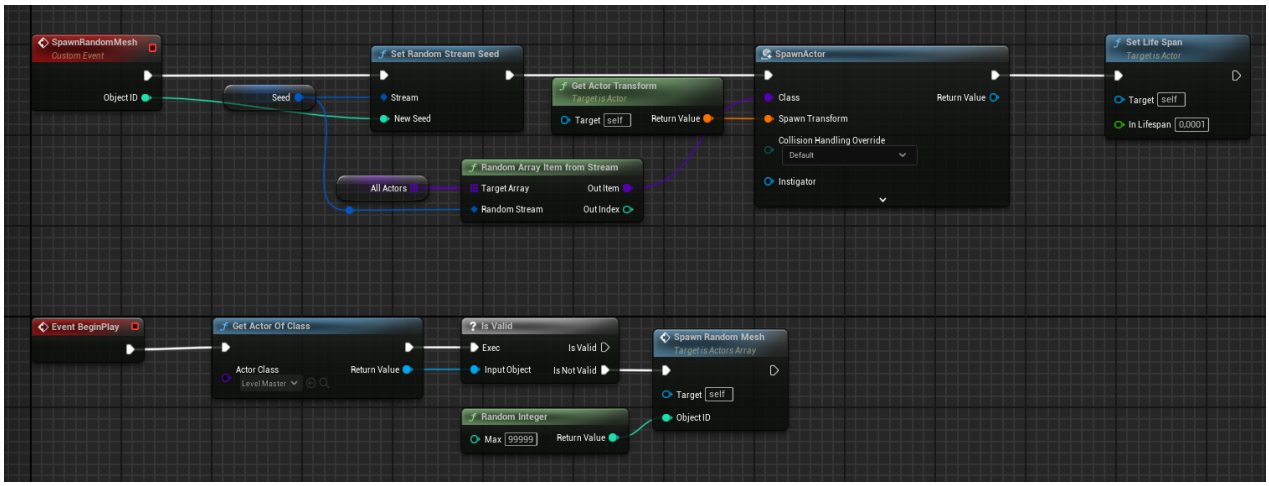


Рисунок 4.4– Логіка генерації об’єктів та освітлення

Після запуску алгоритму перевіряється наявність актора генерації рівня «Level Master». Якщо він існує, з нього береться випадкове число «Random Stream Seed». Якщо його немає, генерується випадкове число. Використовуючи це число, з масиву «All Actors» обирається випадковий об'єкт, який потім спавниться на заданих координатах із певним поворотом і масштабом

4.2 Система створення труб

Розглянемо створення труб які використовують матеріали труб з загального масиву усіх об’єктів. Перш ніж перейти до розглядання самої логіки треба створити компонент «Spline» це лише шлях для визначення та використання позиційних даних[10], далі перейдемо до самої функції. Цей компонент має початкову та кінцеву точку тому після отримання цих точок ми одразу віднімаємо 2 після чого створюємо SplineMeshComponent та додаємо в нього обраний матеріал з масиву асетів після чого обираємо напрямок побудови труб та отримуємо початкові та кінцеві точки після кожного додавання блоку до труби та в самому кінці додаємо колізію (див. рис. 4.5).

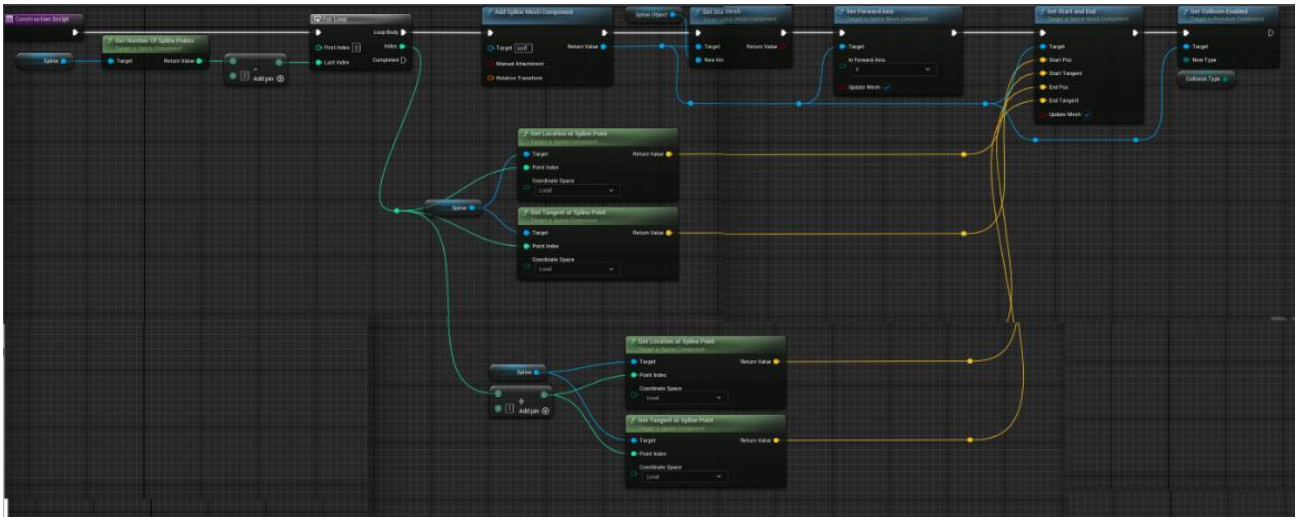


Рисунок 4.5 – Логіка створення труб

4.3 Система створення наративу

Давайте детальніше розглянемо створення наративу, де ми маємо кілька ключових компонентів. Перш за все, віджет для завантаження відео. Він буде відповідати за відтворення відео-контенту. Далі йдуть самі матеріали для цього відео, які забезпечують необхідний контент для відтворення.

Крім того, потрібні плеєри, які оброблятимуть і відтворюватимуть відеоматеріали. Важливим елементом є актор порожнього персонажа, через якого буде завантажуватися і виконуватися необхідний скрипт для управління процесом (див. рис. 4.6).

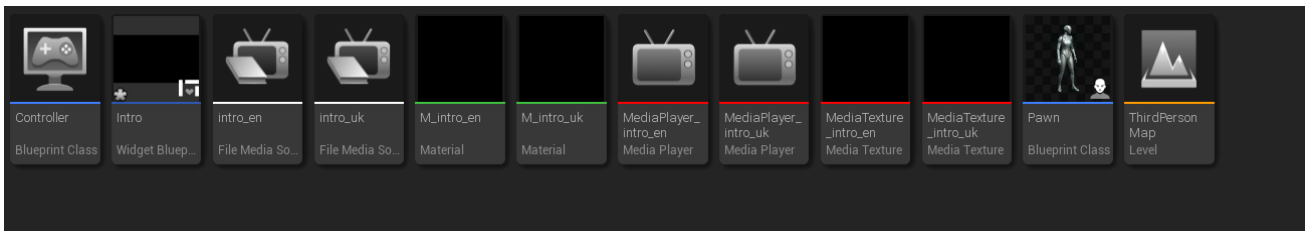


Рисунок 4.6 – матеріали для створення наративу

Спочатку скрипт визначає локалізацію, яка наразі може бути англійською або

українською. Дефолтною локалізацією є англійська. Для кожної локалізації було створено два відео з відповідними коментарями. Під час вибору локалізації скрипт завантажує відеоплеєр і аудіоплеєр. Для розпізнання локалізації вибирається відповідний відеоплеєр і аудіоплеєр, після чого створюється віджет, який додається до вьюпорта (див. рис. 4.7 та 4.8).

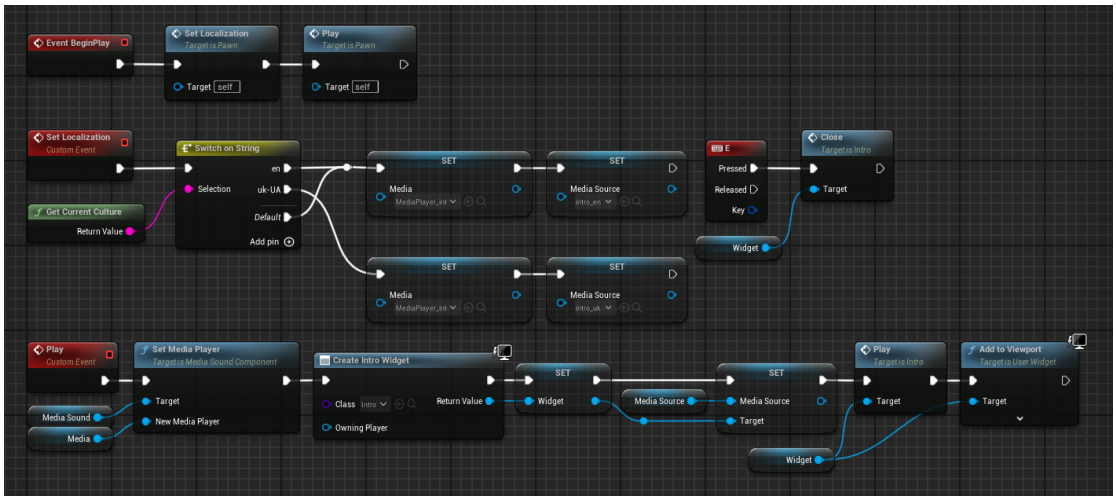


Рисунок 4.7 – Логіка обрання та запуску плеєрів

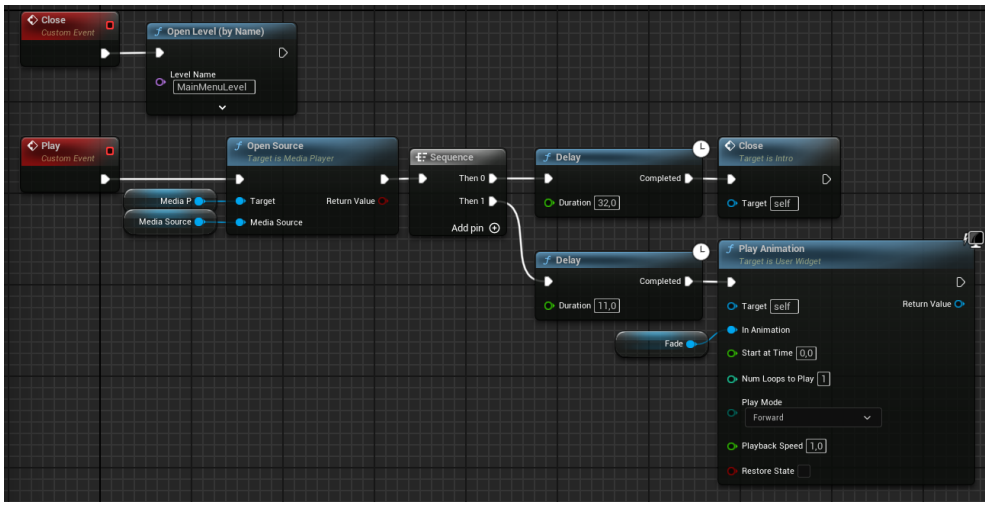


Рисунок 4.8 – Логіка часу відтворення плеєрів

Після 11 секунди відтворення з'являється анімація пропуску катсцени, і гравець може завершити перегляд відео, натиснувши клавішу «E».

4.4 Балансування

Під час балансування було вирішено поділити зброю за ієрархією від звичайного до легендарного. Зброя може бути як легкою, наприклад, УЗІ чи пістолет, так і важкою, як снайперська гвинтівка чи мінімган. Кожна зброя має наступні параметри: патрони в магазині, загальна кількість патронів, швидкість перезарядки, шкода, відстань, розкид, скорострільність та віддача. Проте гравцю будуть показані наступні параметри: шкода, точність, дистанція, скорострільність та мобільність. За допомогою штучного інтелекту була створена формула для визначення нормованого значення, де:

– 0 - мінімальне,

– 100 - максимальне.

Після отримання значень кожного параметра проводилася балансування у наступних діапазонах: від 0.8 до 0.7 для звичайної зброї, від 0.7 до 0.6 для рідкісної, від 0.6 до 0.4 для епічної та від 0.4 до 0.2 для легендарної(див. рис.4.9.1 та 4.9.2).

НАЗВА		ПАТРОНІВ У МАГАЗИНІ	УСЬОГО ПАТРОНІВ(max)	ШВИДКІСТЬ ПЕРЕЗАРЯДКИ	ШКОДА	ВІДСТАНЬ	РОЗКИД	СКОРОСТРІЛЬНІСТЬ	ВІДДАЧА
ТИП	МЕХАНІЧНА ЧАСТИНА	ЗВИЧАЙНИЙ							
AK-47		24	120	0,9	50	1500	500	0,2	0,15
SMG		40	220	1,1	40	4000	750	0,05	2
Sniper Rifle		6	30	0,3	200	8000	1	1	8
Shotgun		6	36	1	250	500	800	0,5	6
Rifle		30	150	1	28	5000	300	0,1	0,1
Pistol		12	72	1	65	1000	300	0,12	1,2
Starpiercer		4	50	0,9	300	10000	1	0,55	10
Sunburst		10	60	0,3	170	4000	1000	0,4	4
Photon Pulse		15	90	0,3	100	5000	200	0,3	1
Heavy Laser Rifle		20	120	1	70	2000	600	0,21	1,6
ТИП		РІДКИЙ							
Soul Reaper		10	80	1	100	2000	100	0,35	0,5
Viper		100	500	1	20	2000	400	0,03	1
Grenade Launcher		8	32	1	200	1000	100	0,8	1,5
Plasma Pistol		18	200	1,6	210	2000	1500	0,2	2
Railgun		6	40	0,9	300	8000	100	0,5	0,9
ТИП		ЕПІЧНИЙ							
Rocket Launcher		1	20	0,6	370	2000	1	0,5	6
Ghostbullet		8	60	0,8	150	5000	1	0,55	5
Vortex Rifle		100	400	1	50	2000	500	0,15	0,5
Thunderbolt		6	60	1	150	2000	200	0,35	0,5
ТИП		ЛЕГЕНДАРНИЙ							
Destroyer		4	30	0,5	400	5000	50	1,5	25
Destroyer Jr.		3	40	1	300	8000	75	1	20
Minigun		200	600	0,6	20	5000	700	0,01	0,5
Salvo		4	100	0,2	500	2000	4000	1	8
Pulse Rifle		130	500	1	15	2000	600	0,02	0,8

Рисунок 4.9.1 – Механічні параметри балансування зброї

НАЗВА		ШКОДА	ТОЧНІСТЬ	ДИСТАНЦІЯ	СКОРОСТРІЛЬНІСТЬ	МОБІЛЬНІСТЬ	ЦІНА	КОЛІР	РАХУНОК	ШКОДА/С
ТИП	ВІЗУАЛЬНА ЧАСТИНА	ЗВИЧАЙНИЙ							(0.8 - 0.7)	
AK-47		8	88	11	87	25	300		0,72	250
SMG		6	61	37	97	25	300		0,73	800
Sniper Rifle		39	98	79	34	18	400		0,75	200
Shotgun		49	43	1	67	38	320		0,71	500
Rifle		4	93	48	94	54	360		0,75	280
Pistol		11	76	6	93	16	270		0,76	542
Starpiercer		59	98	100	64	47	530		0,76	545
Sunburst		33	46	37	74	12	300		0,76	425
Photon Pulse		18	81	48	81	15	320		0,79	333
Heavy Laser Rifle		12	66	17	87	50	300		0,73	333
ТИП		РІДКИЙ							(0.7 - 0.6)	
Soul Reaper		18	90	17	77	53	410		0,60	286
Viper		2	75	17	99	32	350		0,60	667
Grenade Launcher		39	83	6	48	54	350		0,69	250
Plasma Pistol		41	51	17	87	95	420		0,68	1050
Railgun		59	87	79	67	37	550		0,67	600
ТИП		ЕПІЧНИЙ							(0.6 - 0.4)	
Rocket Launcher		73	99	17	67	6	700		0,48	740
Ghostbullet		29	100	48	64	51	680		0,43	273
Vortex Rifle		8	79	17	91	100	560		0,45	333
Thunderbolt		29	86	17	77	41	575		0,45	429
ТИП		ЛЕГЕНДАРНИЙ							(0.4 - 0.2)	
Destroyer		80	63	48	1	45	950		0,31	267
Destroyer Jr.		59	61	79	34	68	850		0,38	300
Minigun		2	76	48	100	1	910		0,25	2000
Salvo		100	1	17	34	37	950		0,28	500
Pulse Rifle		1	73	17	99	22	720		0,28	750

Рисунок 4.9.2 – Візуальні параметри балансування зброї

Тепер детально розглянемо формулу для параметра балансування "рейтинг": =((L3*2) + M3 + N3 + O3 + (P3*0.5)) / Q3. У цій формулі ми перемножуємо всі параметри зброї на їх коефіцієнти, а потім ділимо на ціну самої зброї, таким чином отримуючи параметри в заданому діапазоні. Наступним параметром розглянемо формулу для "шкода за секунду" - =F3/I3, де параметр "шкода" ділиться на скорострільність.

Перейдемо до таблиці балансування ворогів. Кожен ворог ділиться на легкого і швидкого або повільного і важкого та має спеціальну зброю, відповідно до їх характеристик, від УЗІ до мінімгана. Кожен тип ворога має такі параметри: "HP, шкода, скорострільність, шкода за секунду, час до смерті (сек), час до вбивства гравця". Розглянемо формулу для параметра "HP": =СРЗНАЧ($T$3:$T$12; $T$14:$T$18; $T$20:$T$23; $T$25:$T$29) * F32 * 0.4, де середнє значення "шкода за секунду" від зброї гравця множиться на час до смерті ворога і на коефіцієнт влучання(див. рис.4.10).

НАЗВА	HP	ШКОДА	СКОРОСТРІЛЬНІСТЬ	ШКОДА/С	Час, щоб вмерти (сек)	Час щоб вбити гравця (сек)
AI_Heavy_Laser_Rifle	464	67	0,35	191	2,20	5,5
AI_Minigun	1687	14	0,05	280	8,00	4
AI_Pistol	190	37	0,25	148	0,90	7
AI_Rifle	211	24	0,15	160	1,00	6,5
AI_SMG	148	14	0,08	175	0,70	6

Рисунок 4.10 – Параметри ворогів

Перейдемо до таблиці персонажів. На даний момент у грі є лише три персонажі, кожен з яких має такі параметри: модифікатор HP, відсоток броні (%), HP, броня та кількість одиниць шкоди, необхідна для вбивства гравця(див. рис.4.11).

НАЗВА	HP Модифікатор	Відсоток броні (%)	HP	Броня	AI_Heavy_Laser_Rifle	AI_Minigun	AI_Pistol	AI_Rifle	AI_SMG
character1	1	30	741,84	317,93	1052,86	1120	1036	1040	1050
character2	0,7	60	296,736	445,10	737,00	784	725,2	728	735
character3	1,5	20	1271,725714	317,93	1579,29	1680	1554	1560	1575

Рисунок 4.11 – Параметри персонажів

Щоб дізнатися HP персонажа, потрібно взяти середнє значення шкоди та помножити його на коефіцієнт броні. Формула виглядає так: =СРЗНАЧ(F39:J39)*((100-C39)/100).

Також є формула для розрахунку одиниць броні відносно середнього урону від ворогів: =СРЗНАЧ(F39:J39)*(C39)/100.

І нарешті, формула для отримання одиниць шкоди від різних типів противників, з урахуванням модифікатора HP, що множиться на "шкоду за секунду" та час, необхідний для вбивства гравця.

5 ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

5.1 Розробка мапи думок тестування

Тестування програмного забезпечення гри здійснювалося згідно з тест-планом, наведеним у додатках, та мапою думок, показаною на рисунку 5.1.

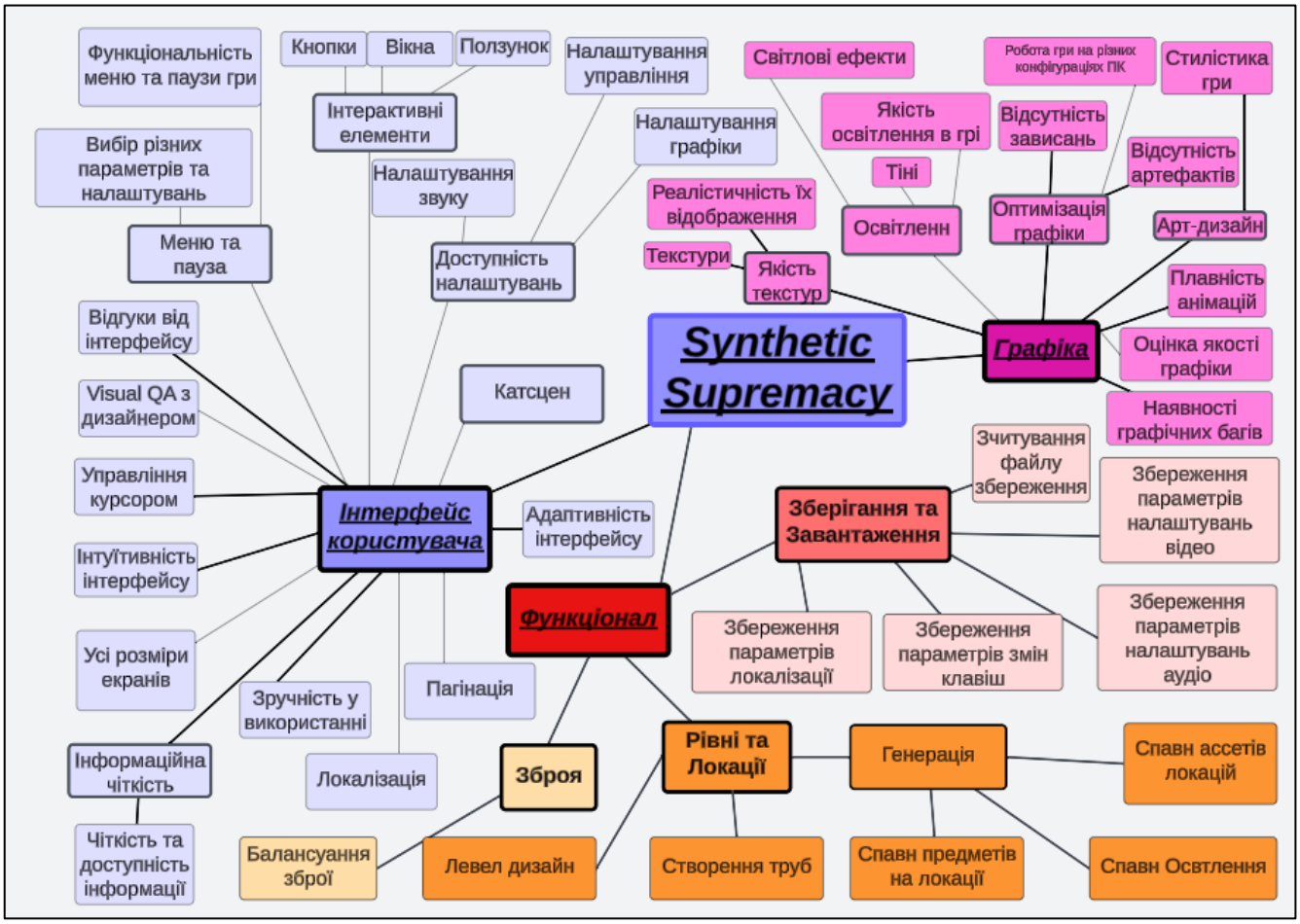


Рисунок 5.1 - Mind Map для тестування проекту

Механіки та функціонал тестувалися за методологією «Білої скриньки» - це категорія тестування програмного забезпечення, яка відноситься до методів тестування внутрішньої структури та дизайну програмного забезпечення [11], оскільки ми знайомі з внутрішньою структурою системи. Інтерфейс, навпаки, перевіряли за методологією «Чорної скриньки» - це процес тестування системи або

програмного забезпечення без попереднього знання того[12], як вона працює зсередини, адже основну увагу приділяли візуальному вигляду.

На цій діаграмі можна виділити наступні частини проекту, які потребують тестування:

– Інтерфейс користувача: Кат-сцени, адаптивність інтерфейсу, пагінація, локалізація, зручність у використанні, усі розміри екранів, інтуїтивність інтерфейсу, відгуки від інтерфейсу, меню та пауза, інформаційна чіткість, інтерактивні елементи, доступність налаштувань.

– Графіка: Освітлення, якість текстур, оптимізація графіки, арт-дизайн, плавність анімацій, оцінка якості текстур, наявність графічних багів, стилістика гри, відсутність зависань, світлові ефекти, тіні та якість осветлення.

– Функціонал: Зброя, рівні та локації, зберігання та завантаження.

Тож, суцільний програмний модуль був сегментований на окремі категорії, щоб його можна було протестувати.

5.2 Розробка тестових випадків

На основі класифікації, що випливає з Mind Map-діаграми, був складений перелік тестових випадків для комплексного та вичерпного тестування системи. Кожен тест-кейс містить опис ситуації, очікуваний та фактичний результат, а також можливі додаткові коментарі.

Таблиці 5.1 - 5.5 містять список тест-кейсів, які були розроблені під час тестування програмного модуля. Пріоритет тест-кейсу визначається значенням від P1 (найвищий) до P4 (найнижчий). Критичність потенційного багу оцінюється від S1 (некритичний) до S4 (руйнує геймплей).

Таблиця 5.1 – Тест у випадку з актором світла

Тест № 1	
Назва тесту:	Актор світла проходить крізь текстуру будинка
Опис тесту:	Після додавання актора світла для вітрини світло проходить крізь текстури час натискання будь-яких клавіш навігації камери.
Компонент системи:	Левелдизайн(Створення рівнів)
Пріоритет:	P2
Критичність:	S2
Кроки відтворення:	Після додавання актора світла для вітрини світло проходить крізь текстури
Очікуваний результат:	Світло не проходить крізь текстури
Фактичний результат:	Світло проходить крізь текстури

Таблиця 5.2 – Тест у випадку простора у текстурах стін

Тест № 2	
Назва тесту:	Простір між текстурами стін
Опис тесту:	При створенні локацій розробники підставили різні по ширині стіни
Компонент системи:	Левелдизайн(Створення рівнів)
Пріоритет:	P1
Критичність:	S3
Кроки відтворення:	При створенні локацій розробники підставили різні по ширині стіни
Очікуваний результат:	Немає отвору у стінах
Фактичний результат:	Є отвір у стінах

Таблиця 5.3 – Тест у випадку наплива графіті на текстуру асетів

Тест № 3	
Назва тесту:	Наплив графіті на текстуру асета
Опис тесту:	При нанесенні графіті на стани у 2 кімнаті графіті налізає на генератор
Компонент системи:	Левелдизайн(Створення рівнів)
Пріоритет:	P3
Критичність:	S2
Кроки відтворення:	При нанесенні графіті на стани у 2 кімнаті графіті налізає на генератор

Продовження таблиці 5.3

Очікуваний результат:	Не залізає на асет генератора
Фактичний результат:	Залізає на асет генератора

Таблиця 5.4 – Тест у випадку артефактів у текстурах стін

Тест № 4	
Назва тесту:	Артефакти на стінах та балконах
Опис тесту:	При створенні рівнів було дублювання балконів та стін що сприяло виникненню артефактів
Компонент системи:	Левелдизайн(Створення рівнів)
Пріоритет:	P1
Критичність:	S3
Кроки відтворення:	При створенні рівнів було дублювання балконів та стін що сприяло виникненню артефактів
Очікуваний результат:	Немає безліч копій на одному ріні
Фактичний результат:	Є безліч копій на одному ріні

Таблиця 5.5 – Тест у випадку з неповної локалізації

Тест № 5	
Назва тесту:	Неповна локалізація
Опис тесту:	Після переходу на українську локалізацію назви та самі кнопки для обрання не переходять на українську локалізацію бо локалізація працює тільки з текстовими полями
Компонент системи:	UI(локалізація)
Пріоритет:	P2
Критичність:	S2
Кроки відтворення:	Після переходу на українську локалізацію назви та самі кнопки для обрання не переходять на українську локалізацію бо локалізація працює тільки з текстовими полями
Очікуваний результат:	Локалізація працює зі зміною клавіш
Фактичний результат:	Локалізація не працює зі зміною клавіш

Всі інші випробування проводилися в процесі створення коду. Такі випробування спрямовані на перевірку функціональності різних базових механік або геймплейних особливостей, пов'язаних із функціонуванням демонстраційного додатку.

6 ВПРОВАДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

6.1 Наукове впровадження проекту

На основі тематики кваліфікаційної роботи була написана наукова стаття у форматі тез для XXVIII Міжнародного молодіжного форуму "Радіоелектроніка та молодь у XXI столітті", який проходив з 16 по 18 квітня 2024 року в онлайн-режимі на базі ХНУРЕ.

Стаття під назвою "ОПТИМІЗАЦІЯ ОСВІТЛЕННЯ В UNREAL ENGINE 5" була опублікована у відповідному збірнику конференції №6 [13].

У ній представлені технології LUMEN та Virtual Shadow Mapping, а також проведено їх порівняння з попередніми рішеннями, такими як Ray trace та shadow mapping. Детально розглянуті як переваги, так і недоліки всіх цих технологій. Та зроблено висновок

6.2 Практичне впровадження проекту

На основі матеріалів кваліфікаційної роботи було взято участь у виставці технічної творчості молоді 2024 року, яка проходила на базі ХНУРЕ в онлайн-режимі.

Під час виставки було представлено та захищено роботу за темою "Ігровий програмний застосунок у жанрі 3D Rogue-like шутер від третьої особи, а саме Механіки бойової системи, штучного інтелекту, генерації рівнів, економіки та інтерфейсу бою, а також Левел дизайн, UI/UX, балансування зброї, генерація ігрових об'єктів, наратив" у секції "Ігрові технології" (див. рис. 6.1).

Робота викликала значний інтерес серед відвідувачів виставки. Розробка отримала схвальні відгуки від фахівців у галузі ігрових технологій.


МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

ДИПЛОМ

НАГОРОДЖУЄТЬСЯ

Фурсов Данило
Борисенко Артемій

за I місце

в номінації «Ігрові технології»
на виставці технічної творчості молоді
28-го Міжнародного молодіжного форуму
«Радіоелектроніка та молодь в XXI столітті»

Голова оргкомітету Форуму Ігор РУБАН

18 квітня 2024 р.

Рисунок 6.1 – Нагорода на 28-му Міжнародному молодіжному форумі

Виставка проходила з 16 по 18 квітня 2024 року, і розроблений та презентований продукт посів I місце у своїй категорії [14].

ВИСНОВКИ

Під час роботи над моєю частиною комплексного проекту 3D Rogue-like шутера від третьої особи я провів ретельне дослідження. Я вивчив предметну область, проаналізував основних конкурентів як з точки зору жанру, так і з точки зору реалізації особливих механік. Виокремивши переваги та недоліки кожного з них, я врахував їх при розробці власного ігрового застосування.

Я розробив UI/UX частину, яка включає інтерфейс головного меню, інтерфейс паузи, інтерфейси налаштувань, а також інтерфейси перемоги та поразки. Кожен з них має чітку логіку та інтуїтивно зрозумілий дизайн. Щоб запобігти випадковому виходу з гри, гравцеві буде продемонстровано інтерфейс підтвердження дії.

При проектуванні та створенні левел-дизайну я розробив 9 унікальних локацій, кожна з яких має власне освітлення та наповнення асетами. Для створення особливої атмосфери локацій я розробив систему генерації об'єктів та освітлення. Також для більш якісного опрацювання левел-дизайну я використовував технологію "Spline", яка допомогла мені створити чітку структуру труб на кожній з локацій.

Не менш важливу роль у створенні локацій відіграли графіті. Завдяки дизайнерському рішенню я додав не тільки базові графіті з набору асетів, але й створені спеціально для нашої гри.

Для якісного освітлення я використовував технології "Virtual Shadow Maps" та "Lumen", які допомагають створити більш атмосферну та захоплюючу атмосферу в грі.

Для створення наративу я розробив локалізовані відеоматеріали з коротким описом самої ідеї гри. Для цього я створив спеціальну логіку, яка обирає потрібні матеріали залежно від обраної локалізації, що робить продукт більш детально опрацьованим.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Що таке рогалик-гра? plarium.com. URL: https://plarium.com/ru/glossary/roguelike-games/ (дата звернення: 12.05.2024).
2. Шутер від третьої особи. uk.wikipedia.org. URL: https://uk.wikipedia.org/wiki/Шутер_від_третьої_особи (дата звернення: 12.05.2024).
3. Enter the Gungeon. igrodrom.net. URL: https://igrodrom.net/ua/enter-the-gungeon/ (дата звернення: 12.05.2024).
4. DRL en.wikipedia.org. URL: https://en.wikipedia.org/wiki/DRL_(video_game)
5. Cyberpunk 2077. store.steampowered.com. URL: https://store.steampowered.com/app/1091500/Cyberpunk_2077/?l=ukrainian (дата звернення: 12.05.2024).
6. Beneath Apple Manor. en.wikipedia.org. URL: https://en.wikipedia.org/wiki/Beneath_Apple_Manor (дата звернення: 12.05.2024).
7. Lumen Global Illumination and Reflections in Unreal Engine. dev.epicgames.com. URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/lumen-global-illumination-and-reflections-in-unreal-engine?application_version=5.0 (дата звернення: 12.05.2024)
8. Virtual Shadow Maps in Unreal Engine. dev.epicgames.com. URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/virtual-shadow-maps-in-unreal-engine (дата звернення: 12.05.2024)
9. Random Streams. docs.unrealengine.com. URL: https://docs.unrealengine.com/4.26/en-S/ProgrammingAndScripting/Blueprints/UserGuide/RandomStreams/ (дата звернення: 12.05.2024)
10. Blueprint Spline Components Overview. docs.unrealengine.com. URL: https://docs.unrealengine.com/4.27/en-US/BuildingWorlds/BlueprintSplines/Overview/ (дата звернення: 12.05.2024)

11. Тестування чорних скриньок. www.zaptest.com. URL: https://www.zaptest.com/uk/тестування-чорної-скриньки-що-це-так (дата звернення: 12.05.2024)
12. Тестування білих скриньок. www.zaptest.com. URL: https://www.zaptest.com/uk/тестування-білого-ящика-що-це-таке-як (дата звернення: 12.05.2024)
13. Т. 6. Конференція «Комп'ютерної інженерії та захисту інформації». URL: https://openarchive.nure.ua/handle/document/26351 (дата звернення: 12.05.2024).
14. Каталог виставки технічної творчості молоді [Електронний ресурс]. URL: https://openarchive.nure.ua/handle/document/26355 (дата звернення: 12.05.2024).

ДОДАТОК А

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ

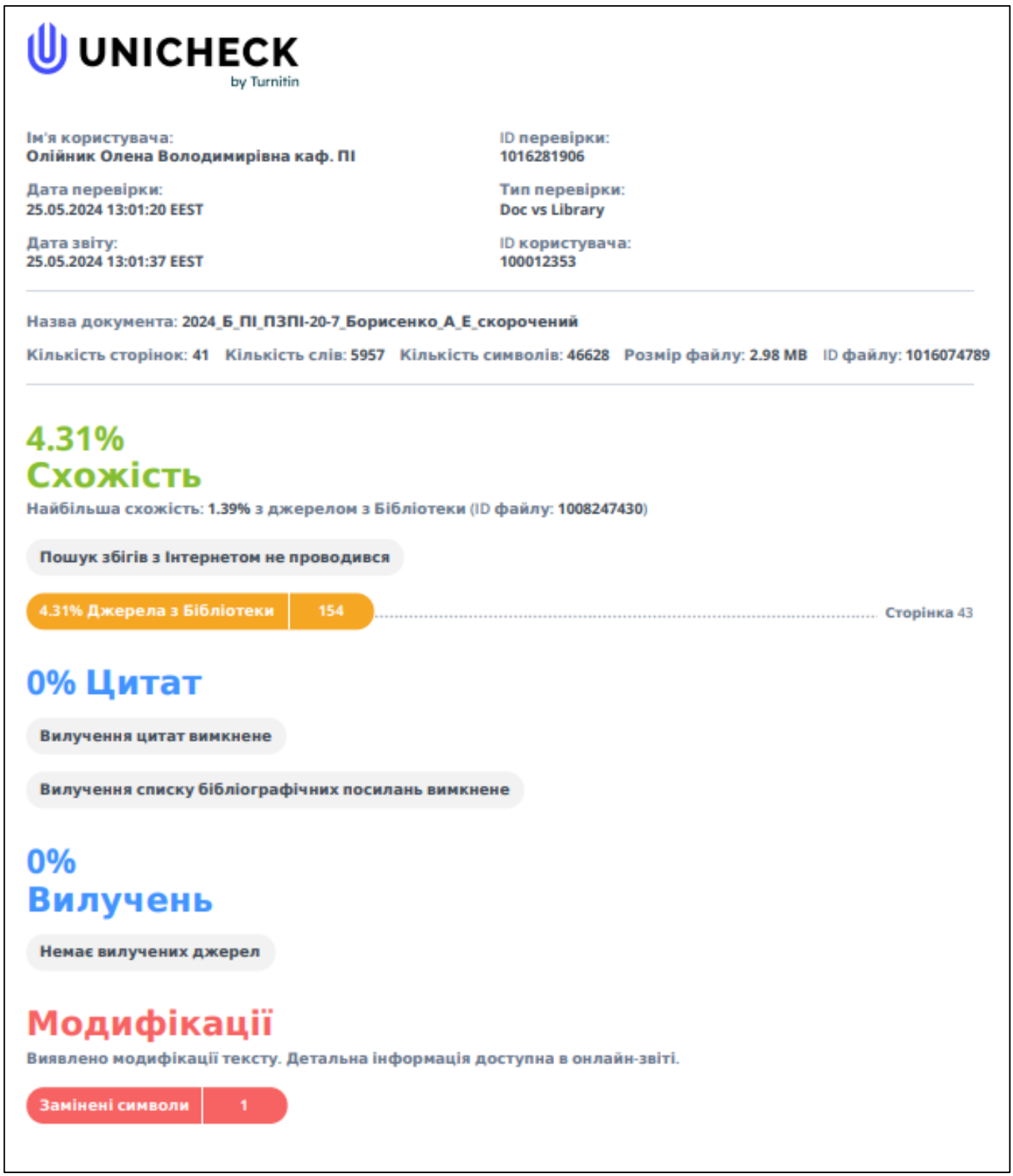


Рисунок А.1 – Перевірка на плагіат

ДОДАТОК Б

Слайди презентації

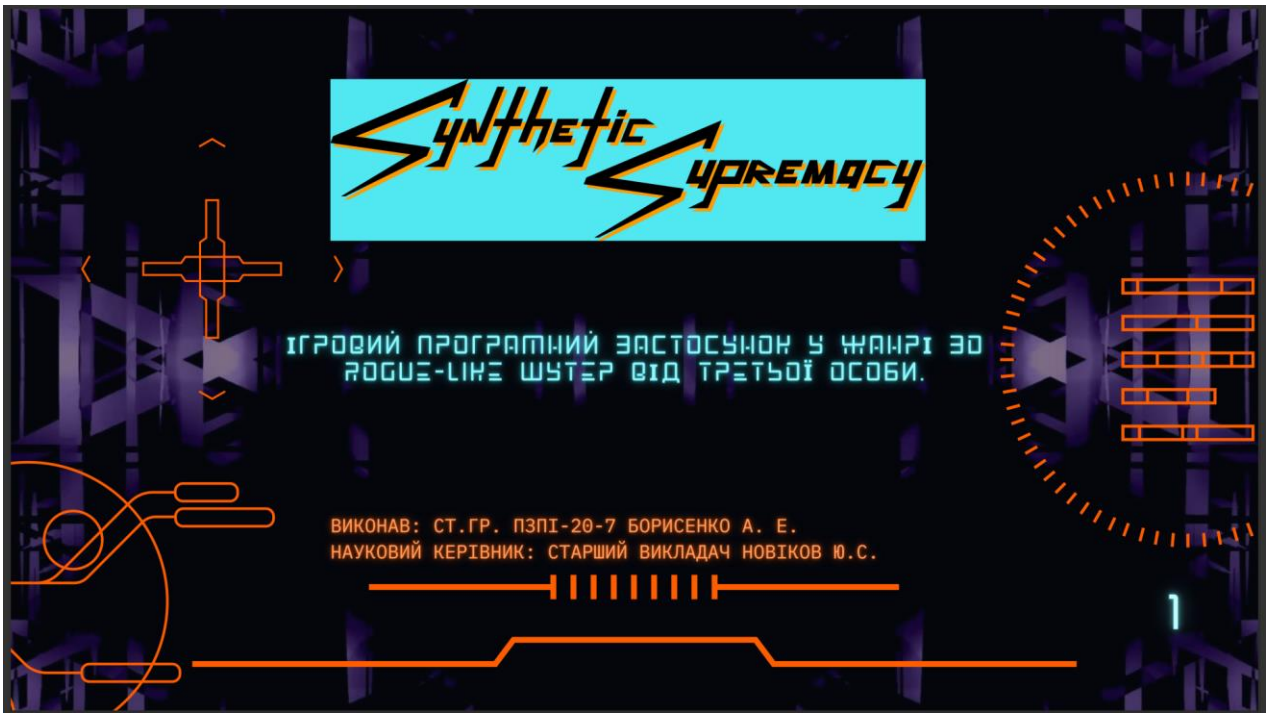


Рисунок Б.1 – Слайд 1

Рисунок Б.2 – Слайд 2

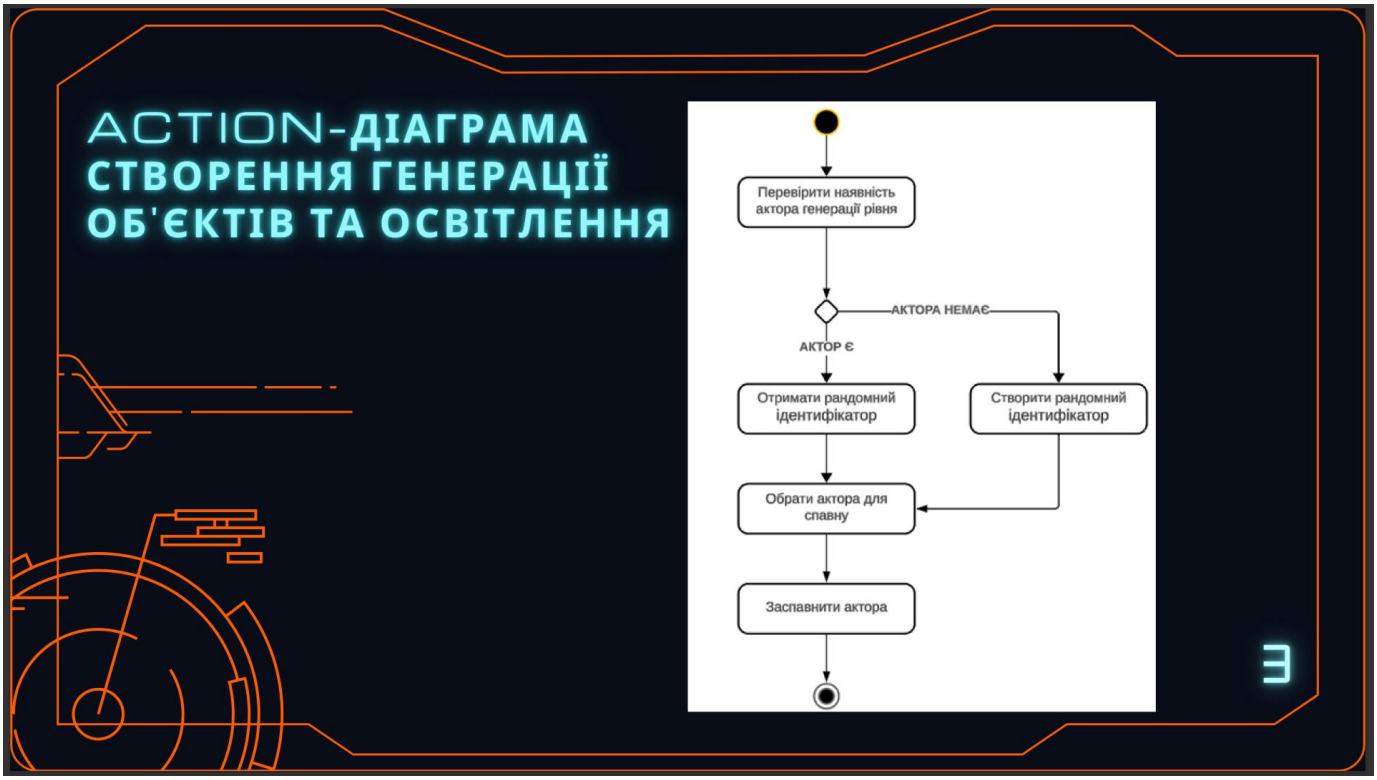


Рисунок Б.3 – Слайд 3

Рисунок Б.4 – Слайд 4

Рисунок Б.5 – Слайд 5

НАЗВА	HP	ШКОДА	СКОРОСТРІЛЬНІСТЬ	ШКОДА/С	Час, щоб вмерти (сек)	Час щоб вбити гравця (сек)
AI_Heavy_Laser_Rifle	464	67	0,35	191	2,20	5,5
AI_Minigun	1687	14	0,05	280	8,00	4
AI_Pistol	190	37	0,25	148	0,90	7
AI_Rifle	211	24	0,15	160	1,00	6,5
AI_SMG	148	14	0,08	175	0,70	6

НАЗВА	HP Модифікатор	Відсоток броні (%)	HP	Броня	AI_Heavy_Laser_Rifle	AI_Minigun	AI_Pistol	AI_Rifle	AI_SMG
character1	1	30	741,84	317,93	1052,86	1120	1036	1040	1050
character2	0,7	60	296,736	445,10	737,00	784	725,2	728	735
character3	1,5	20	1271,725714	317,93	1579,29	1680	1554	1560	1575

Рисунок Б.6 – Слайд 6

Рисунок Б.7 – Слайд 7

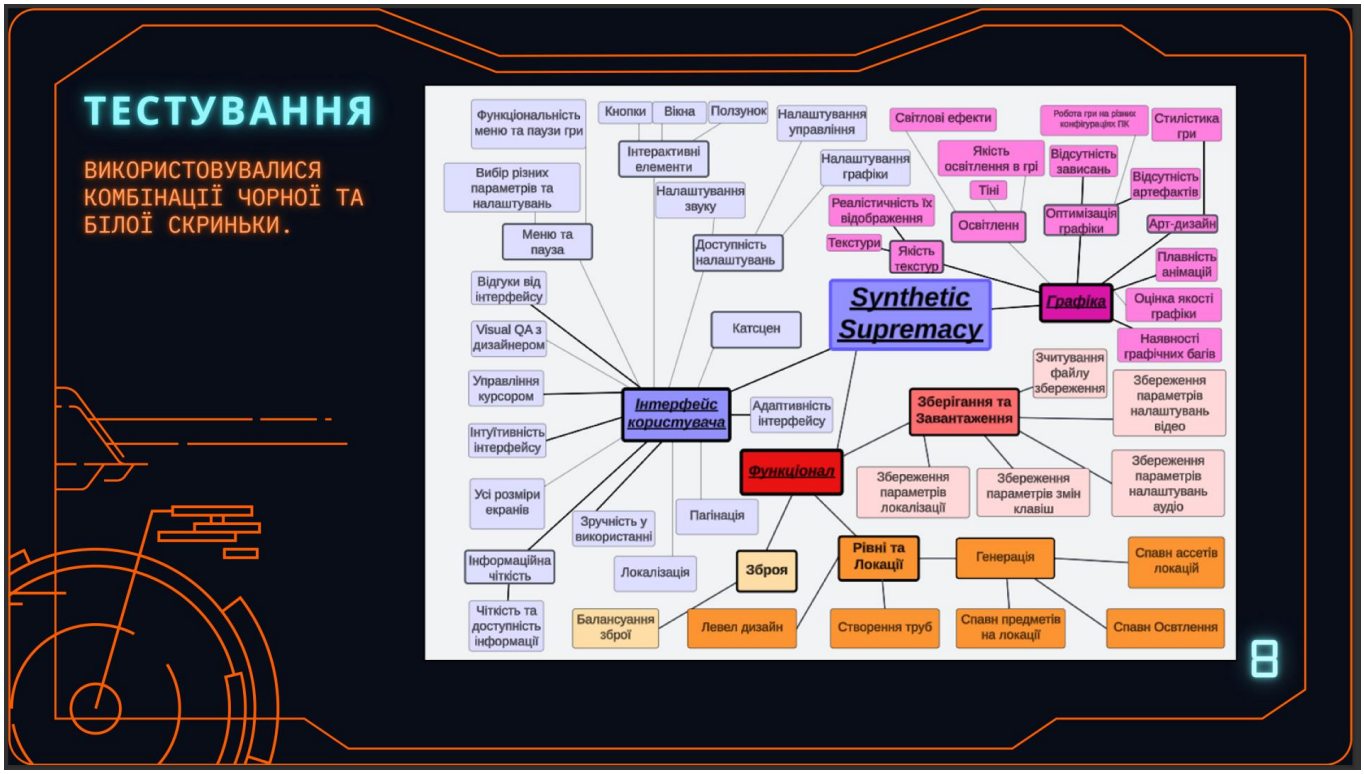


Рисунок Б.8 – Слайд 8

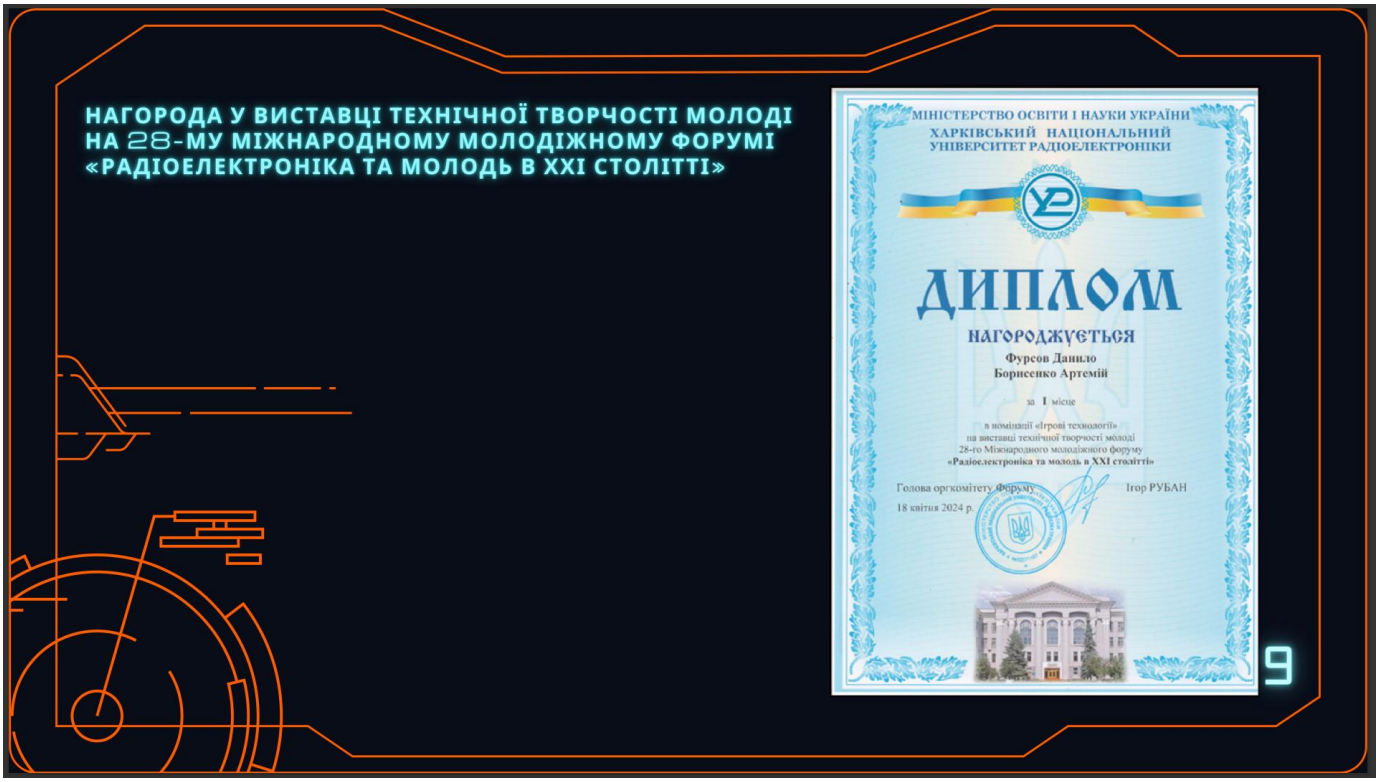


Рисунок Б.9 – Слайд 9

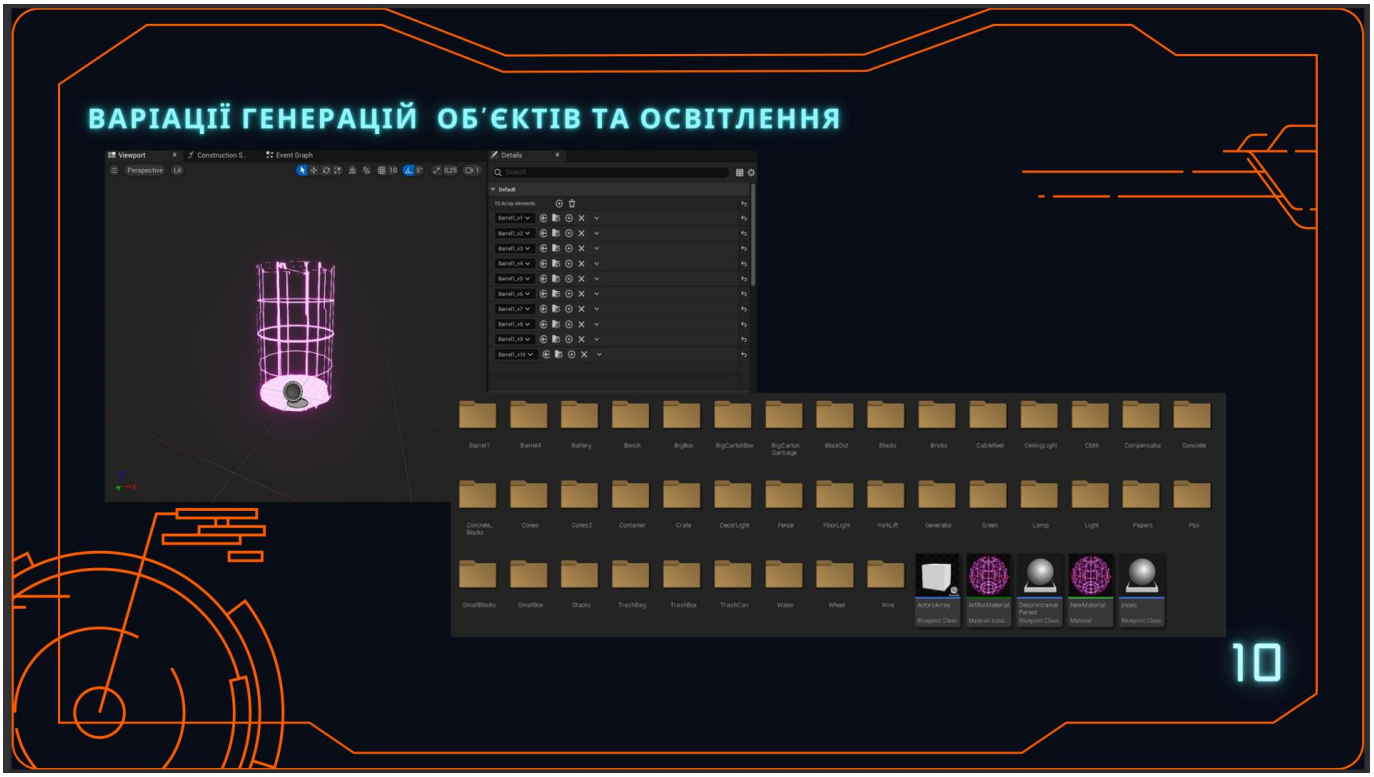


Рисунок Б.10 – Слайд 10

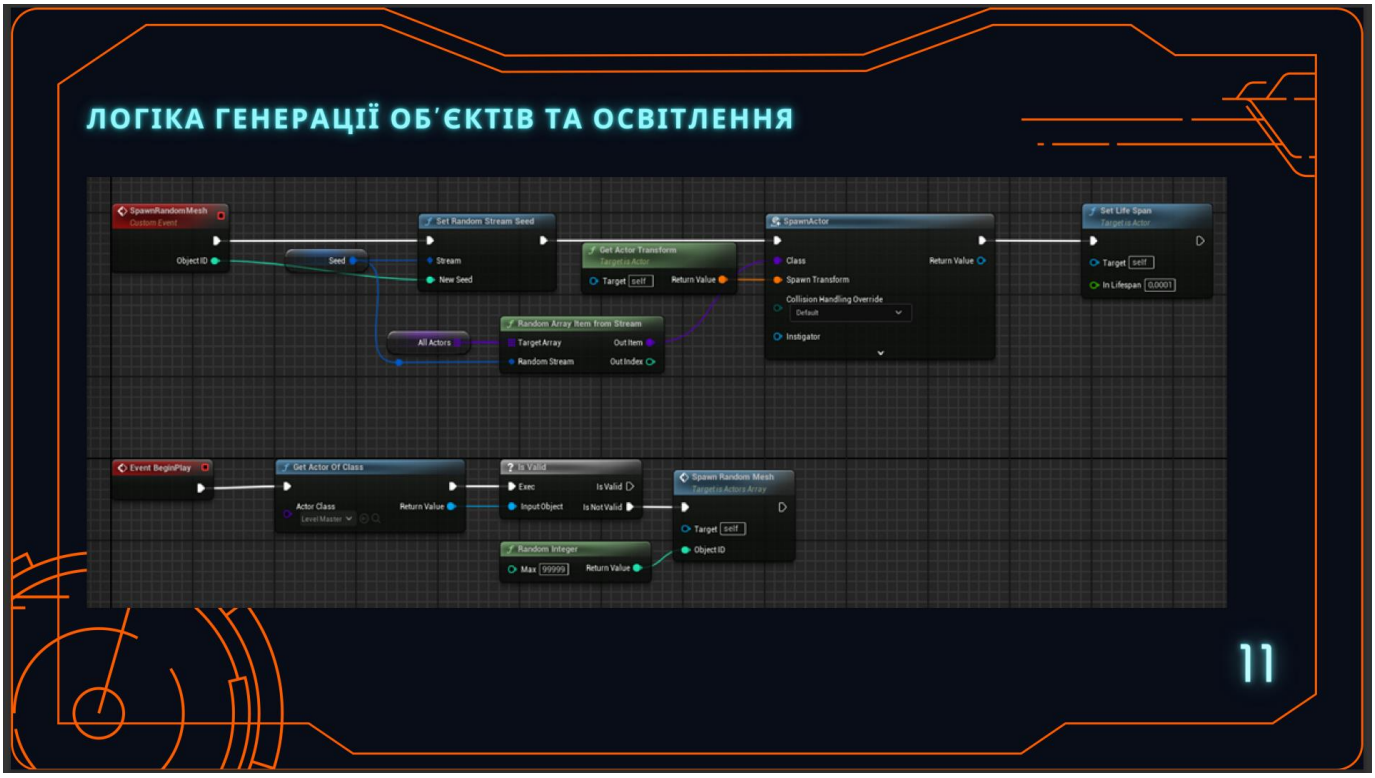


Рисунок Б.11 – Слайд 11

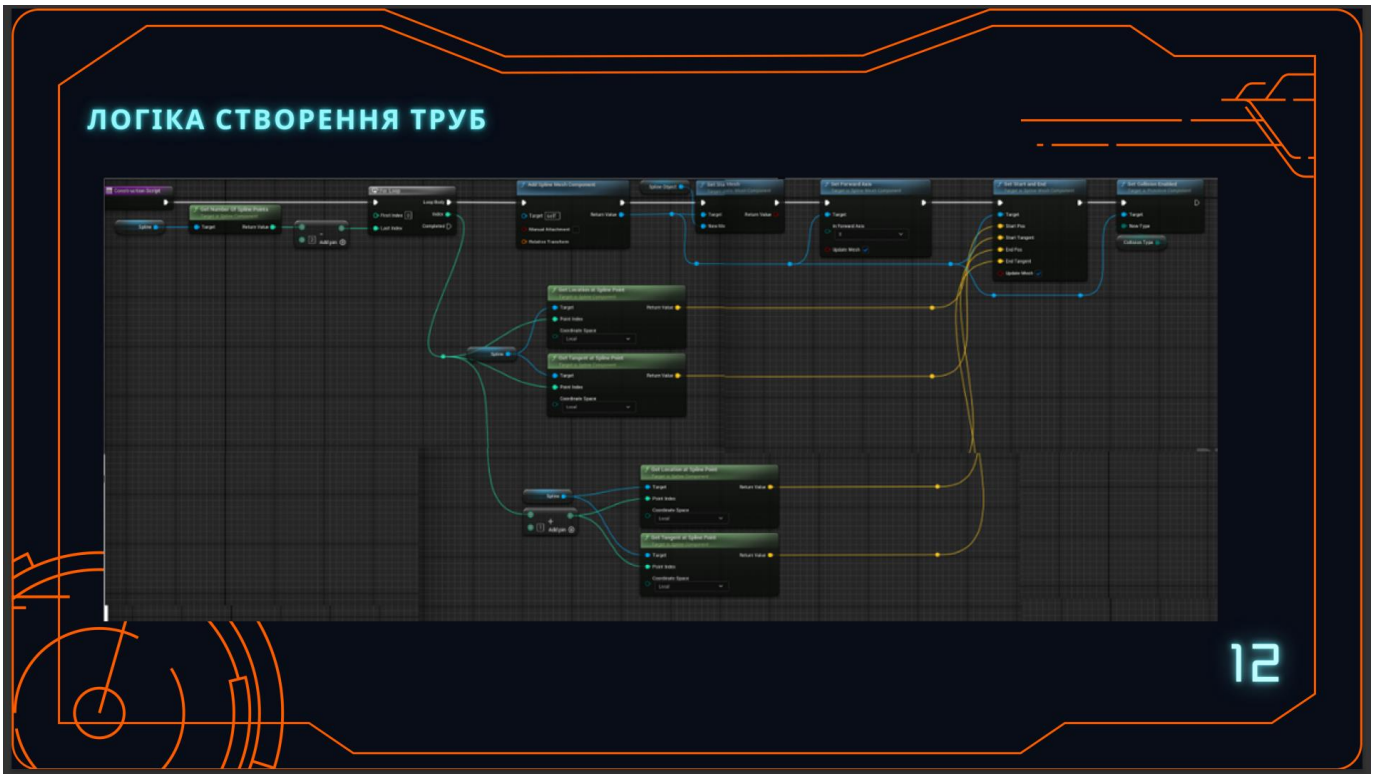


Рисунок Б.12 – Слайд 12

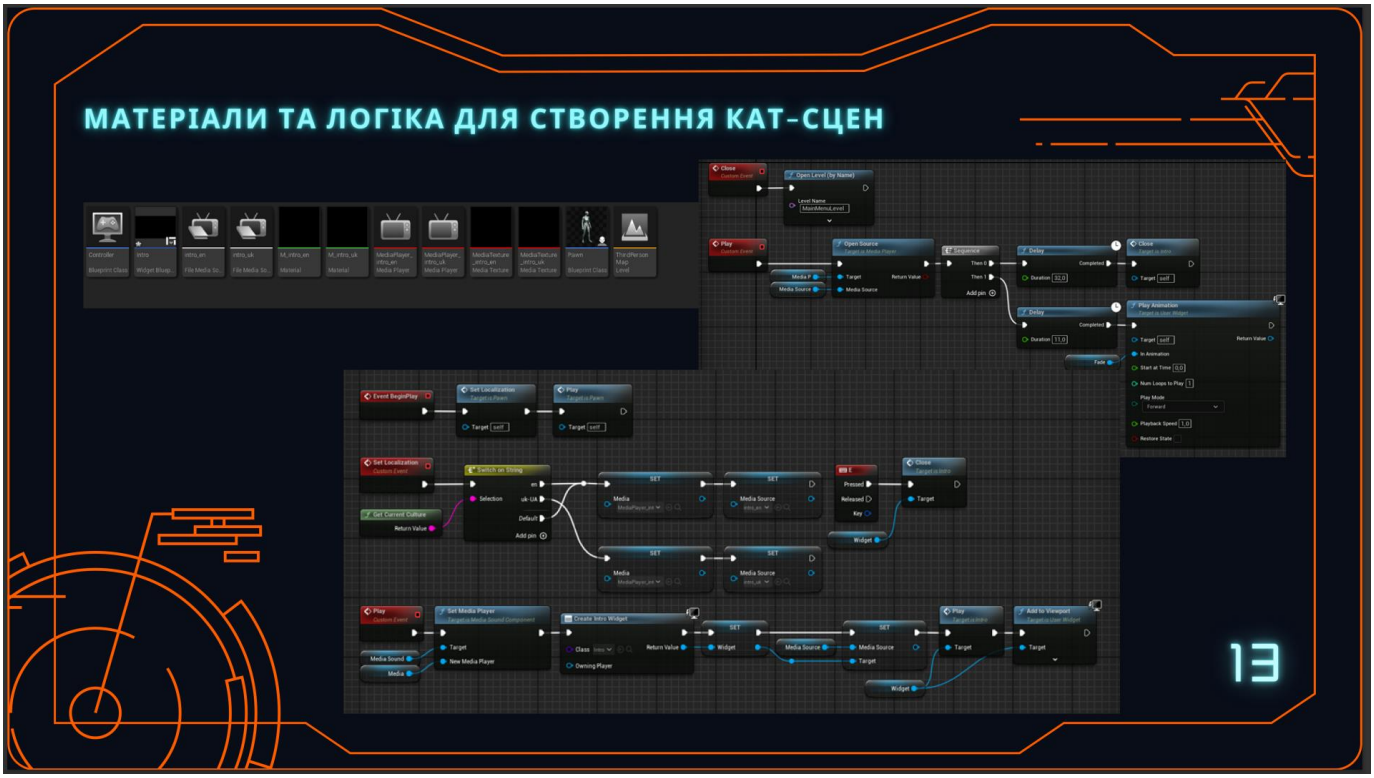


Рисунок Б.13 – Слайд 13

Рисунок Б.14 – Слайд 14

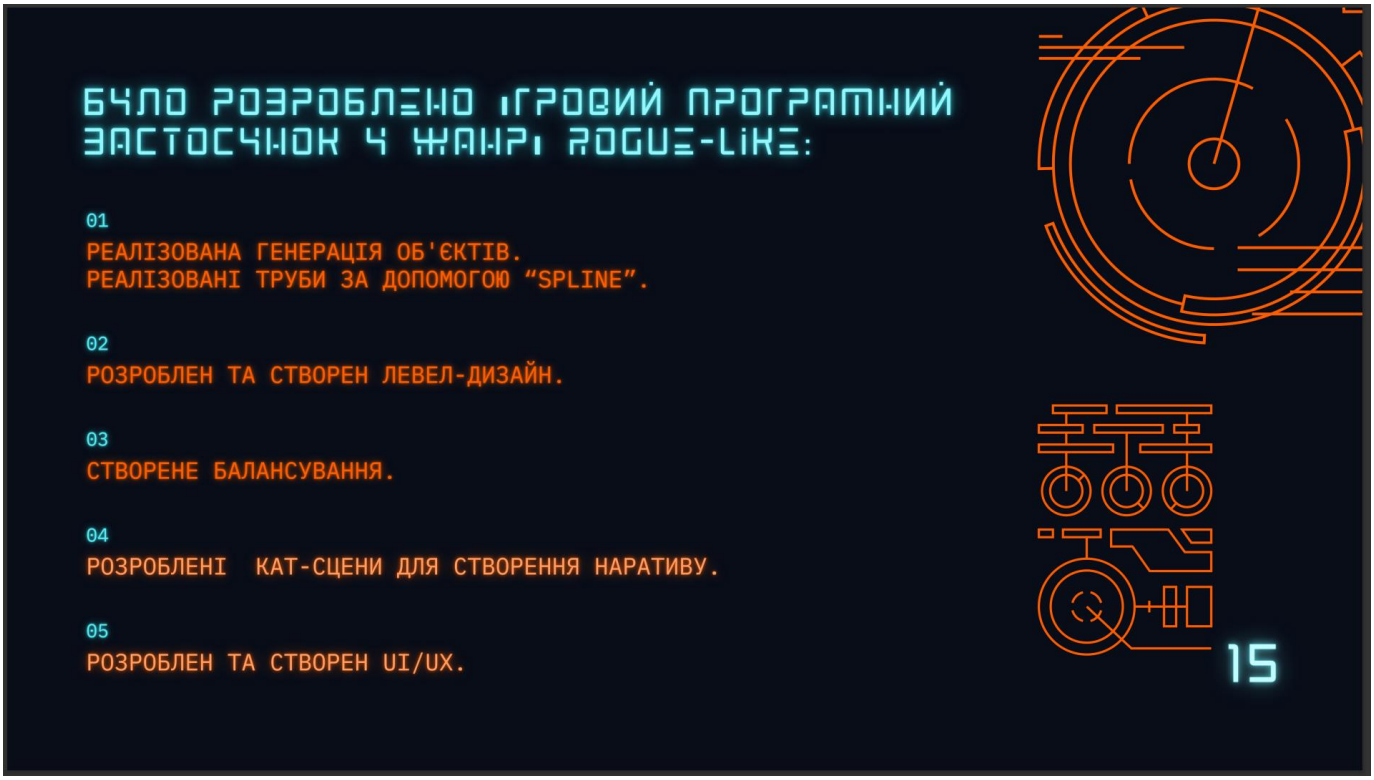


Рисунок Б.15 – Слайд 15

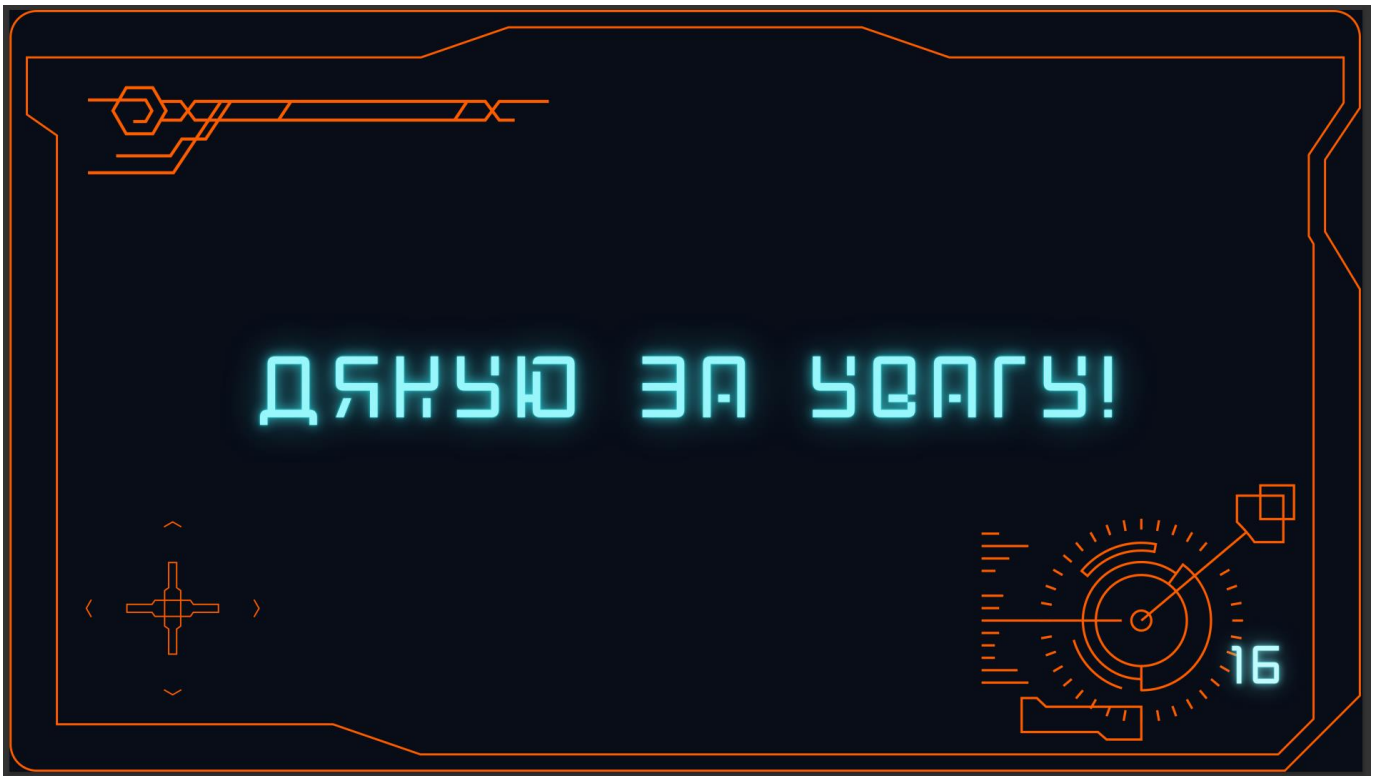


Рисунок Б.16 – Слайд 16

ДОДАТОК В

Геймдизайн-документ

«Synthetic Supremacy»

Зміст:

Рисунок В.1 – Геймдизайн-документ сторінка 1

1) Тетра

Механіка: Roguelike/Шутер

Технологія: Персональний комп’ютер

Історія: ГГ - бойовий робот

Естетика: Кіберпанк

2) Вступ

Дія гри розгортається в майбутньому, в гігантському мегаполісі, що дистопійно перетворився на символ безжальної корпоративної влади та беззаконня, де корпорації та мафія утримують усі путі влади в своїх руках. У цьому майбутньому світі головним героєм є бойовий робот, створений для виконання завдань корпораціями, але загадкові обставини привели до того, що він стає символом опору та визволення від цього соціального кошмару.

Мегаполіс, де розгортається подія гри, - це гігантське місто, вкрите хмарами смогу та сірих будівель. Величезні скляні вежі корпорацій відносяться високо над горизонтом, пануючи над масами мешканців міста. Вулиці вкриті плакатами, пропагандуючи корпоративні ідеали та маніпулюючи свідомістю громадян. Правоохоронні органи в службі корпорацій забезпечують "спокій" у місті, підтримуючи тиранію та беззаконня.

Головний герой, бойовий робот, колись був ідеальним виконавцем різноманітних завдань для корпорацій, але після зіткнення з подіями, які розкрили йому жахливу істину про корпоративний режим, він обирає інший шлях. Звільнившись від корпоративного програмування, він стає символом опору і починає боротьбу з корпораціями та мафією, ставлячи на карту не тільки свою власну долю, але й майбутнє цього знущаного міста.

Рисунок В.2 – Геймдизайн-документ сторінка 2

Гра пропонує гравцям зануритися у цей темний, технологічний світ. Відкрийте для себе різноманітні можливості бойового робота, вступайте в соціальні конфлікти та розкривайте масштабні заговори корпорацій. Граючи за цього воїна, ви станете головним символом боротьби за свободу і справедливість у цьому майбутньому світі.

3) Цільова аудиторія

Основна цільова аудиторія гри - це гравці віком 16+. Дану аудиторію має залучати реалізм, наявність насильства та необхідність швидко приймати рішення. Гра має вікове обмеження 16+ через наявність насильства та необхідність стратегічного та тверезого мислення під час боїв.

4) USP

1. «Звільни місто від корпоративного пекла та стань символом опору в майбутньому гігаполісі!»
2. «Заглибся в дистопічний світ, граючи за бойового робота, вибери свій шлях до визволення.»
3. «Вступай в боротьбу за майбутнє великого мегаполісу і переписуй історію власними руками.»
4. «Час прийшов для тебе стати водночас рятівником і правосуддям!»
5. «Змінюй світ, вибираючи свій шлях у мегаполісі майбутнього. Твої рішення формують долю цього міста.»

Рисунок В.3 – Геймдизайн-документ сторінка 3

5) Час ігрової сесії

Ігрова сесія розрахована на 20–60 хвилин, залежить від навичків гравця та випадкової генерації. Гра є реіграбельною, тому повне проходження гри не передбачено.

6) Технологічні характеристики

Платформа: персональний комп'ютер.

Керування: клавіатура та миш.

Гравець повинен мати середній або хороший комп'ютер, мишку, клавіатуру та монітор.

7) Опис гри

7.1) Опис ігрового процесу

Головне меню

Після запуску гри гравець потрапляє до головного меню, де він може обрати персонажа з унікальною зброєю та айтемами в розділі «CHARACTER». Далі він може вибрати розділ «CLASSIC» для першого запуску гри.

Якщо гравець вже грав і має досвід проходження гри, то замість гри спочатку, він може продовжити з конкретного місця, обравши розділ «CONTINUE», і продовжити проходження гри з того моменту, де він раніше завершив.

Проходження гри

Після початку гри генерується локація та персонаж з'являється в початковій кімнаті. Навколо нього знаходяться сусідні кімнати, він йде в

Рисунок В.4 – Геймдизайн-документ сторінка 4

одну з цих кімнат та починає бій. Використовуючи зброю та предмети гравець перемагає ворогів. Далі він йде до інших кімнат.

В процесі проходження гравець знаходить нову зброю та предмети, які покращують його характеристики та змінюють геймплей, роблячи його унікальним для кожного проходження. Перемагаючи ворогів, персонаж отримує гроші, які він може витратити на покупку ресурсів, таких як здоров'я та патрони, або він має можливість купити нову зброю та предмети.

Проходячи усі кімнати, гравець зустрічає боса. Подолавши боса, гравець переходить на наступний рівень. Кожен наступний рівень більший та складніший за попередній. Якщо гравець пройшов 5 рівнів або його персонаж помер, гра завершується та гроші, зароблені під час проходження, переносяться до головного меню. В головному меню гравець може купити нових персонажів та почати гру повторно.

7.2) Механіки

Персонаж

Перед початком гри гравець обирає персонажа в головному меню. Кожен персонаж має зброю, один або декілька предметів та певні характеристики:

- **Здоров'я:** шкала - що вказує на те, скільки шкоди персонаж може витримати. Якщо дана шкала впаде до 0, то персонаж вмирає та гра закінчується.
- **Щит:** додаткова шкала здоров'я. Кожен раз, коли персонаж отримує шкоду, дана шкала зменшується першою, і лише якщо впаде до 0, то шкода буде діяти на здоров'я. Якщо персонаж не буде отримувати шкоду певний час, то щит почте відновлюватись.

Рисунок В.5 – Геймдизайн-документ сторінка 5

- **Захист:** параметр, який впливає на кількість шкоди, яку отримує персонаж.
- **Бонус до шкоди:** параметр, який впливає на шкоду, яку персонаж гравця завдає ворогам. Даний бонус може бути як для всієї зброї загалом, так і для певних типів зброї.
- **Додаткові бонуси до зброї:** дані параметри не відображаються при виборі персонажу, але присутні і мають вплив. Це бонус до кількості патронів в магазині, бонус до швидкості стрільби та бонус до швидкості перезарядки.
- **Швидкість:** параметр, який вказує на швидкість пересування гравця. Він поділяється на швидкість ходьби, спринту, ходьби крадькома та висоту стрибку.
- **Бонус до випадаючих предметів:** параметр, який збільшує кількість здоров'я, патронів або грошей, які можно отримати з випадаючих предметів.
- **Гроші:** валюта, яку гравець може витратити в автоматах, або зберегти та витратити на покупку персонажів в головному меню.

Після вибору персонажу та початку гри, персонаж отримує змогу пересуватися. Персонаж може ходити в різних напрямках, біжати або сповільнюватись, стрибати, пригинатися та взбиратися на перепони.

Рівні

Після того, як гравець почав гру, генерується рівень. Всього має бути 5 рівнів, кожен зі своїм стилем, ворогами та босами. Рівні складаються з кімнат, з'єднаних коридорами. Якщо в кімнаті є вороги, то гравець не може покинути кімнату поки не переможе їх.

Рисунок В.6 – Геймдизайн-документ сторінка 6

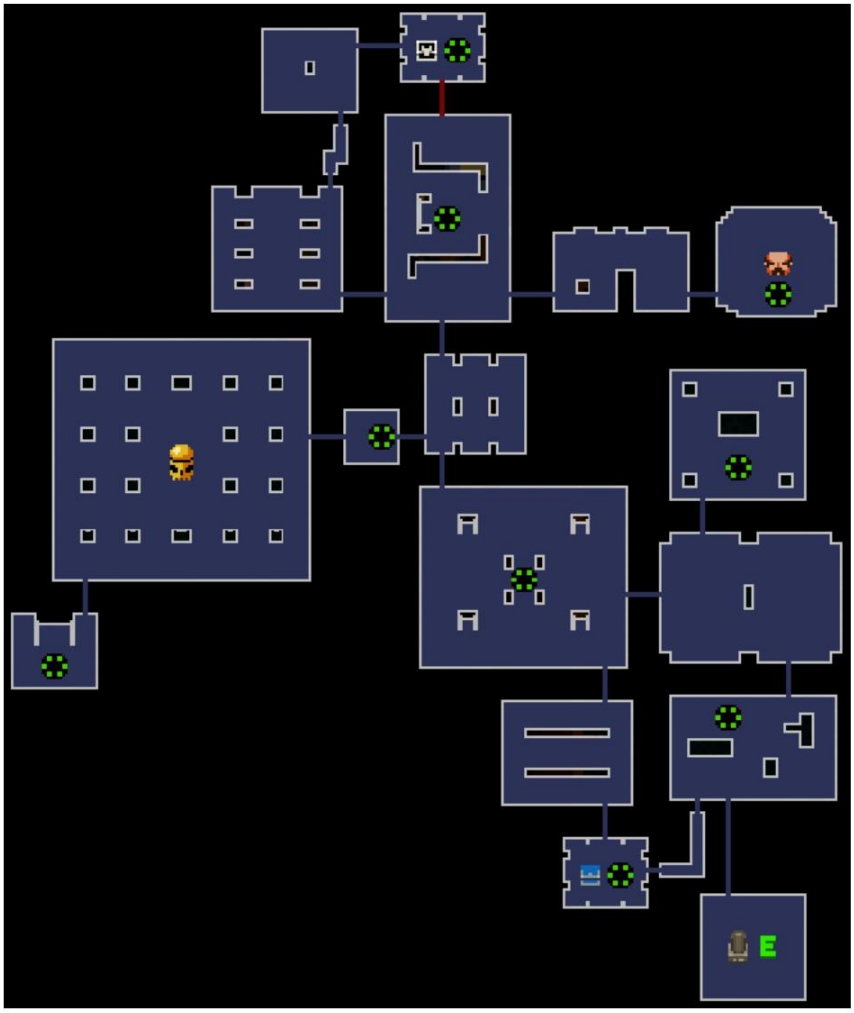


Рисунок 1 - Приклад структури рівнів

Усі кімнати поділяються на декілька типів:

- **Початкова кімната:** кімната, в якій з'являється персонаж на початку гри.

Рисунок В.7 – Геймдизайн-документ сторінка 7

- **Кімната з боєм:** кімната, в якій знаходяться вороги.
- **Кімната з предметом:** кімната, в якій гравець може знайти якийсь випадковий предмет або зброю.
- **Безпечна кімната:** кімната, в якій гравець може знайти автомати та витратити в них гроші.
- **Кімната з босом:** фінальна кімната, в якій гравець може битися з босом. Після проходження даної кімнати гравець отримує можливість пройти на наступний рівень.

Бій

Бій в кімнатах відбувається таким чином:

1. Персонаж заходить в кімнату, в якій він ще не був, двері зачиняються та з'являться вороги.
2. Гравець, використовуючи зброю та предмети, перемагає всіх ворогів у кімнатів. Перша хвиля завершена.
3. Починається друга хвиля, в кімнаті з'являються інші вороги.
4. Гравець знову перемагає їх. Проходить 1-3 хвилі.
5. Битва завершується.

Бій з босом відбувається схожим чином. Відмінністю є те, що хвиль немає та бос зачасту один у кімнаті.

Вороги з'являються групами від 4 до 6 одиниць. Вони, як і персонаж гравця, є роботами та мають певну зброю. Під час бою вони ховаються за укриттями, пересуваються по локації та атакують ворога. Вони не знають точне положення персонажу гравця, а лише місце, де він востаннє був помічений. Вороги можуть ділитися інформацією про гравця та робити рішення на основі дій одне одного. Якщо вони не бачили гравця деякий час, то частина з них піде його шукати.

Рисунок В.8 – Геймдизайн-документ сторінка 8

Після смерті з ворогів випадають певні предмети - ресурси. Шанс їх випадіння залежить від певних характеристик персонажу гравця. Такими предметами є:

- **Аптечка:** предмет, який відновлює певну кількість здоров'я персонажу гравця. Шанс випадіння залежить від кількості здоров'я гравця, тобто чим менше здоров'я, тим частіше випадає цей предмет.
- **Патрони:** предмет, який відновлює певну кількість патронів у зброї, яку персонаж тримає в руках. Даний предмет надає гравцю приблизно один магазин. Шанс випадіння залежить від загальної кількості патронів усієї зброї гравця, тобто чим менше патронів має гравець, тим частіше випадає цей предмет.
- **Гроші:** предмет, який надає гравцю певну кількість грошей.

Зброя

Персонаж може одночасно мати 2 одиниці зброї. На початку гри персонаж має лише одну зброю. В процесі гри є можливість знайти іншу зброю, підібрати її та, якщо персонаж вже має 2 зброї, замінити. Кожна зброя є унікальною та має такі параметри:

- **Магазин:** зброя може мати певний розмір магазину, які можна вистрілити до перезарядки. Також є певна кількість патронів, які можна мати одночасно. Деяка зброя перезаряджається цілим магазином, а деяка по одній кулі.
- **Шкода:** вказує на шкоду, яку завдає дана зброя.
- **Точність:** вказує на розкид під час стрільби.
- **Дальність:** вказує на відстань, на якій шкода завдається в повному розмірі.

Рисунок В.9 – Геймдизайн-документ сторінка 9

- **Швидкість стрільби:** вказує на швидкість стрільби даною зброєю.
- **Зручність:** вказує на розмір віддачі та вагу зброї (сповільнення персонажа).
- **Тип:** зброя може мати певний тип. Є такі типи: легка та важка вогнепальна зброя, легка та важка лазерна зброя, незвичайна зброя.

Також, окрім даних параметрів, зброя може мати й інші відмінності, які не вказано в описі:

- автоматичний режим вогню;
- дробовик;
- снайперський приціл (дозволяє використати приціл для приближення, але забороняє стрільбу без прицілювання);
- сповільнена шкода (наприклад отруєння або підпалення);
- пробиваючий постріл (крізь стіни та ворогів);
- безшумний постріл;
- заряд (треба затиснути кнопку пострілу на певний час та відпустити, шкода залежить від часу затискання);
- взривний постріл.

Предмети

Предмети є двох типів: активні та пасивні. На початку гри персонаж може мати один активний предмет та декілька пасивних. В процесі гри є можливість знайти інші предмети.

Активні предмети виконують певну дію лише після активації гравцем, мають час дії та затримку перед повторним використання. Вони можуть як підвищувати певні характеристики персонажа, так і виконувати певні дії. Дані предмети мають заряд, після використання предмета заряд

Рисунок В.10 – Геймдизайн-документ сторінка 10

обнуляється. В деяких предметів заряд може відновлюватися по таймеру, в інших за проходження кімнат. Предмет можна повторно використати лише після того, як заряд заповниться. Персонаж гравця може мати одночасно лише один активний предмет.

Пасивні предмети виконують певну дію одразу після підняття та мають постійний ефект. Персонаж може взяти необмежену кількість пасивних предметів.

Гроші

В процесі гри гравець отримує ресурс - гроші. Гроші використовуються при взаємодії з автоматами, які знаходяться в безпечних кімнатах. Автомати поділяються на такі типи:

- **Автомат з аптечками:** дозволяє відновити здоров'я персонажа за гроші.
- **Автомати з патронами:** дозволяє відновити патрони в зброї, яку тримає персонаж, за гроші.
- **Автомати зі зброєю:** дозволяє купити за гроші ту зброю, яка зображена на цьому автоматі.
- **Автомат з предметами:** дозволяє купити за гроші той предмет, який зображений на цьому автоматі.
- **Банкомат:** дозволяє перенести гроші з даного проходження в головне меню без втрат.

Якщо персонаж гравця вмирає, то мала частина грошей переноситься до головного меню, де у гравця є можливість витратити їх на покупку персонажів. Якщо персонаж проходить всі рівні повність, то до головного меню переноситься 100% грошей (так само, як і з банкоматом).

Рисунок В.11 – Геймдизайн-документ сторінка 11

7.3) Інтерфейс

Інтерфейс головного меню

Спочатку гравець бачить інтерфейс головного меню, звідки він може:

1. Натиснути «START» і розпочати класичний режим гри, також він може натиснути «CLASSIC» і теж почне класичний режим гри.
2. Натиснути «CHARACTER» і перейти до характеристик персонажа та його зброї і айтемів.
3. Натиснути шестерню, та потрапити в розділ за налаштувань.
4. Натиснути «EXIT» і вийти з гри.
5. Натиснути «CONTINUE» і продовжити минулу компанію.

Рисунок 2 - Інтерфейс головного меню

Інтерфейс характеристик персонажа, зброї та айтемів

Гравець бачить інтерфейс характеристик, звідки він може:

Рисунок В.12 – Геймдизайн-документ сторінка 12

1. Натиснути «BUY» щоб придбати персонажа з індивідуальними характеристиками та айтемами(активними та пасивними).
2. Натиснути «SELECT» і обрати персонажа з індивідуальними характеристиками та айтемами(активними та пасивними).
3. Натиснути стрілки для свайпу і роздивляння персонажів та їх індивідуальні характеристики та айтемами(активні та пасивні).

Рисунок 3 - Інтерфейс характеристик персонажа, зброї та айтемів

Інтерфейс паузи

Гравець повинен натиснути «ESC», щоб побачити інтерфейс паузи, звідки він може:

1. Натиснути «RESUM» щоб продовжити гру.
2. Натиснути «SETTINGS» щоб перейти до розділу налаштувань.
3. Натиснути «EXIT TO MAIN MENU» щоб вийти до головного меню.

Рисунок В.13 – Геймдизайн-документ сторінка 13

4. Натиснути «EXIT TO MAIN MENU» щоб вийти з гри.

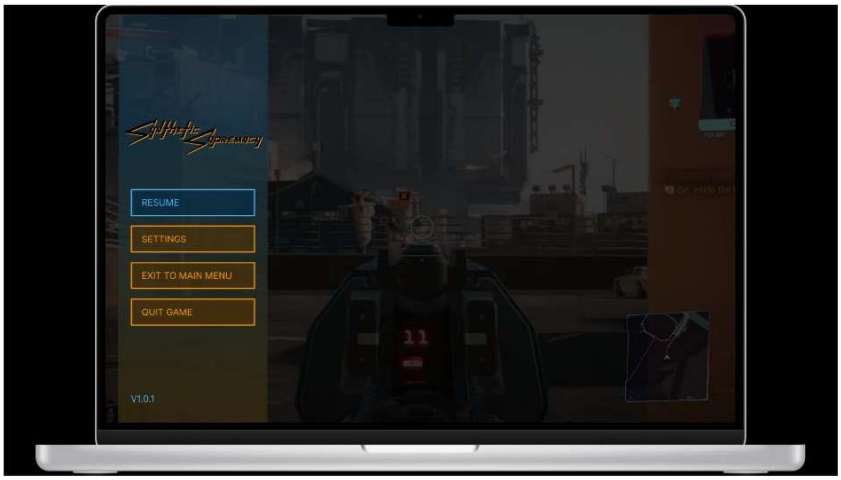


Рисунок 4 - Інтерфейс паузи

Інтерфейс налаштувань

Гравець бачить інтерфейс налаштувань, звідки він може:

1. Натиснути «AUDIO» щоб перейти до розділу налаштувань звука.
2. Натиснути «VIDEO» щоб перейти до розділу налаштувань відео.
3. Натиснути «LOCALIZATION» щоб перейти до розділу обрання мови.
4. Натиснути «CONTROLS» щоб перейти до розділу керування.

Рисунок В.14 – Геймдизайн-документ сторінка 14

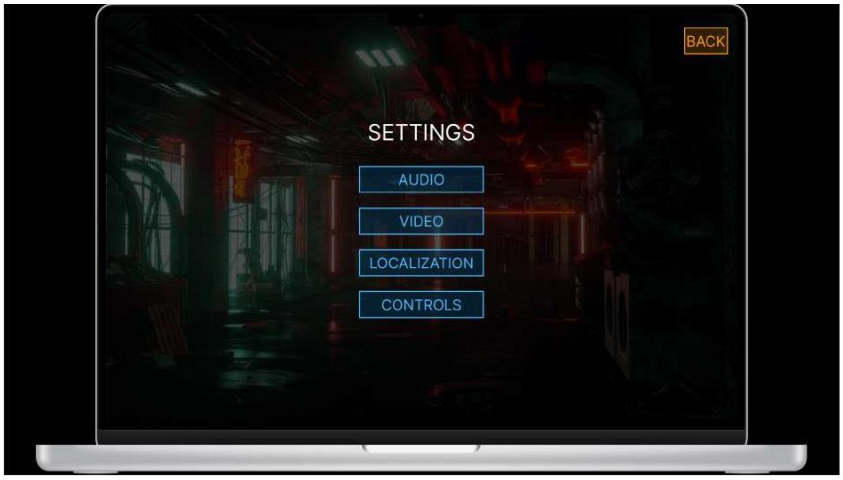


Рисунок 5 - Інтерфейс налаштувань

Інтерфейс завантаження гри

Гравець бачить інтерфейс завантаження гри:

1. Натиснути будь яку клавішу щоб почати гру .

Рисунок В.15 – Геймдизайн-документ сторінка 15

Інтерфейс запуск гри

Гравець бачить інтерфейс зупуска гри:

1. Натиснути будь яку клавішу щоб запустити гру.

Рисунок 7 - Інтерфейс запуску гри

Рисунок В.16 – Геймдизайн-документ сторінка 16

Рисунок 10 - Робо-місто майбутнього

Рисунок В.17 – Геймдизайн-документ сторінка 18

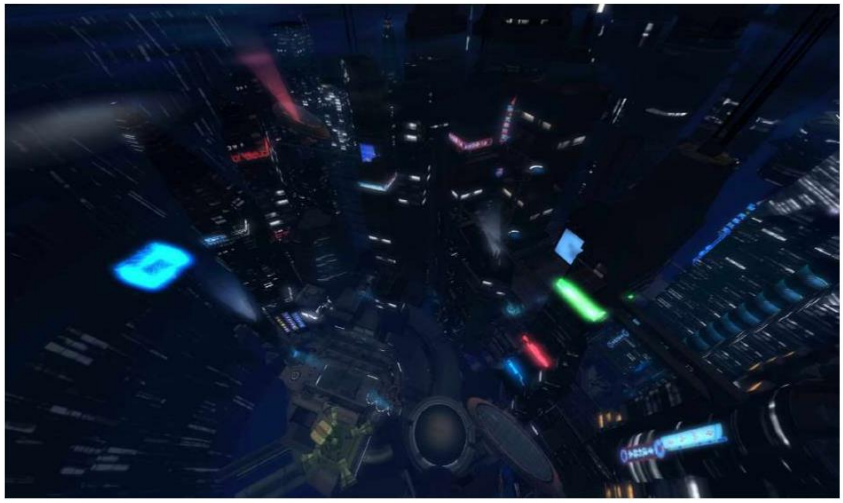


Рисунок 8 - Робо-місто майбутнього

Рисунок 9 - Робо-місто майбутнього

Рисунок В.18 – Геймдизайн-документ сторінка 17

Рисунки 11-14 - Концепти роботів

Рисунок В.19 – Геймдизайн-документ сторінка 19

Зброя

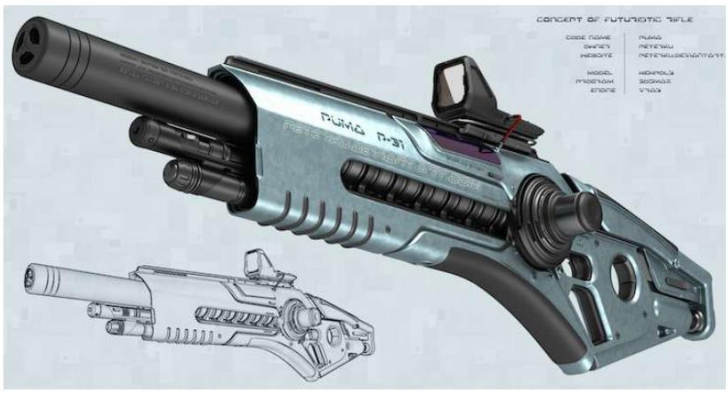


Рисунок 15 - Концепт зброї

Рисунок 16 - Концепт зброї

Рисунок В.20 – Геймдизайн-документ сторінка 20

Рисунок 17 - Концепт зброї

Автомати

Рисунок В.21 – Геймдизайн-документ сторінка 21

Рисунок 19 - Автомат для поповнення здоров’я

Рисунок В.22 – Геймдизайн-документ сторінка 23

Рисунок 18 - Автомат для купівлі боєприпасів

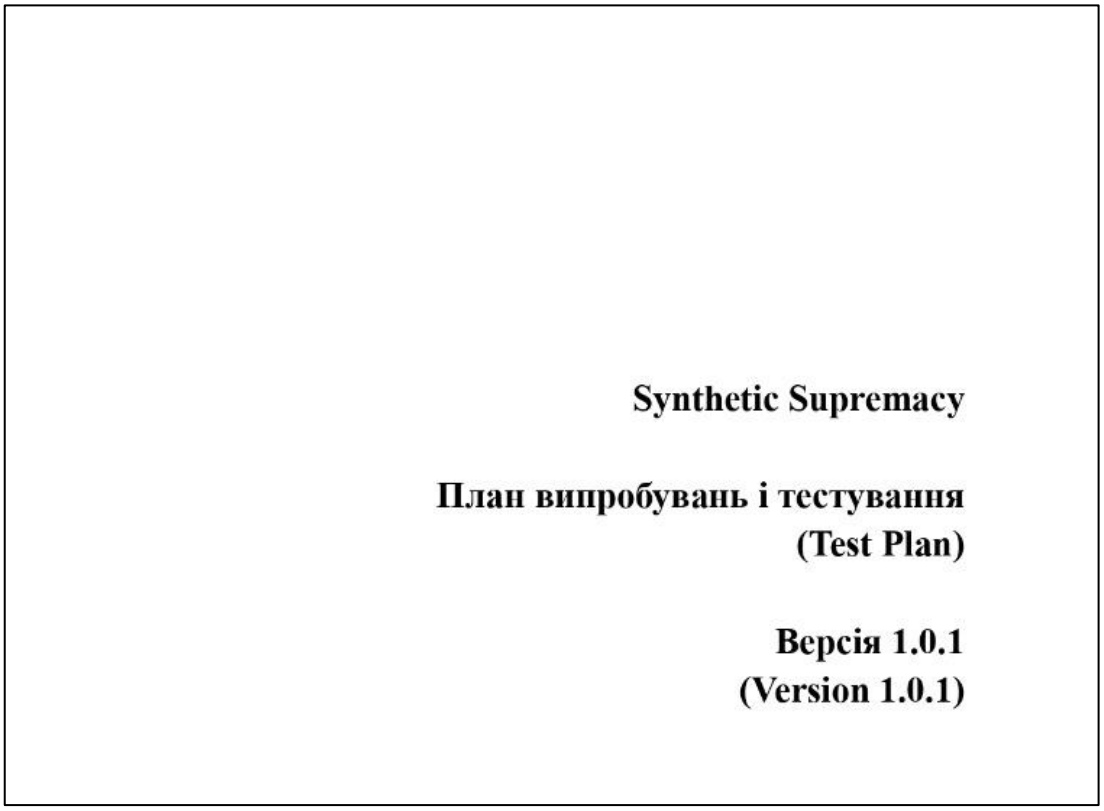
Рисунок В.23 – Геймдизайн-документ сторінка 22

Рисунок 20 - Автомат для придбання зброї та айтемів

Рисунок В.24 – Геймдизайн-документ сторінка 24

ДОДАТОК Г

Тест-план

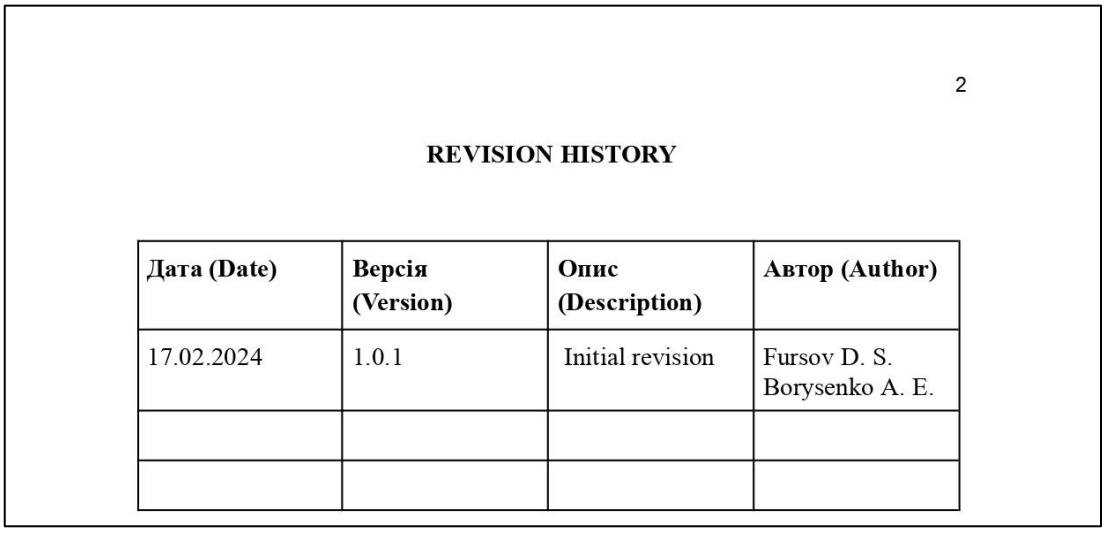


Synthetic Supremacy

План випробувань і тестування
(Test Plan)

Версія 1.0.1
(Version 1.0.1)

Рисунок Г.1 – Тест-план сторінка 1



2

REVISION HISTORY

Дата (Date)	**Версія (Version)**	**Опис (Description)**	**Автор (Author)**
17.02.2024	1.0.1	Initial revision	Fursov D. S. Borysenko A. E.

Рисунок Г.2 – Тест-план сторінка 2

ЗМІСТ

Рисунок Г.3 – Тест-план сторінка 3

1. Вступ (Introduction)

1.1 Мета (Purpose)

Метою складання даного тест плану є опис процесу тестування гри «Synthetic Supremacy». Мета документу - координація роботи процесу розробки у сфері контролю якості продукту. Документ призначений групі тестування для ознайомлення з характером майбутніх робіт, аналізу і розбиття на підзадачі. Документ дозволяє отримати уявлення про заходи з тестування проекту.

1.2 Довідкова інформація (Background)

Гра відбувається з виглядом від третьої особи. Дія гри розгортається в майбутньому, в гігантському мегаполісі, що дистопійно перетворився на символ безжальної корпоративної влади та беззаконня, де корпорації та мафія утримують усі путі влади в своїх руках. У цьому майбутньому світі головним героєм є бойовий робот, створений для виконання завдань корпораціями, але загадкові обставини привели до того, що він стає символом опору та визволення від цього соціального кошмару.

Мегаполіс, де розгортається подія гри, - це гігантське місто, вкрите хмарами смогу та сірих будівель. Величезні скляні вежі корпорацій відносяться високо над горизонтом, пануючи над масами мешканців міста. Вулиці вкриті плакатами, пропагандуючи корпоративні ідеали та маніпулюючи свідомістю громадян. Правоохоронні органи в службі корпорацій забезпечують "спокій" у місті, підтримуючи тиранію та беззаконня.

Головний герой, бойовий робот, колись був ідеальним виконавцем різноманітних завдань для корпорацій, але після зіткнення з подіями, які розкрили йому жахливу істину про корпоративний режим, він обирає інший шлях. Звільнившись від корпоративного програмування, він стає символом опору і починає боротьбу з корпораціями та мафією, ставлячи на карту не тільки свою власну долю, але й майбутнє цього знущаного міста.

Гра пропонує гравцям зануритися у цей темний, технологічний світ. Відкрийте для себе різноманітні можливості бойового робота, вступайте в

Рисунок Г.4 – Тест-план сторінка 4

соціальні конфлікти та розкривайте масштабні заговори корпорацій. Граючи за цього воїна, ви станете головним символом боротьби за свободу і справедливість у цьому майбутньому світі.

1.3 Галузь застосування (Scope)

Метою тестування гри «Cyberpunk 2077» є перевірка коректної роботи її функціоналу та зручності для користувача.

Підсумком процесу тестування повинен стати розгорнутий огляд, що дає розробникам, менеджерам і користувачам даного продукту картину якості ігрового процесу та юзабіліті.

Тестування системи відбувається з використанням підходу «білої скриньки».

Гра перевіряється на платформі Windows 10.

Будуть перевірені наступні компоненти системи:

- Інтерфейс;
- Коректність роботи функціоналу гри.

Найважливішими показниками ефективності для перевірки є:

- Взаємодію користувача з клієнтом гри;
- Коректність роботи функціоналу гри.

1.4 Визначення проекту (Project Identification)

У таблиці 1.1 наведено документацію та її готовність, для розробки плану тестування.

Рисунок Г.5 – Тест-план сторінка 5

Таблиця 1.1 - Документація

Документ і версія / дата	Створено або доступно	Отримано або Перевірено	Автор або ресурс	Примітка
Концепція гри (Game Conception)	Так	Так	Fursov D. S. Borysenko A. E.	
Дизайн-документ гри (Game Design Document)	Так	Так	Fursov D. S. Borysenko A. E.	
Референс-документ гри (Game References Document)	Ні	Ні	Fursov D. S. Borysenko A. E.	
План проекту (Project Plan)	Ні	Ні	Fursov D. S. Borysenko A. E.	

2. Вимоги до тестування (Requirements for Test)

2.1 Функціональні вимоги

У наведеному нижче переліку виявлено ті елементи (випадки використання, функціональні вимоги, нефункціональні вимоги), які були визначені як цілі для тестування. Цей список представляє те, що буде перевірено.

Перелік функцій системи, які будуть тестуватися:

- Тестування обчислень;
- Тестування графіки гри;
- Тестування функціональності;
- Тестування інтерфейсу;

Рисунок Г.6 – Тест-план сторінка 6

- Тестування локалізації;
- Тестування поведінки ШІ;
- Тестування генерації світу.

3. Типи тестування. Стратегія тестування (Test Strategy)

На системі будуть проводитися наступні види тестування:

- Функціональне тестування;
- Тестування інтерфейсу користувача;
- Тестування продуктивності.

На системі не будуть проводитися наступні види тестування:

- Бізнес-цикл тестування;
- Стресове тестування;
- Тестування інсталяції.
- Тестування навантаження;
- Тестування конфігурації;
- Тестування безпеки і контролю доступу.

3.1 Типи тестування (Testing Types)

3.1.1 Функціональне тестування (Function Testing)

Функціональне тестування забезпечує відповідність вимогам грою. Цей тип тестування стосується результатів обробки. Він імітує фактичне використання системи(гри), але не робить жодних припущень про структуру системи. Він базується на вимогах гравця. Нижче наведено план тестування (див. табл. 3.1).

Рисунок Г.7 – Тест-план сторінка 7

Таблиця 3.1 – Функціональне тестування

Мета випробування	Забезпечити належне тестування функціональності, в тому числі проходження рівнів, перевірка механік бойової системи та економіки
Технічний прийом	Виконання кожного use-case або тест кейсів, щоб виявити: - Очікувані результати виникають при очікуваних діях; - Результати виконання розрахунків співпадають з результатами формул.
Критерії завершення	Основний функціонал (інтерфейс гри, ігровий процес).
Спеціальні рекомендації	Головний функціонал інтерфейсу потрібно тестувати на різних екранах.

3.1.2 Тестування інтерфейсу користувача (User Interface Testing)

Тестування інтерфейсу користувача — це процес тестування продукту інтерфейсу користувача для забезпечення його відповідності до специфікації (див. табл. 3.2).

Таблиця 3.2 – Тестування інтерфейсу користувача

Мета випробування	Завданням тестування графічного інтерфейсу користувача є виявлення помилок наступного характеру: - Помилки у функціональності за допомогою інтерфейсу; - Необроблені виключення при взаємодії з інтерфейсом; - Втрата або перекручення даних, переданих через елементи інтерфейсу; - Помилки в інтерфейсі (невідповідність проектної документації, відсутність елементів інтерфейсу).
Технічний прийом	Перевірка коректності відображення даних та елементів інтерфейсу. Тестування з різними розмірами екрану.

Рисунок Г.8 – Тест-план сторінка 8

9

Критерії завершення	На екрані відображаються усі елементи, працюють усі кнопки.
Спеціальні рекомендації	Тестувати систему з різними розмірами екрану.

3.1.3 Тестування продуктивності (Performance Profiling)

Тестування продуктивності — це процес тестування продукту, який полягає у вимірюванні стабільності та швидкості програмного забезпечення під час користування гравцем (див. табл. 3.3).

Таблиця 3.3 – Тестування продуктивності

Мета випробування	Перевірка продуктивності для призначених операцій при дотриманні наступних умов: - нормальний очікуваний обсяг; - очікується гірший випадок навантаження.
Технічний прийом	- Використання випробувань і тестування функцій. - Зміна файлів даних зі збільшенням числа збережених об’єктів на рівні гри, а також предметів в інвентарі персонажа. - Сценарії мають бути запущені на одній машині і повторюватися з декількома клієнтами.
Критерії завершення	Успішне завершення тестових скріптів без збоїв і в межах очікування або в межах виділеного часу на транзакцію.
Спеціальні рекомендації	Тестування має проводитися на виділеному пристрої або в означений час – це дозволяє повністю контролювати і точніше проводити вимірювання.

Рисунок Г.9 – Тест-план сторінка 9

3.2 Інструменти (Tools)

Таблиця 3.4 - Інструменти

Процес	Інструмент
Створення тест кейсів	Гугл форма
Трекінг багів	Гугл таблиця
Виконання тест кейсів	Вручну
Структура проекту	Mind Map

4. Ресурси (Resources)

4.1 Ролі (Roles)

Таблиця 4.1 показує припущення щодо кадрового забезпечення проекту.

Таблиця 4.1 – Припущення кадрового забезпечення проекту

Працівник	Рекомендований мінімальний обсяг осіб	Конкретні обов’язки або коментарі
Тест-менеджер, менеджер з тестування	2	Забезпечує управління наглядом. Обов’язки: - технічна підтримка; - придбання відповідних ресурсів; - забезпечення управлінської звітності.
Проектувальник тестів	2	Визначення, пріоритетів, і реалізація тестів. Обов'язки: - створення плану тестування,

Рисунок Г.10 – Тест-план сторінка 10

		- генерація тестових моделей, - оцінка ефективності тестових зусиль
Тестувальник	2	Виконання тестів. Обов'язки: - виконання тестів, - журнал результатів, - відновлення в журналі реєстрації після помилок, - документ зміни.
Тестовий системний адміністратор	2	Забезпечує тестове середовище і управління активами. Обов'язки: - адміністрування тестової системи управління, - встановлення і управління доступом до тест-системи.

4.2 Система (System)

Таблиця 4.2 – Система

Ресурси	Назва / Тип
Клієнт випробувань ПК (Client Test PC's)	CPU: AMD Ryzen 5 5600H 3/30 GHz GPU: NVIDIA GeForce RTX 3060 Laptop RAM: 16 gb SSD: 21 gb
Репозиторій тестування (Test Repository)	-

5. Етапи проекту (Project Milestones)

Рисунок Г.11 – Тест-план сторінка 11

Тестування повинно включати тестові заходи для кожного випробування, визначеного в попередніх розділах. Необхідно визначити окремі етапи проекту, щоб повідомити про досягнення статусу проекту.

Таблиця 5.1 – Етапи проекту

Цільове завдання	Обсяг робіт	Дата початку	Дата закінчення
План випробувань	5 год.	18.02.2024	18.02.2024
Тест – дизайн	12 год.	19.02.2024	21.02.2024
Реалізація випробувань	100 год.	22.02.2024	27.02.2024

6. Кінцевий продукт (Deliverables)

6.1 Тестова модель

В ході виконання тестування будуть отримані такі елементи:

- план тестування – опис цілей та стратегій тестування, методів реалізації процесу тестування;
- тест-кейси – документи, що відповідають примірникам тестового сценарію. Мають містити: унікальний номер, опис, кроки відтворення, пріоритет, важливість тестового сценарію та очікуваний результат;
- тестовий журнал – запис всіх дій пов'язаних з проведенням тестування, послідовності необхідних операцій та результатів;
- баг-репорти – звіти про знайдені недоліки у системі з зазначенням рівню серйозності проблеми.

6.2 Тестовий журнал

Рисунок Г.12 – Тест-план сторінка 12

Під час тестування цієї системи для кожного виду випробувань будуть створюватися звіти про виявлені дефекти та тестові випадки. Ведення тестового журналу буде відбуватися з використанням текстових редакторів (MS Word, Excel).

6.3 Звіти з дефектів

Звіти з дефектів будуть створені з використанням текстових редакторів (MS Word, Excel).

Рисунок Г.13 – Тест-план сторінка 13

ДОДАТОК А

Задачі проекту (Appendix A Project Tasks)

План тестування:

- визначення вимог тестування;
- оцінка ризику;
- розробка стратегії тестування;
- визначення тест-ресурсів;
- створення графіку;
- створення плану тестування.

Дизайн випробувань:

- підготовка аналізу робочого навантаження;
- визначення та опис тестів;
- визначення та структура тестових процедур;
- огляд та оцінка тестового покриття.

Впровадження випробувань:

- запис або сценарії програмних випробувань;
- визначення тестових функцій у розробці та запровадження моделі;
- встановлення зовнішніх наборів даних.

Виконання тесту:

- виконання процедури випробувань;
- оцінка виконання випробувань;
- перевірка результатів;
- дослідження несподіваних результатів, журнал дефектів.

Рисунок Г.14 – Тест-план сторінка 14

15

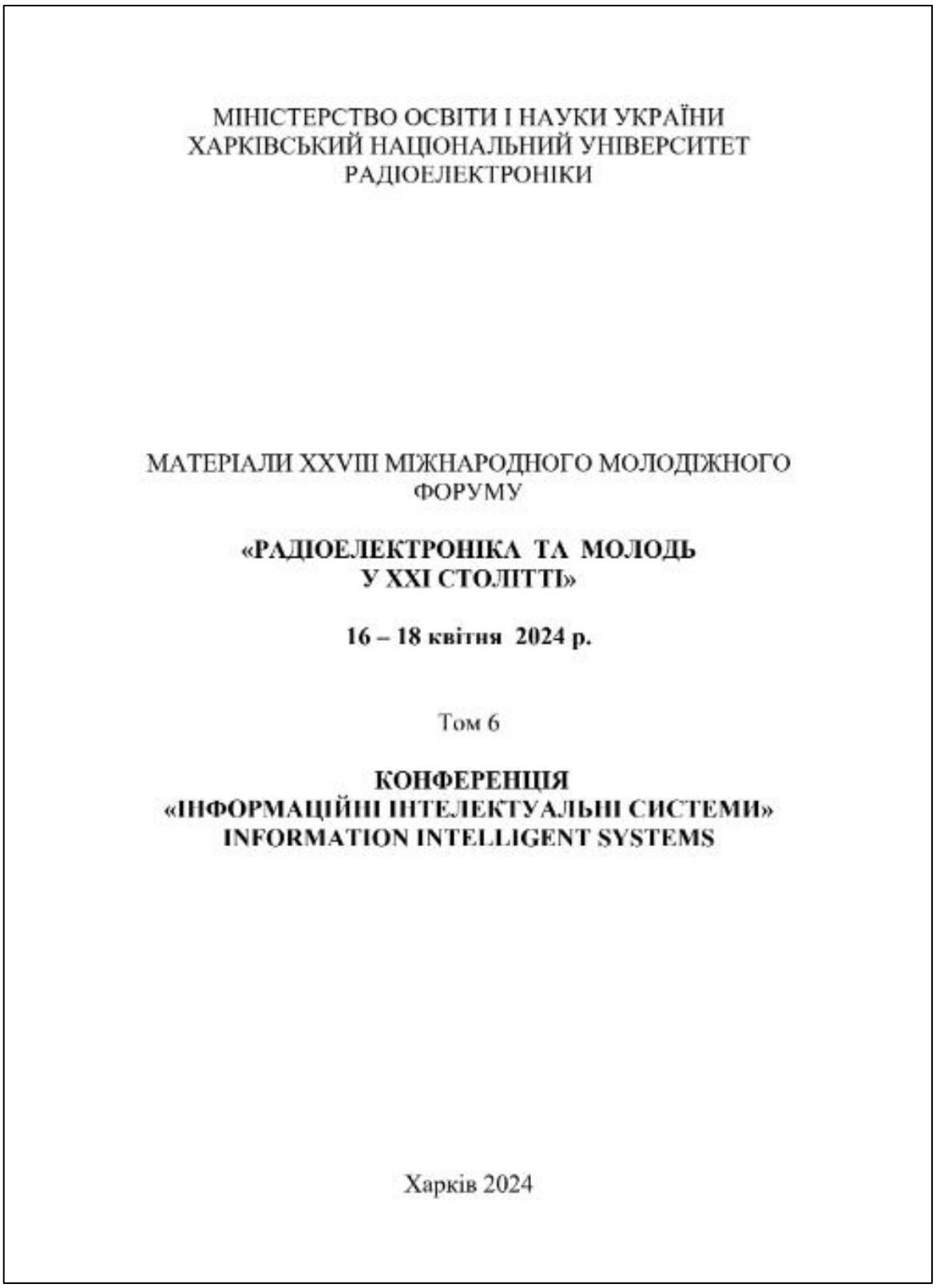
Оцінка випробувань:

- оцінку випробувань разі покриття;
- оцінки покриття коду;
- аналіз дефектів;
- визначення критеріїв завершення випробувань і критеріїв успіху, що були досягнуті.

Рисунок Г.15 – Тест-план сторінка 15

ДОДАТОК Ґ

Тези доповіді для науково-практичної інтернет-конференції


МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
РАДІОЕЛЕКТРОНІКИ

МАТЕРІАЛИ XXVIII МІЖНАРОДНОГО МОЛОДІЖНОГО ФОРУМУ

«РАДІОЕЛЕКТРОНІКА ТА МОЛОДЬ У XXI СТОЛІТТІ»

16 – 18 квітня 2024 р.

Том 6

КОНФЕРЕНЦІЯ
«ІНФОРМАЦІЙНІ ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ»
INFORMATION INTELLIGENT SYSTEMS

Харків 2024

Рисунок Ґ.1 – Обкладинка збірника

АЛФАВІТНИЙ ПОКАЖЧИК

Рисунок Г.2 –Алфавітний покажчик збірника

УДК 004.051

ОПТИМІЗАЦІЯ ОСВІТЛЕННЯ В UNREAL ENGINE 5

Борисенко А. Е.
Науковий керівник – ст. викл. Новіков Ю. С.
Харківський національний університет радіоелектроніки, каф. ПІ,
м. Харків, Україна
e-mail: artemii.borysenko@nure.ua

This work explores the application of Virtual Shadow Maps (VSM) and Lumen Global Illumination and Reflections (Lumen GI) to improve the visual quality and realism of virtual environments in Unreal Engine 5. The work describes how to improve the level of detail and lighting in a game environment in Unreal Engine 5. It also provides a visual comparison of Virtual Shadow Maps (VSM) and Shadow Mapping in Unreal Engine 5. The reproduction of clear, realistic shadows using Virtual Shadow Maps (VSM) and Lumen Global Illumination and Reflections (Lumen GI) is demonstrated.

Майже всі ігри Unreal Engine 5 зараз використовують лише програмну версію Lumen, яка не використовує апаратне прискорення за допомогою сучасних відеокарт.

Проблема з трасуванням променів у комп'ютерних іграх полягає в тому, що воно дуже важке для гри, яка його використовує, особливо враховуючи, що до отримання кращих результатів, ніж коли його вимкнено. Lumen – це нова, повністю динамічна глобальна система освітлення та дзеркального відображення в Unreal Engine 5, і саме вона намагається «виправити» цю проблему.

Однією з ключових особливостей Lumen є можливість у реальному часі розраховувати освітлення й тіні на поверхнях, навіть у разі зміни освітлення в грі. Це дозволяє досягти більш реалістичних ефектів освітлення та заощадити час.

Lumen забезпечує пряме та непряме світло, тобто розраховується відбиття променів від поверхонь з позиції гравця. Під час імітації шляху враховуються й матеріали, що віддзеркалюють зображення. Таким чином технологія дозволяє реалізовувати відображення.

Важлива особливість технології – це кластеризація тривимірного оточення для поділу на менші сегменти. Завдяки цьому збільшується ефективність глобального освітлення, яке окремо розраховується для кожної частини. В результаті зростає загальна продуктивність.

З одного боку технологія Lumen – це простота в налаштуванні та використанні, вона прораховує освітлення менш точно, і не для кожного променя, а одразу для пучка променів, гарантуючи реалістичне освітлення та тіні, дозволяючи реалізовувати дзеркала. Але з іншого боку, рейтрейсинг накладає вимоги до ПК, причому як для роботи, так і для запуску фінального продукту та не поєднується зі звичайними методами

Рисунок Ґ.3 – Матеріал тез сторінка 1

реалізації статичного освітлення. Крім цього Lumen розроблено для консолей нового покоління й комп'ютерів високого класу. На рисунках 1-2 показано наявність та відсутність Lumen.

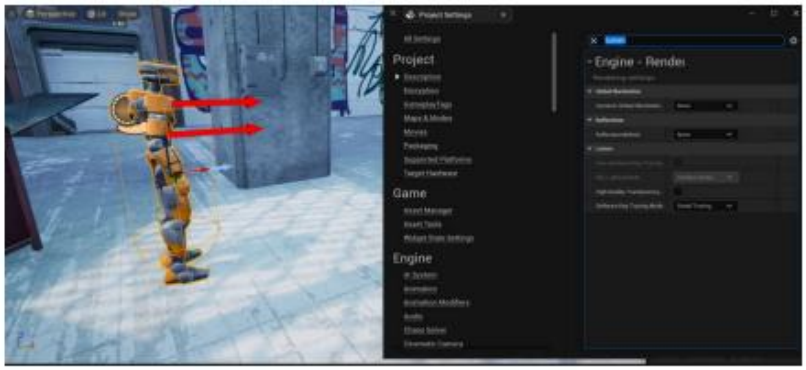


Рисунок 1 – Вимкнено Lumen

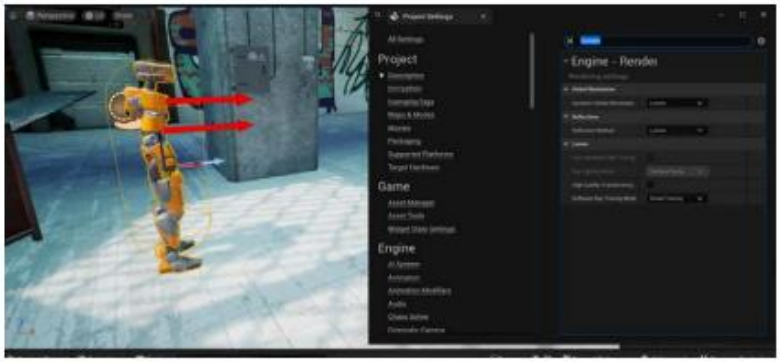


Рисунок 2 – Увімкнено Lumen

Virtual Shadow Maps (VSM) це – нова методика мапування тіней для їхньої послідовної реалізації з високою роздільною здатністю. Вона працює з дуже деталізованими асетами кінематографічної якості та великими відкритими просторами. позиціонується, як єдиний уніфікований шлях створення тіней в Unreal Engine 5, який може замінити решту способів. Ця технологія йде опліч з Nanite і Lumen.

VSM З одного боку має вищу якість та кращу стабільність з меншою витратою часу на обробку проти Shadow Maps, але з іншого боку ця методика створена як доповнення до віртуалізованої геометрії, тому в плані якості та продуктивності дуже залежить від правильного застосування Nanite. На рисунках 3-4 показана різниця між Shadow Mapping та Visual Shadow Mapping.

Рисунок Ґ.4 – Матеріал тез сторінка 2

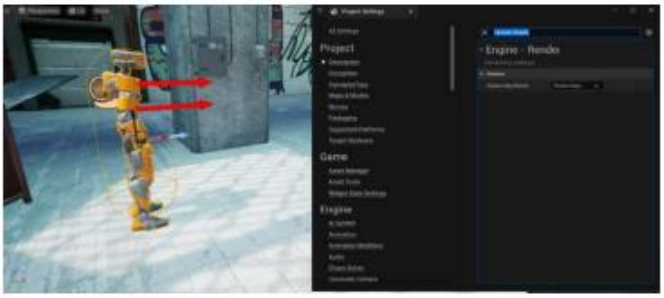


Рисунок 3 – Використання Shadow Mapping

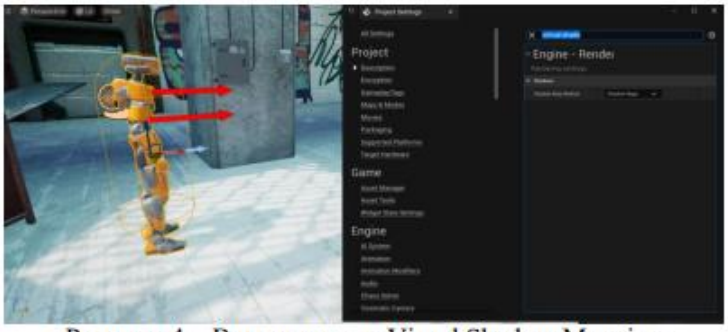


Рисунок 4 – Використання Visual Shadow Mapping

Хоча Lumen та VSM прості у використанні та здаються високоякісними, Lumen може значно навантажувати систему. Тому рекомендується комбінувати Lumen з іншими, більш традиційними методами освітлення.

Список використаних джерел:

1. Огляд технологій Unreal Engine 5 з розробниками: застосування, переваги та перспективи. URL: https://gamedev.dou.ua/articles/unreal-engine-technologies-review/ (дата звернення: 25.03.2024).

2. Радіонов О. М., Новіков Ю.С. Використання карти нормалей у динамічному освітленні. Матеріали XXVII міжнародного молодіжного форуму «Радіоелектроніка та молодь у XXI столітті», Харків, 2023.

3. Козлов Д.О., Новіков Ю.С. Використання градієнта ambient освітлення для імітації зміни пори доби в 2d просторі платформи unity. Science progress in European countries: new concepts and modern solutions Харків, 2019.

Рисунок Ґ.5 – Матеріал тез сторінка 3

ДОДАТОК Д

Тези доповіді для науково-практичної інтернет-виставки

Ігровий програмний застосунок у жанрі 3D Rogue-like шутер від третьої особи «Synthetic Supremacy»

Автори: ***Фурсов Данило Сергійович***, ПЗПІ-20-7, ХНУРЕ,
Борисенко Артемій Едуардович, ПЗПІ-20-7, ХНУРЕ.
Науковий керівник: Новіков Юрій Сергійович, старший викладач кафедри ПІ, ХНУРЕ.

Розроблено ігровий програмний застосунок у жанрі 3D Rogue-like шутер від третьої особи під назвою «Synthetic Supremacy». Дана гра розроблена на ігровому рушії Unreal Engine 5 з використанням технологій оптимізації графіки Nanite та освітлення Lumen. За сюжетом гри гравець виступає в ролі бойового робота, головною метою якого є пройти всі рівні, подолавши всіх ворогів на шляху. Події гри відбуваються в футуристичному мегаполісі, ігровий світ виконаний у стилістиці кіберпанку. Рівні складаються з кімнат та коридорів, які генеруються автоматично під час кожного проходження. Основними функціями гри є битва з ворогами, знаходження предметів і зброї та покращення персонажу гравця.

Напрям та секція: «Ігрові технології».

Рисунок Д.1 – Опис роботи для презентації

ДОДАТОК Е

Приклад програмного коду(клас ActorsArray)

```
// Random stream functions
//
	/** Returns a uniformly distributed random number between 0 and Max - 1 */
	UFUNCTION(BlueprintPure, Category="Math|Random", meta=(ScriptMethod="RandomInt",
ScriptMethodMutable))
	static ENGINE_API int32 RandomIntegerFromStream(const FRandomStream& Stream, int32
Max);

	UE_DEPRECATED(5.2, "Use RandomIntegerFromStream taking the FRandomStream as the first
argument.")
	static int32 RandomIntegerFromStream(int32 Max, const FRandomStream& Stream)
	{
		return RandomIntegerFromStream(Stream, Max);
	}

	/** Return a random integer between Min and Max (>= Min and <= Max) */
	UFUNCTION(BlueprintPure, Category="Math|Random", meta=(ScriptMethod="RandomIntInRange",
ScriptMethodMutable))
	static ENGINE_API int32 RandomIntegerInRangeFromStream(const FRandomStream& Stream,
int32 Min, int32 Max);

	UE_DEPRECATED(5.2, "Use RandomIntegerInRangeFromStream taking the FRandomStream as the
first argument.")
	static int32 RandomIntegerInRangeFromStream(int32 Min, int32 Max, const FRandomStream&
Stream)
	{
		return RandomIntegerInRangeFromStream(Stream, Min, Max);
	}

	/** Returns a random bool */
	UFUNCTION(BlueprintPure, Category="Math|Random", meta=(ScriptMethod="RandomBool",
ScriptMethodMutable))
	static ENGINE_API bool RandomBoolFromStream(const FRandomStream& Stream);

	/** Returns a random float between 0 and 1 */
	UFUNCTION(BlueprintPure, Category="Math|Random", meta=(ScriptMethod="RandomFloat",
ScriptMethodMutable))
	static ENGINE_API float RandomFloatFromStream(const FRandomStream& Stream);
```

```
	/** Generate a random number between Min and Max */
	UFUNCTION(BlueprintPure, Category="Math|Random",
meta=(ScriptMethod="RandomFloatInRange", ScriptMethodMutable))
	static ENGINE_API float RandomFloatInRangeFromStream(const FRandomStream& Stream, float
Min, float Max);

	UE_DEPRECATED(5.2, "Use RandomFloatInRangeFromStream taking the FRandomStream as the
first argument.")
	static float RandomFloatInRangeFromStream(float Min, float Max, const FRandomStream&
Stream)
	{
		return RandomFloatInRangeFromStream(Stream, Min, Max);
	}

	/** Returns a random vector with length of 1.0 */
	UFUNCTION(BlueprintPure, Category="Math|Random", meta=(ScriptMethod="RandomUnitVector",
ScriptMethodMutable))
	static ENGINE_API FVector RandomUnitVectorFromStream(const FRandomStream& Stream);

	/** Returns a random point within the specified bounding box using the first vector as
an origin and the second as the half size of the AABB. */
	UFUNCTION(BlueprintPure, Category="Math|Random",
meta=(ScriptMethod="RandomPointInBoundedBox", ScriptMethodMutable))
	static ENGINE_API FVector RandomPointInBoundingBoxFromStream(const FRandomStream&
Stream, const FVector Center, const FVector HalfSize);

	UE_DEPRECATED(5.2, "Use RandomPointInBoundingBoxFromStream taking the FRandomStream as
the first argument.")
	static FVector RandomPointInBoundingBoxFromStream(const FVector Center, const FVector
HalfSize, const FRandomStream& Stream)
	{
		return RandomPointInBoundingBoxFromStream(Stream, Center, HalfSize);
	}

	/** Returns a random point within the specified bounding box. */
	UFUNCTION(BlueprintPure, Category="Math|Random", meta=(DisplayName="Random Point In
Bounding Box From Stream (Box)"), meta=(ScriptMethod="RandomPointInBox",
ScriptMethodMutable))
	static ENGINE_API FVector RandomPointInBoundingBoxFromStream_Box(const FRandomStream&
Stream, const FBox Box);

	UE_DEPRECATED(5.2, "Use RandomPointInBoundingBoxFromStream_Box taking the FRandomStream
as the first argument.")
	static FVector RandomPointInBoundingBoxFromStream_Box(const FBox Box, const
```

```
FRandomStream& Stream)
	{
		return RandomPointInBoundingBoxFromStream_Box(Stream, Box);
	}

	/** Create a random rotation */
	UFUNCTION(BlueprintPure, Category="Math|Random", meta=(ScriptMethod="RandomRotator",
ScriptMethodMutable))
	static ENGINE_API FRotator RandomRotatorFromStream(const FRandomStream& Stream, bool
bRoll);

	UE_DEPRECATED(5.2, "Use RandomRotatorFromStream taking the FRandomStream as the first
argument.")
	static FRotator RandomRotatorFromStream(bool bRoll, const FRandomStream& Stream)
	{
		return RandomRotatorFromStream(Stream, bRoll);
	}

	/** Reset a random stream */
	UFUNCTION(BlueprintCallable, Category="Math|Random", meta=(ScriptMethod="Reset",
ScriptMethodMutable))
	static ENGINE_API void ResetRandomStream(const FRandomStream& Stream);

	/** Create a new random seed for a random stream */
	UFUNCTION(BlueprintCallable, Category="Math|Random",
meta=(ScriptMethod="GenerateNewSeed"))
	static ENGINE_API void SeedRandomStream(UPARAM(ref) FRandomStream& Stream);

	/** Set the seed of a random stream to a specific number */
	UFUNCTION(BlueprintCallable, Category="Math|Random", meta=(ScriptMethod="SetSeed"))
	static ENGINE_API void SetRandomStreamSeed(UPARAM(ref) FRandomStream& Stream, int32
NewSeed);

	/**
	 * Returns a random vector with length of 1, within the specified cone, with uniform
random distribution.
	 * @param ConeDir                              The base "center" direction of the cone.
	 * @param ConeHalfAngleInRadians The half-angle of the cone (from ConeDir to edge), in
radians.
	 * @param Stream                               The random stream from which to obtain
the vector.
	 */
	UFUNCTION(BlueprintPure, Category="Math|Random", meta = (Keywords = "RandomVector",
ScriptMethod = "RandomUnitVectorInConeInRadians", ScriptMethodMutable))
```

```
	static ENGINE_API FVector RandomUnitVectorInConeInRadiansFromStream(const
FRandomStream& Stream, const FVector& ConeDir, float ConeHalfAngleInRadians);

	UE_DEPRECATED(5.2, "Use RandomUnitVectorInConeInRadiansFromStream taking the
FRandomStream as the first argument.")
	static FVector RandomUnitVectorInConeInRadiansFromStream(const FVector& ConeDir, float
ConeHalfAngleInRadians, const FRandomStream& Stream)
	{
		return RandomUnitVectorInConeInRadiansFromStream(Stream, ConeDir,
ConeHalfAngleInRadians);
	}

	/**
	 * Returns a random vector with length of 1, within the specified cone, with uniform
random distribution.
	 * @param ConeDir                              The base "center" direction of the cone.
	 * @param ConeHalfAngleInDegrees The half-angle of the cone (from ConeDir to edge), in
degrees.
	 * @param Stream                               The random stream from which to obtain
the vector.
	 */
	UFUNCTION(BlueprintPure, Category="Math|Random", meta = (Keywords = "RandomVector",
ScriptMethod = "RandomUnitVectorInConeInDegrees", ScriptMethodMutable))
	static inline FVector RandomUnitVectorInConeInDegreesFromStream(const FRandomStream&
Stream, const FVector& ConeDir, float ConeHalfAngleInDegrees)
	{
		return RandomUnitVectorInConeInRadiansFromStream(Stream, ConeDir,
FMath::DegreesToRadians(ConeHalfAngleInDegrees));
	}

	UE_DEPRECATED(5.2, "Use RandomUnitVectorInConeInDegreesFromStream taking the
FRandomStream as the first argument.")
	static inline FVector RandomUnitVectorInConeInDegreesFromStream(const FVector& ConeDir,
float ConeHalfAngleInDegrees, const FRandomStream& Stream)
	{
		return RandomUnitVectorInConeInDegreesFromStream(Stream, ConeDir,
ConeHalfAngleInDegrees);
	}

	/**
	* Returns a random vector with length of 1, within the specified cone, with uniform
random distribution.
	* The shape of the cone can be modified according to the yaw and pitch angles.
	*
```

```
	* @param MaxYawInRadians  The yaw angle of the cone (from ConeDir to horizontal edge), in radians.
	* @param MaxPitchInRadians The pitch angle of the cone (from ConeDir to vertical edge), in radians.
	* @param Stream                         The random stream from which to obtain the vector.
	*/
	UFUNCTION(BlueprintPure, Category = "Math|Random", meta = (Keywords = "RandomVector", ScriptMethod = "RandomUnitVectorInEllipticalConeInRadians", ScriptMethodMutable))
	static ENGINE_API FVector RandomUnitVectorInEllipticalConeInRadiansFromStream(const FRandomStream& Stream, const FVector& ConeDir, float MaxYawInRadians, float MaxPitchInRadians);

	UE_DEPRECATED(5.2, "Use RandomUnitVectorInEllipticalConeInRadiansFromStream taking the FRandomStream as the first argument.")
	static FVector RandomUnitVectorInEllipticalConeInRadiansFromStream(const FVector& ConeDir, float MaxYawInRadians, float MaxPitchInRadians, const FRandomStream& Stream)
	{
		return RandomUnitVectorInEllipticalConeInRadiansFromStream(Stream, ConeDir, MaxYawInRadians, MaxPitchInRadians);
	}

	/**
	* Returns a random vector with length of 1, within the specified cone, with uniform random distribution.
	* The shape of the cone can be modified according to the yaw and pitch angles.
	*
	* @param MaxYawInDegrees  The yaw angle of the cone (from ConeDir to horizontal edge), in degrees.
	* @param MaxPitchInDegrees The pitch angle of the cone (from ConeDir to vertical edge), in degrees.
	* @param Stream                         The random stream from which to obtain the vector.
	*/
	UFUNCTION(BlueprintPure, Category = "Math|Random", meta = (Keywords = "RandomVector", ScriptMethod = "RandomUnitVectorInEllipticalConeInDegrees", ScriptMethodMutable))
	static inline FVector RandomUnitVectorInEllipticalConeInDegreesFromStream(const FRandomStream& Stream, const FVector& ConeDir, float MaxYawInDegrees, float MaxPitchInDegrees)
	{
		return RandomUnitVectorInEllipticalConeInRadiansFromStream(Stream, ConeDir, FMath::DegreesToRadians(MaxYawInDegrees), FMath::DegreesToRadians(MaxPitchInDegrees));
	}
```

```
	UE_DEPRECATED(5.2, "Use RandomUnitVectorInEllipticalConeInDegreesFromStream taking the FRandomStream as the first argument.")
	static inline FVector RandomUnitVectorInEllipticalConeInDegreesFromStream(const FVector& ConeDir, float MaxYawInDegrees, float MaxPitchInDegrees, const FRandomStream& Stream)
	{
		return RandomUnitVectorInEllipticalConeInDegreesFromStream(Stream, ConeDir, MaxYawInDegrees, MaxPitchInDegrees);
	}

	/**
	 * Generates a 1D Perlin noise from the given value.  Returns a continuous random value between -1.0 and 1.0.
	 *
	 * @param	Value  The input value that Perlin noise will be generated from.  This is usually a steadily incrementing time value.
	 *
	 * @return	Perlin noise in the range of -1.0 to 1.0
	 */
	UFUNCTION(BlueprintPure, Category="Math|Random")
	static ENGINE_API float PerlinNoise1D(const float Value);
```