

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 122 «Комп'ютерні науки»
(освітня програма «Комп'ютерна графіка та розробка ігор»)
на тему:

**«Інтерактивний 3D світ для ігрового застосунку
"Forbidden Forest"»**

Виконав студент групи КН-20-2
Турлій Артур Олександрович

Керівник роботи:
КУЗЬМЕНКО Олександр Вікторович

Рецензент:
ГРАФ Марина Сергіївна

Житомир – 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук
Марина ГРАФ
«25» січня 2024 р.

ЗАВДАННЯ
на кваліфікаційну роботу освітнього ступеня «бакалавр»
за спеціальністю 122 «Комп'ютерні науки»
(освітня програма «Комп'ютерна графіка та розробка ігор»)

Здобувач вищої освіти: **ТУРЛІЙ Артур Олександрович**
Керівник роботи: **КУЗЬМЕНКО Олександр Вікторович**
Тема роботи: **«Інтерактивний 3D світ для ігрового застосунку "Forbidden Forest"»,**
затверджена наказом закладу вищої освіти від **«25» січня 2024 р., №22/с**
Термін здачі закінченої роботи: **«11» червня 2024 р.**
Вихідні дані роботи (зазначається предмет і об'єкт дослідження): використанням рушія
UNITY, а також налаштування Particle Effect та Shader Graph для досягнення потрібних
візуальних ефектів в інтерактивному 3D світі ігрового застосунку "Forbidden Forest".

Консультанти кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Дата	
		Завдання видав	Завдання прийняв
1	Олександр КУЗЬМЕНКО	01.09.2023	01.11.2023
2	Олександр КУЗЬМЕНКО	01.11.2023	01.04.2024
3	Олександр КУЗЬМЕНКО	01.04.2024	01.05.2024

Календарний план

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка задачі. Пошук, огляд та аналіз аналогічних розробок. Формулювання технічного завдання. Опрацювання літературних джерел.	2 лютого 2023 – 5 березня 2023	Виконано
2	Проектування структури розробки графіки	6 березня 2024 – 25 березня 2024	Виконано
3	Розробка базових об'єктів	26 березня 2024 – 26 квітня 2024	Виконано
4	Додавання шейдерів та візуальних ефектів	27 квітня 2024 – 14 травня 2024	Виконано
5	Оптимізація інтерактивного 3D світу	15 травня 2024 – 1 червня 2024	Виконано
6	Попередній захист	17 червня 2024	Виконано

Здобувач вищої освіти **Артур ТУРЛІЙ**
Керівник **Олександр КУЗЬМЕНКО**

РЕФЕРАТ

Кваліфікаційна робота включає в себе процес створення комп'ютерної графіки та супровідну пояснювальну записку. Остання містить у собі вступ, три розділи, висновки, список використаної літератури та додаток. Текстова частина викладена на 83 сторінках друкованого тексту. Пояснювальна записка має 19 сторінок додатків. Список використаних джерел містить 26 найменувань і займає 3 сторінки. В роботі наведено 77 рисунків та 4 таблиці. Загальний обсяг роботи – 102 сторінки.

В процесі виконання кваліфікаційної роботи було створено інтерактивний 3D світ для ігрового додатку в жанрі Roguelite. Було реалізовано спеціальні шейдери через Shader Graph, динамічні ефекти з використанням Particle System та проведено тестування гри. Використані інструменти включають: Unity, Shader Graph, Blender, Particle System та Unity Profiler. Мета була досягнута: розроблений ігровий додаток відповідає встановленим вимогам, має високу якість графіки та стабільну продуктивність.

Ключові слова: UNITY, ІГРОВИЙ ДОДАТОК, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, РЕФЕРЕНСИ, LEVEL, DESIGN, 3D, TEXTURE, ROUGELITE, PBR, SHADER GRAPH, NODES, PARTICLE SYSTEM.

					КН.КР.Б – 122 – 24 – ПЗ				
Змн.	Арк.	№ докум.	Підпис	Дата					
Розроб.		А. О. ТУРЛИЙ			Інтерактивний 3D світ для ігрового застосунку "Forbidden Forest" Пояснювальна записка	Літ.		Арк.	Аркушів
Керівник		О. В. КУЗЬМЕНКО						3	102
Рецензент		М.С. ГРАФ				Житомирська політехніка, група КН-20-2			
Н. контр.									
Зав. каф.		М.С. ГРАФ							

ABSTRACT

The qualification work includes the process of creating computer graphics and an accompanying explanatory note. The latter contains an introduction, three sections, conclusions, a list of references, and an appendix. The textual part is presented on 83 pages of printed text. The explanatory note consists of 19 pages of appendices. The list of used sources includes 26 titles and occupies 3 pages. The work includes 77 figures and 4 tables. The total volume of the work is 102 pages.

During the qualification work, an interactive 3D world was created for a game application in the Roguelite genre. Special shaders were implemented using Shader Graph, dynamic effects were created using Particle System, and the game was tested. The tools used include Unity, Shader Graph, Blender, Particle System, and Unity Profiler. The goal was achieved: the developed game application meets the established requirements, has high-quality graphics, and stable performance.

Keywords: UNITY, GAME APPLICATION, SOFTWARE, REFERENCES, LEVEL, DESIGN, 3D, TEXTURE, ROUGELITE, PBR, SHADER GRAPH, NODES, PARTICLE SYSTEM.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		4

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ.....	6
ВСТУП	7
РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ.....	9
1.1 Постановка задачі	9
1.2. Аналіз наявного програмного забезпечення за тематикою кваліфікаційної роботи.....	11
1.3 Обґрунтування вибору інструментальних засобів.	20
Висновки до першого розділу	23
РОЗДІЛ 2. ПРОЕКТУВАННЯ ТА РОЗРОБКА ГРАФІЧНИХ ЕЛЕМЕНТІВ ІГРОВОГО ЗАСТОСУНКУ.....	24
2.1 Візуалізація процесів розробки графічних елементів ігрового застосунку	24
2.2 Розробка інтерактивного 3D світу	30
2.3 Створення візуальних ефектів для ігрового застосунку.....	42
2.4 Реалізація програмного коду для створення шейдерів.....	54
Висновки до другого розділу.....	64
РОЗДІЛ 3. ОПИС РОБОТИ З ІГРОВИМ ЗАСТОСУНКУ ТА ЙОГО ТЕСТУВАННЯ.....	65
3.1 Опис роботи з ігровим додатком	65
3.2 Порядок встановлення, налаштування та тестування застосунку в ігровому рушії	72
Висновки до третього розділу	78
Висновки	79
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	80
ДОДАТКИ.....	83

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ПЗ	Програмне забезпечення
UV	(UV mapping) Координатна система, яка використовується в комп'ютерній графіці для текстуровання 3D моделей.
3D	Вид комп'ютерної графіки, яка використовує тривимірні моделі для створення зображень, що мають глибину, ширину та висоту
RGB	(Red, Green, Blue) колірна модель, що використовується для представлення та відображення кольорів у електронних пристроях.
UML	(Unified Modeling Language) – це уніфікована мова моделювання, яка використовується для візуалізації
PBR	(Physically Based Rendering) шейдери в Unity дозволяють створювати реалістичні матеріали за допомогою візуальних інструментів.

ВСТУП

Актуальність теми. Roguelite - це жанр відеоігор, який має цікаве походження та історію розвитку. Назва жанру походить від ігри «Rogue», яка була випущена у 1980 році. Оригінальна гра «Rogue» мала досить складні правила та особливості, такі як випадково генеровані рівні, непрості завдання та надзвичайно високий рівень випробувань для гравців.

Цей жанр не просто пропонує гравцям нові виклики кожного разу, а й створює непередбачуваний ігровий досвід, де кожне проходження приносить нові сюжетні лінії, врагів та ситуації. Гравці постійно змушені адаптуватися до змін, що забезпечує великий рівень цікавості та довготривалого інтересу до гри.

За останні роки жанр «Roguelite» здобув широку популярність завдяки своїм унікальним особливостям та можливостям, які він пропонує гравцям. Цей жанр поєднує в собі елементи випадкової генерації рівнів, стратегічного планування та постійних викликів через смерті персонажа. Він також стимулює розробників до інновацій в геймдизайні, сприяючи виникненню нових ігрових механік та ідей.

У Roguelites гравці можуть зберігати частину свого прогресу або досягнень між геймплеями, що робить гру менш випадковою та дозволяє поступово розвивати свого персонажа.

Однією з ключових особливостей Рогаликів є їхня велика переігровуваність. Гравці повертаються до цих ігор знову та знову, оскільки кожна нова спроба дає можливість вивчити нові стратегії, відкрити нові області гри та покращити свої навички. Ця постійна динаміка робить Рогалики дуже захоплюючими та привабливими для широкої аудиторії гравців.

Roguelites вважаються одними з найбільш інноваційних та захоплюючих жанрів, що продовжують привертати увагу як досвідчених геймерів, так і новачків. їхня актуальність полягає в надзвичайній гнучкості та креативності, яку вони дозволяють розробникам втілити у грі, а також у можливості створення непередбачуваних та захоплюючих ігрових досвідів для гравців.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		7

Сьогодні жанр «Roguelite» продовжує розвиватися та експериментувати з новими ідеями та механіками. Він став одним із символів творчості та інновацій в індустрії відеоігор, надаючи гравцям можливість насолоджуватися непередбачуваним та захоплюючим ігровим досвідом.

Отже, актуальність гри полягає в її здатності забезпечувати унікальний, непередбачуваний ігровий досвід з високим рівнем повторної гри та стимулювати інновації в геймдизайні.

Метою кваліфікаційної роботи є створення інтерактивного 3D світу для ігрового застосунку "Forbidden Forest".

Об'єктом дослідження є використання технологій для створення інтерактивного 3D світу "Forbidden Forest" для подальшого використання в відеогрі.

Предметом дослідження є розробка інтерактивного 3D світу для ігрового застосунку "Forbidden Forest" з використанням рушія UNITY, а також налаштування Particle Effect та Shader Graph для досягнення потрібних візуальних ефектів у грі.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		8

РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ

1.1 Постановка задачі

Метою кваліфікаційної роботи є створення інтерактивного 3D світу для ігрового застосунку "Forbidden Forest".

Основною метою розробки є створення захоплюючого і іммерсивного ігрового досвіду для гравців, який відповідає концепції та особливостям жанру Roguelite

Основні аспекти, які включаються у створення локації для гри в жанрі Roguelite, включають:

1. Процедурна генерація: локації в Roguelite гри можуть бути генеровані процедурно, що дозволяє створювати унікальні та непередбачувані об'єкти, ландшафти та шляхи кожного разу, коли гравець починає нову гру.
2. Різноманітність: важливо забезпечити різноманітність локацій, таких як ліси, гори, річки, поля тощо, що створює різноманітні зони для дослідження гравцями.
3. Елементи ландшафту: створення різних елементів ландшафту, таких як річки, озера, каміння, мости тощо, що додає глибину та реалізм до ігрової зони.
4. Розміщення об'єктів: розміщення об'єктів, які створюють атмосферу та настрої локації, таких як дерева, кущі, декоративні об'єкти, може значно підвищити іммерсію гравців.
5. Логіка та механіка локацій: реалізація логіки та механіки роботи локацій, таких як спеціальні події, взаємодія з об'єктами, можливість зміни ландшафту під впливом гравця тощо.
6. Атмосферна іммерсія: створення детального та реалістичного оточення, що допомагає гравцеві відчуватися частиною гри і погрузитися в світ.
7. Наративні елементи: інтеграція наративних елементів у локації, які допомагають розкрити історію та фонові події у грі.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				9
Змн.	Арк.	№ докум.	Підпис	Дата		

У рамках кваліфікаційної роботи передбачається створення ігрового застусинку "Forbidden Forest", які включатимуть широкий спектр елементів для створення захоплюючого іммерсивного ігрового досвіду. Для досягнення поставленої задачі потрібно виконати такі етапи розробки:

1. Аналіз та планування:

- a. Розглядання концепції та ідеї для локації, включаючи тематику, об'єкти, структуру та наративні елементи.
- b. Розробити детальний план розробки, визначивши кроки, терміни виконання та ресурси, необхідні для кожного етапу. Розробка системи рівнів деталізації.

2. Дослідження та концептуалізація:

- a. Провести дослідження стилевих та тематичних аспектів для ігрової локації.
- b. Знайти референси, концепт-арти, скетчі та схеми для візуалізації дизайну локації.

3. Моделювання та текстурування:

- a. Створити 3D моделі для об'єктів та елементів локації (дерева, стежки, будівлі тощо).
- b. Додавання текстур та матеріалів для кожної моделі, щоб надати їм реалістичний вигляд.

4. Освітлення та атмосфера:

- a. Налаштування освітлення, враховуючи час доби, погодні умови та атмосферу локації.
- b. Додавання ефектів, такі як туман, світла, тіней та інші атмосферні елементи.

5. Деталізація та додаткові ефекти:

- a. Додавання деталей, такі як рослини, камені, вода, трава тощо, для збагачення об'єму локації.
- b. Створення шейдерів анімації та взаємодіючі елементи, які підвищують іммерсію гравця (рухливі листя, рух води тощо).

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				10
Змн.	Арк.	№ докум.	Підпис	Дата		

6. Тестування та оптимізація:

- а. Провести тестування локації на різних пристроях.
- б. Оптимізація локації для забезпечення плавної та стабільної роботи гри.

7. Відладка та завершення

- а. виправлення помилок та недоліків, виявлені під час тестування.
- б. Завершення розробки локації та підготовка її для використання у грі.

1.2. Аналіз наявного програмного забезпечення за тематикою кваліфікаційної роботи.

Для аналізу було обрано декілька ігор зі схожим геймплеєм та візуалізацією.

«City of Brass» [\[1\]](#) - це пригодницька гра з відкритим світом, яка поєднує в собі елементи roguelike та розширюваного жанру. Гра розроблена студією Uppercut Games і випущена у 2018 році.

Головні особливості гри включають великі локації, випадково генеровані рівні, бойову систему з різноманітними зброями та магічними навичками, атмосферний графічний дизайн і високий рівень динаміки у геймплеї. Гра отримала позитивні відгуки за свій геймплей, дизайн та атмосферу, і вона вважається однією з цікавих представниць свого жанру (рис.1.1).



Рисунок 1.1 – Заставка гри «City of Brass»

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		11

Гра «City of Brass»: використовує стилізовану 3D графіку з деталізованими об'єктами та архітектурою, що створює вражаючий відкритий світ для дослідження. Колірна палітра використовується для створення атмосфери містичності та таємничості.

Таблиця 1.1

Ключові аспекти дизайну гри «City of Brass»

Параметр	Переваги	Недоліки
Дизайн локацій	Локації в грі «City of Brass» вражають своїм стилем, який поєднує елементи фантастичного міста, пустельних містечок, магічних печер та казкових ландшафтів. Кожна локація має унікальний дизайн і архітектурні особливості, забезпечуючи грі глибину та різноманітність	Для деяких гравців може виявитися, що деталізація графіки не на тому рівні, на якому вони очікували, особливо у порівнянні з іншими сучасними іграми.
Деталізація	Гра вражає своєю деталізацією як у великих структурах, так і у дрібних деталях. Кожен елемент, від текстурованих стін до декорацій, допомагає поглибити імерсію та створити реалістичний світ.	Деякі гравці можуть відчувати, що рівень деталізації графіки не відповідає сучасним стандартам, особливо порівняно з іншими іграми.
Атмосфера	Стиль графіки в грі "City of Brass" відтворює атмосферу магічного світу, де кожен куток може приховувати таємниці та небезпеки.	Хоча артстиль може бути чарівним для багатьох, деякі гравці можуть не відчувати його відповідним своїм вподобанням або стандартам.
Різнманітність локацій	Від великих міст до занедбаних печер, від палаців до таємничих таємниць лісів - "City of Brass" пропонує широкий спектр різноманітних локацій, кожна з яких має свої унікальні особливості та виклики для гравця.	Велика кількість різних локацій та архітектурних стилів може призводити до неоднорідності в графічному виконанні та стилістиці, що впливає на єдність гри.

Гра "City of Brass" має багато сильних сторін у своєму дизайні, особливо що стосується дизайну локацій, деталізації та атмосфери. Вражаючі локації, які поєднують елементи фантастичного міста, пустельних містечок, магічних печер та казкових ландшафтів, забезпечують грі глибину та різноманітність. Кожна локація має унікальний дизайн і архітектурні особливості, що створює захоплюючий імерсивний досвід (рис.1.2).

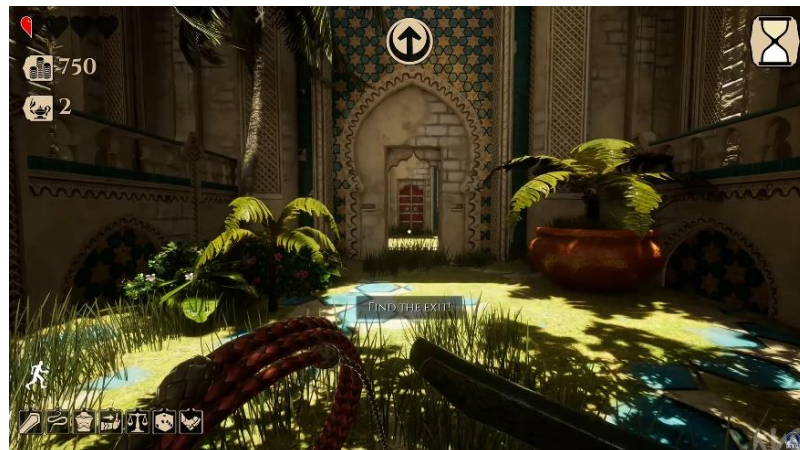


Рисунок 1.2 – Фрагмент гри «City of Brass», приклад дизайну локації

Проблеми можуть вплинути на сприйняття гри деякими гравцями, але враховуючи загальну позитивну оцінку гри і високий рівень задоволення від геймплею, багато гравців знаходять у "City of Brass" велику цінність у вигляді захоплюючого та вишуканого пригодницького досвіду. Графічні недоліки можуть бути виправлені або покращені за допомогою оновлень та патчів.

Важливо також враховувати, що різні гравці мають різні очікування від графіки та стилю гри, тому частина критики може бути суб'єктивною. Незважаючи на ці проблеми, гра "City of Brass" залишається популярною серед гравців завдяки своєму унікальному дизайну, атмосфері та різноманітності локацій.

У цілому, "City of Brass" пропонує багатий і захоплюючий ігровий досвід з унікальним візуальним стилем і різноманітними локаціями, але гра також має певні аспекти, які можуть вимагати вдосконалення для задоволення очікувань ширшої аудиторії.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		13



Рисунок 1.3 – Заставка гри «Borderlands»

«Borderlands» [2] - це серія науково-фантастичних шутерів від третьої особи з елементами рольової гри, розроблена компанією Gearbox Software і випущена відомим видавництвом 2K Games. Головні аспекти цієї серії, які будуть аналізовані за тематикою кваліфікаційної роботи, включають геймплей, візуальну естетику та особливості механіки гри (рис.1.3).

"Borderlands" розгортається в далекому майбутньому на планетах, які стали жертвами післяпостапокаліптичних подій. Головний упор в сюжеті робиться на пошуках скарбів, які лежать по всій галактиці, а також на боротьбі з різноманітними ворогами та небезпечними угрупованнями.

Таблиця 1.2

Ключові аспекти дизайну гри «Borderlands»

Параметр	Переваги	Недоліки
Штучний коміксний стиль	Одним із найвизначніших елементів графіки є її коміксний стиль, який надає грі унікальний вигляд. Великі контури, насичені кольори та чітка лінія створюють враження, ніби гравець перебуває у живому коміксі.	Оскільки оригінальна гра «Borderlands» була випущена у 2009 році, а графічні стандарти швидко змінюються, деякі гравці можуть вважати графіку застарілою або менш ефектною порівняно з новішими іграми.

Деталізація	Гра вражає своєю деталізацією як у великих структурах, так і у дрібних деталях. Кожен елемент, від текстурованих стін до декорацій, допомагає поглибити імерсію та створити реалістичний світ.	Незважаючи на те, що гра має свій унікальний стиль, деяким гравцям може здатися, що бракує деталізації у деяких областях, особливо порівняно з більш сучасними іграми.
Унікальний арт-стиль	«Borderlands» відрізняється своєрідним художнім стилем, який комбінує елементи коміксів з яскравими кольорами та чіткими обрисами персонажів та об'єктів. Цей арт-стиль став визначним для серії і робить гру відрізняючою серед інших шутерів	Деякі гравці можуть не оцінити стилізовану графіку та віддати перевагу реалістичнішій графіці. Арт-стиль може також обмежувати аудиторію гри, оскільки не всім гравцям подобаються такі художні рішення.
Імерсивний світ	Гра пропонує гравцям великий та детально опрацьований світ для дослідження. Кожна локація насичена атмосферними деталями, що допомагає занурити гравців у гру. Відкритий світ дозволяє вільно досліджувати, виконувати завдання та взаємодіяти з оточенням, що створює відчуття реалістичності та глибини.	Величезний світ може викликати у деяких гравців відчуття перевантаженості та розгубленості через велику кількість контенту та завдань. Крім того, можливо, що деякі частини світу можуть здаватися пустими або недопрацьованими в порівнянні з іншими.
Експресивність персонажів	Персонажі у грі мають яскраві та виразні обличчя, що допомагає підсилити їхню індивідуальність та характер. Вони виглядають живими та запам'ятовуються завдяки своєму дизайну. Завдяки ретельному опрацюванню міміки та жестів.	Стилізована міміка та жести персонажів виглядають надто карикатурно або нереалістично. Крім того, такий підхід може не відповідати загальному тону гри для тих, хто віддає перевагу більш серйозним та реалістичним персонажам.

Рівні	Процедурно генеровані рівні, що забезпечує унікальний досвід кожного разу при грі.	Процедурна генерація може призвести до нерівномірного якості рівнів і відсутності належного балансу.
Тематика локацій	Різноманітність тематик і стилів локацій, від міст футуристичного стилю до занедбаних пустельних містечок, створюють унікальну атмосферу та глибину світу.	Можливість зіткнення з несподіваними або несумісними тематиками у локаціях, що може порушити єдність гри або атмосферу.

В "Borderlands" гравці відіграють роль мисливців за скарбами, які шукають легендарні скарби, що приховані на різних планетах. У своїх пригодах вони зіштовхуються з численними ворогами, включаючи бандитів, мутантів, роботів та інших небезпечних сутностей (рис.1.4).



Рисунок 1.4 - Унікальний арт-стиль гри «Borderlands»

Загалом, Гра «Borderlands» вразила світ своїм унікальним комікським стилем, глибокими персонажами та відкритим світом для дослідження. Її динамічний геймплей, широкий арсенал зброї та інтригуючий сюжет зробили її популярною серед гравців шутерів з елементами RPG. Гра також вирізняється своєю кооперативною мультиплеєрною механікою, що дозволяє гравцям об'єднуватися для проходження місій і боротьби з ворогами. Завдяки своєму

неповторному гумору і незабутнім лиходіям, «Borderlands» стала культовою класикою і вплинула на розвиток жанру лутер-шутерів.



Рисунок 1.5 – Заставка гри «The Cycle»

«The Cycle» [3] - це відеогра у жанрі шутерів з видом від третьої особи (TPS) з елементами батл-роялу та відкритого світу. Гра розроблена компанією YAGER Development і випущена у форматі free-to-play (рис.1.5).

У The Cycle гравці вирушають на віддалену планету Fortuna III, де вони змагаються за ресурси, виконують завдання та виживають у ворожому середовищі. Гра поєднує в собі елементи шутера, ролевої гри та стратегії, а також має унікальну систему гри, де гравці можуть обирати між співпрацею та конкуренцією з іншими учасниками.

Таблиця 1.5

Ключові аспекти дизайну гри «The Cycle»

Параметр	Переваги	Недоліки
Візуальна стилізація	Графіка «The Cycle» має вражаючу візуальну стилізацію, яка поєднує науково-фантастичні елементи з деталями планетних пейзажів та футуристичної архітектури. Це створює враження дивовижного світу, що чекає на дослідження	Можливі вимоги до високопродуктивного обладнання для повноцінного відтворення графіки. Втрати актуальності стилю у майбутньому, що може вплинути на привабливість гри.

Інтерактивність локацій	Локації у грі «The Cycle» мають високий рівень інтерактивності, що дозволяє гравцям взаємодіяти з оточенням та використовувати його для досягнення цілей.	Інтерактивність може бути обмеженою в деяких локаціях, що може знизити загальне задоволення від гри.
Навігація та орієнтування	Дизайн локацій дозволяє гравцям легко орієнтуватися в ігровому світі завдяки чітким візуальним підказкам та зручному інтерфейсу.	Деякі локації можуть бути надто складними для навігації, що може викликати фрустрацію у гравців.
Атмосфера та занурення	Стиль графіки та дизайн локацій створюють глибоку атмосферу, що занурює гравця у фантастичний світ.	Деякі гравці можуть відчувати, що атмосфера гри не відповідає їхнім очікуванням або не настільки захоплююча.
Деталізація середовища	У грі вражаюча деталізація середовища, включаючи різноманітність рослинності, технологічні споруди та неймовірні краєвиди. Це додає іммерсивності та реалістичності геймплею	Збільшення обсягу даних та часу завантаження гри через велику кількість деталей. Можливість розсіювання уваги гравців через перенасиченість деталями.
Ефекти та освітлення	Гра використовує вражаючі спецефекти та динамічне освітлення, що створює атмосферу загадковості та напруження в різних сценах	Хоча артстиль може бути чарівним для багатьох, деякі гравці можуть не відчувати його відповідним своїм вподобанням.

Гра «The Cycle» має багато переваг, включаючи унікальну візуальну стилізацію, високу деталізацію середовища, вражаючі ефекти та освітлення, реалістичну фізику та анімації, а також інноваційні геймплейні механіки та соціальні взаємодії. Особливо слід відзначити дизайн локацій, який вирізняється своєю різноманітністю та інтерактивністю.

Однак, гра також стикається з певними проблемами, такими як можливе відставання графіки від сучасних стандартів, неоднорідність у стилі локацій, складність механік для новачків і токсична поведінка деяких гравців. Ці проблеми можуть вплинути на сприйняття гри деякими користувачами, але загалом гра отримує високу оцінку за свою інноваційність та глибину ігрового процесу.

Отже, було вирішено створити власний інтерактивний 3D світ для ігрового застосунку "Forbidden Forest" з стилізованою графікою. Важливо щоб гра відповідала таким графічним критеріям:

1. Стилізована графіка: гра повинна мати унікальний стиль графіки, який відображає атмосферу та настрій "Forbidden Forest". Це стилізація в сторону фентезі, магії.
2. Деталізація світу: світ гри має бути деталізованим і багатим на різноманітність об'єктів, ландшафтів та архітектурних елементів, щоб створити враження реалістичності та іммерсії.
3. Яскравість та насиченість: графіка повинна бути яскравою та насиченою кольорами, що додасть життя та динаміку до світу гри.
4. Атмосферне освітлення: використання атмосферного освітлення допоможе створити настрій у різних частинах світу, таких як лісові чагарники, темні печери або містичні замки.
5. Ефекти та анімація: додавання спеціальних ефектів, таких як частки, світлові ефекти, рухливість рослин та інші деталізовані анімації, підвищить рівень реалізму та іммерсії гри.
6. Графічний стиль повинен відповідати тематиці гри та підкреслювати її основні ідеї, створюючи атмосферу, яка відображає концепцію гри.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				19
Змн.	Арк.	№ докум.	Підпис	Дата		

1.3 Обґрунтування вибору інструментальних засобів.

На даний момент популярними платформами для розробки ігор є Unity, Unreal Engine, Godot Engine, CryEngine, GameMaker Studio та інші. Ці платформи володіють великою кількістю користувачів та широким спектром можливостей для створення різних видів ігор. Unity та Unreal Engine пропонують багатий набір інструментів для створення високоякісної графіки, складної фізики та анімації, що робить їх ідеальними для розробки як інді, так і AAA проєктів.

Було проведено аналіз двох популярних платформ для створення гри, Unreal Engine та Unity. Обидві програми є потужними платформами для створення гри, проте кожна має свої унікальні переваги. Unity відома своєю простотою використання та гнучкістю, тоді як Unreal Engine вражає своїми візуальними можливостями та реалістичною графікою.

Однак вони мають різні переваги та недоліки:

Таблиця 1.4

Порівняння ігрових рушіїв Unreal Engine та Unity

Параметр	Unreal Engine	Unity
Графіка та візуальні можливості	Визначається своєю потужною графікою та можливостями фотореалістичних зображень. Це особливо корисно для створення ігор з високою рівнем деталізації та реалізму.	Володіє широким спектром можливостей для розробки 2D та 3D ігор.
Ефекти та освітлення	Unreal Engine має вбудовані інструменти для створення різноманітних ефектів, освітлення та анімації, що дозволяє розробникам створювати вражаючі візуальні ефекти.	Хоча Unity також має добре розвинену графіку, вона частіше використовується для створення ігор з меншою ступенем деталізації.

Спільнота та підтримка	Надає досить велику та активну спільноту розробників, а також широку базу знань та документацію. Однак, офіційна підтримка може бути менш доступною у порівнянні з Unity.	Відомий своєю великою та активною спільнотою, що дозволяє легко знаходити відповіді на питання та отримувати допомогу від інших розробників. Unity також має широкий вибір підтримки та документації.
Складність використання	Unreal Engine має деяку складність у використанні, особливо для початківців. Його інтерфейс може здаватися складним у порівнянні з Unity через більшу кількість налаштувань та опцій.	Легкий у використанні, особливо для тих, хто тільки починає знайомитися з розробкою ігор. Unity має простий інтерфейс та широкий набір готових рішень, що дозволяє швидко створювати прототипи та експериментувати з ідеями.
Можливості для VR та AR	Має великий потенціал для розробки VR та AR додатків. Його фотореалістична графіка та потужність дозволяють створювати іммерсивні віртуальні середовища.	Є одним з лідерів у розробці VR та AR проектів. Unity має готові рішення та плагіни для роботи з цими технологіями, що робить його популярним в цих сферах.
Ціноутворення	Unreal Engine пропонує безкоштовний доступ до своїх інструментів з ліцензійною угодою, що включає роялті у розмірі 5% від доходу гри після досягнення певного порогу прибутку. Це може бути вигідним для невеликих команд та інді-розробників	Unity пропонує різні плани, включаючи безкоштовну версію для індивідуальних розробників та невеликих команд, а також платні підписки для професійного використання.

Інтеграція з іншими інструментами	Unreal Engine добре інтегрується з інструментами для створення контенту, такими як Maya, 3ds Max та Houdini. Це дозволяє легко імпортувати та використовувати високоякісні моделі та анімації в проектах. Крім того, Unreal Engine має потужні вбудовані інструменти для створення візуальних ефектів та матеріалів.	Unity має широкі можливості інтеграції з різними інструментами та плагінами. Він підтримує роботу з багатьма популярними програмами для створення контенту, такими як Blender, Photoshop та Substance Painter. Unity також має власний Asset Store, де можна знайти безліч готових ресурсів та інструментів для прискорення розробки.
Платформи	Unreal Engine підтримує широкий спектр платформ, включаючи ПК, консолі, мобільні пристрої, VR та AR. Це дозволяє розробникам створювати гри для різних пристроїв та аудиторій.	Unity також підтримує безліч платформ, від мобільних пристроїв до ПК та консолей, а також VR та AR. Завдяки широкій підтримці платформ та легкості розгортання проектів, Unity є популярним вибором для розробників, які прагнуть досягти максимальної аудиторії.
Навчальні ресурси	Unreal Engine надає великий вибір офіційних навчальних ресурсів, включаючи відео-уроки, документацію та форуми. Epic Games також пропонує безкоштовні курси та освітні програми, що допомагає розробникам швидко освоїти використання двигуна.	Unity надає великий вибір навчальних матеріалів, включаючи відео-уроки, документацію, курси та форуми. Unity Learn – це офіційний портал для навчання, що пропонує безкоштовні та платні курси для всіх рівнів, від початківців до професіоналів.

Було обрано платформа розробки Unity:

1. Легкість використання: Unity відомий своєю легкістю використання, особливо для початківців. Інтерфейс Unity є інтуїтивно зрозумілим, що дозволяє швидко розпочати роботу над проектом.
2. Велика спільнота та підтримка: Unity має велику та активну спільноту розробників, яка надає багато корисних ресурсів, порад та допомогу. Це важливо для отримання підтримки та вирішення проблем під час розробки гри.
3. Мультиплатформеність: Unity дозволяє легко експортувати ігри на різні платформи, включаючи Windows, macOS, Linux, iOS, Android, та інші. Це забезпечує широкий охоплення аудиторії.

Висновки до першого розділу

Було визначено мету та сформульовано завдання для розробки. Проведено аналіз трьох відомих ігор (Borderlands, City of Brass, The Cycle), виявлено їх сильні та слабкі сторони. Визначено необхідну візуалізацію для створення інтерактивного 3D світу ігрового застосунку "Forbidden Forest".

Обґрунтовано вибір інструментарію Unity, для розробки інтерактивного 3D світу та візуалізації, які використовуються в ігровому застосунку "Forbidden Forest". Розглянуто переваги та недоліки, що будуть в нагоді для проекту, що розробляється.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		23

РОЗДІЛ 2. ПРОЕКТУВАННЯ ТА РОЗРОБКА ГРАФІЧНИХ ЕЛЕМЕНТІВ ІГРОВОГО ЗАСТОСУНКУ

2.1 Візуалізація процесів розробки графічних елементів ігрового застосунку

Діаграми UML (Unified Modeling Language) є загальноприйнятною мовою, яку розуміють розробники, аналітики та замовники. Вони використовуються для моделювання різних аспектів системи. Діаграми станів описують поведінку об'єктів у системі, самої системи або етапи розробки графічних об'єктів.

Діаграма активностей (рис. 2.1) відображає етапи розробки концепції 3D світу для ігрового застосунку. Цей інструмент допомагає краще зрозуміти процес і послідовність дій, необхідних для створення ігрового продукту, представляє етапи розробки концепції комп'ютерної гри.

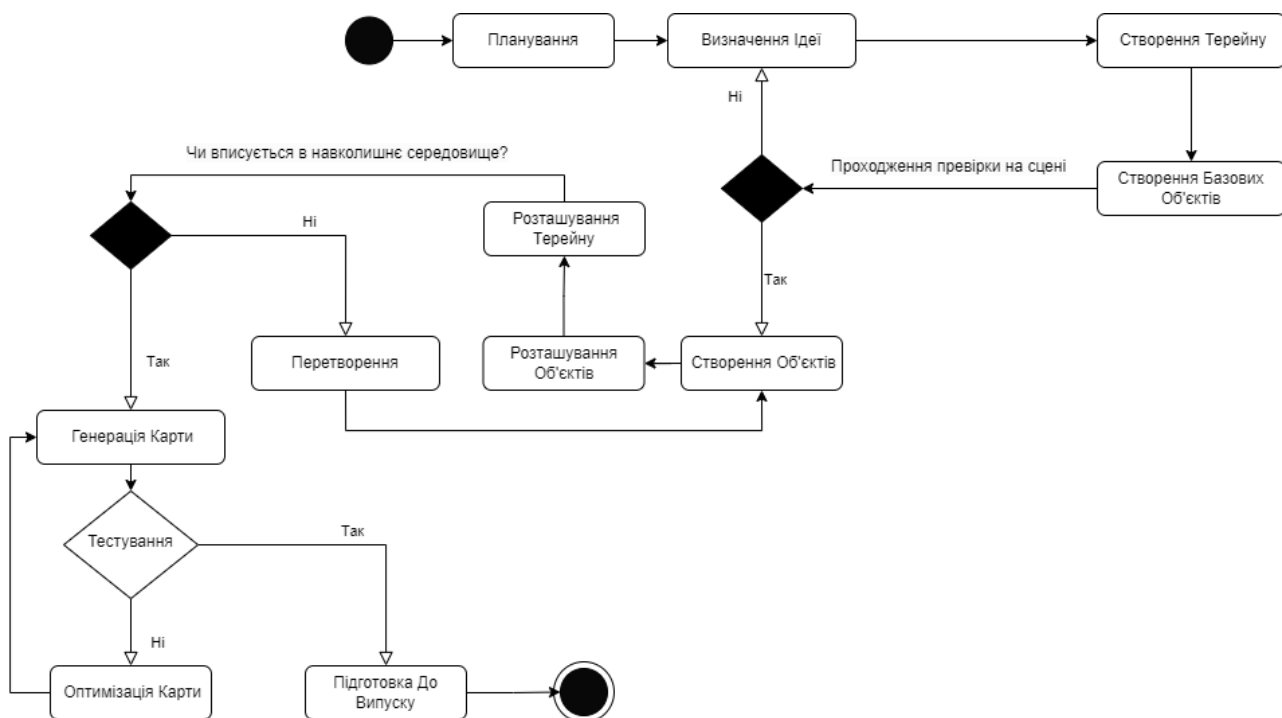


Рисунок 2.1 – Діаграма (activity) етапів розробки ігрової карти

За початковий етап процесу створення ігрової карти у Roguelite гри в середовищі Unity слід вважати планування. Цей етап включає визначення загальної концепції карти і визначення основних елементів, які потрібно створити.

Після планування йде етап створення терейну. На цьому етапі розробляється основна структура мапи, включаючи ландшафт, рельєф та рівнини. Після цього йде додавання деталей, що включає в себе розміщення різних об'єктів, рослинності, будівель тощо.

Далі слід зацікавитися розташуванням деталей, що передбачає побудову додаткових елементів до терейну, таких як водойми, гори, печери тощо. Після цього проводиться розміщення раніше створених деталей на мапі з метою забезпечення логічного та естетичного розташування.

Після вищезгаданих етапів виконується генерація процедурної мапи, яка передбачає створення процедурних алгоритмів для генерації випадкових елементів мапи. Потім йде тестування мапи для перевірки правильності генерації та відтворення мапи для тестування ігрової механіки. Після тестування виконується оптимізація мапи для вдосконалення мапи з метою забезпечення оптимальної продуктивності та якості гри.

На завершення процесу створення ігрової карти у Roguelite гри в Unity стає затвердження мапи, що передбачає підтвердження готовності мапи для включення її в гру та підготовку до випуску (рис. 2.1).

Наступна UML-діаграма активності описує процес створення локації для Roguelite гри з урахуванням можливості зміни концепту гри.

Починаємо з визначення теми, стилю та атмосфери локації, включаючи вибір типу середовища, архітектурних елементів, пейзажу та інших характеристик, що створюють унікальну атмосферу гри. Важливо враховувати кольорову палітру, освітлення та звукові ефекти, які додадуть глибини та реалістичності кожній локації. Далі переходимо до детального проектування окремих зон, враховуючи інтерактивні елементи та можливі маршрути гравця. Завдяки ретельному плануванню та дизайну, кожна локація буде не лише естетично привабливою, а й функціональною, що забезпечить захоплюючий ігровий процес.

Далі розробляємо загальний план архітектури локації, включаючи планування маршрутів, розміщення ключових об'єктів, визначення зон доступу.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		25

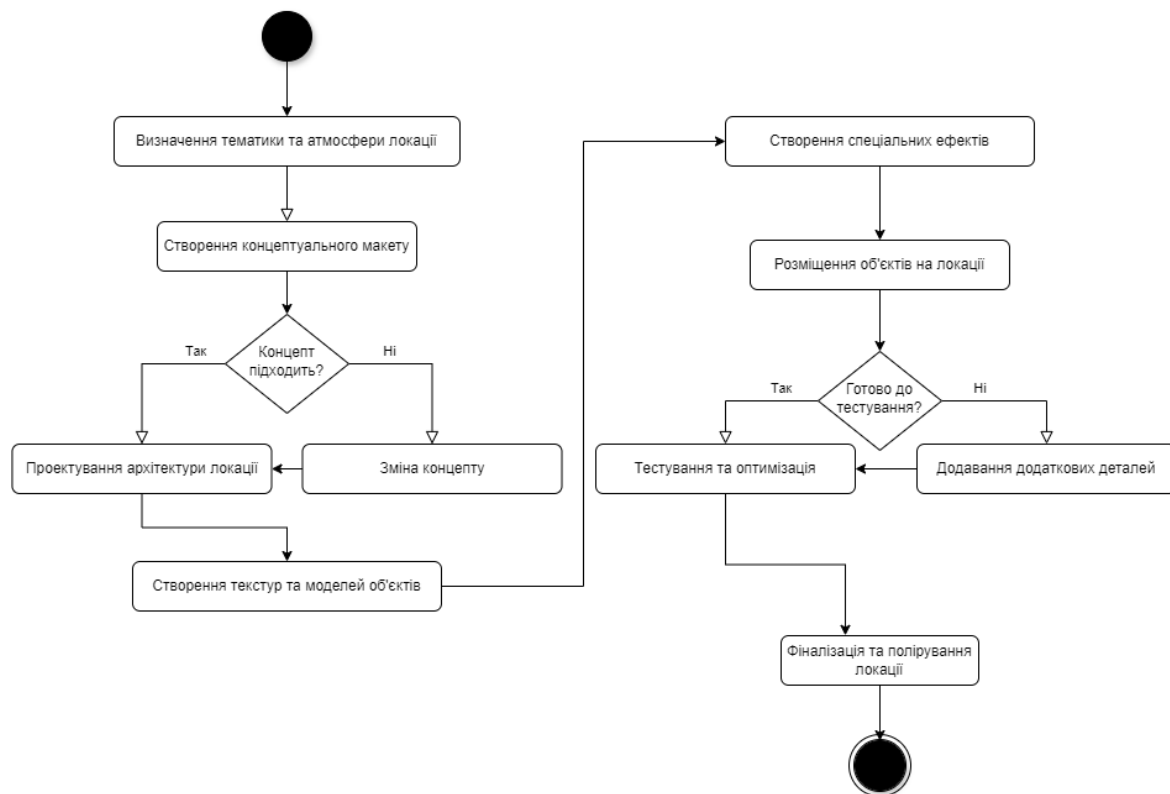


Рисунок 2.2 – Діаграма активності, що описує процес створення локації

Далі створюємо текстури та матеріали для об'єктів, які допомагають створити реалістичний вигляд об'єктам та пейзажам. Після цього розміщуємо створені об'єкти на локації згідно з архітектурним планом, дбаючи про правильне розташування для створення гармонійної та цікавої гри. Тоді додаємо додаткові деталі та ефекти, такі як освітлення, анімації, спеціальні ефекти тощо, для зроблення локації більш живою та іммерсивною.

Після завершення робіт проводимо тестування локації для виявлення помилок та оптимізації продуктивності гри. На останньому етапі завершуємо роботу над локацією, фіналізуємо деталі, додаємо атмосферні звуки, спеціальні ефекти та інші додаткові елементи для створення повноцінного і цікавого геймплею (рис. 2.2).

Діаграми послідовності відображають, як об'єкти взаємодіють у часі, і відображають поведінку ігрового застосунку. Діаграма послідовності (рис. 2.3) описує послідовний процес розробки 3D світу ігрового застосунку "Forbidden Forest". Насправді процес може бути більш складним і потребувати додаткових кроків, залежно від типу ігрового застосунку та використовуваних технологій.

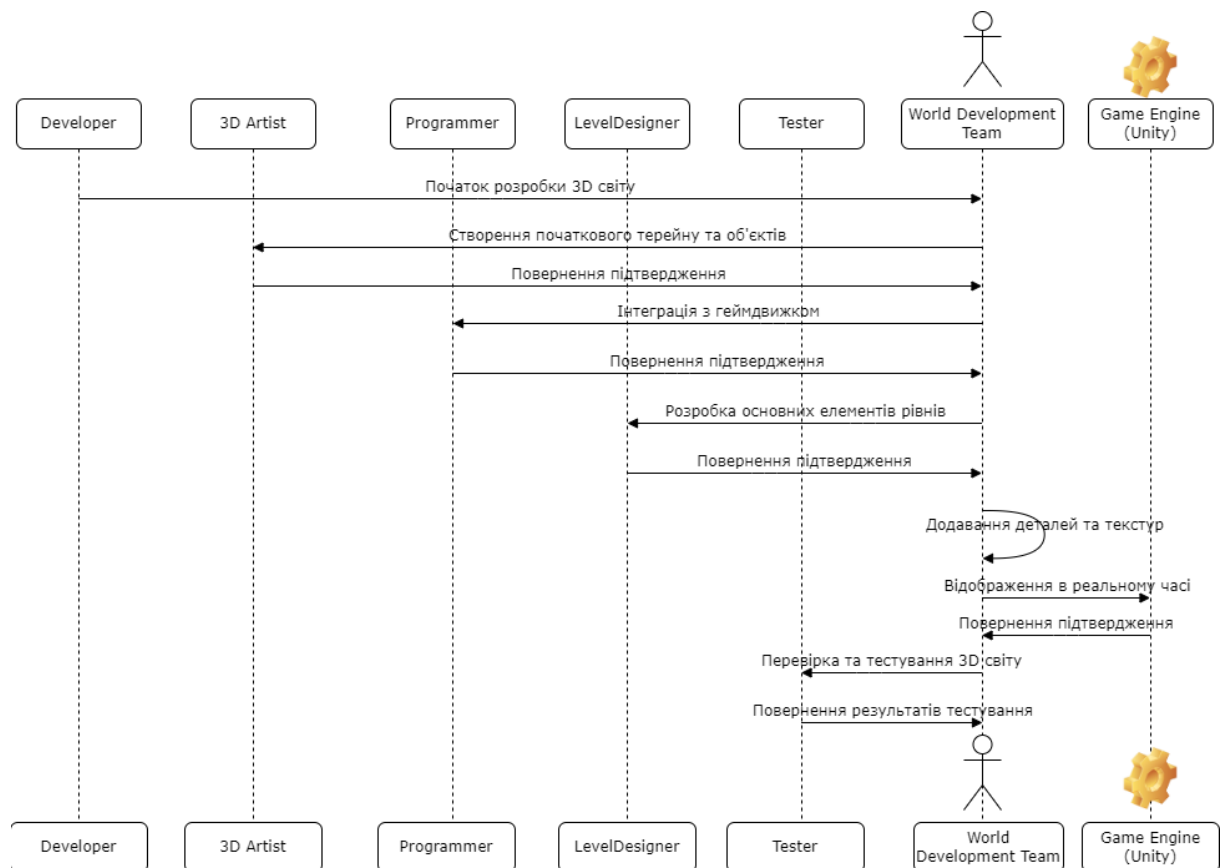


Рисунок 2.3 – Діаграма послідовності процесу розробки 3D світу ігрового застосунку "Forbidden Forest"

Процес розробки інтерактивного 3D світу для ігрового застосунку є ітеративним і вимагає тісної співпраці між різними фахівцями. 3D-художник, програміст, дизайнер рівнів та тестувальник постійно працюють разом, щоб покращити візуальну якість гри та ігровий процес.

Діаграми варіантів використання (use case diagrams) описують функціональні вимоги до програмної системи, представляючи акторів, варіанти використання та відносини між ними. У даному випадку діаграма послідовності (sequence diagram) описує процес отримання та виконання роботи по створенню готового гри. Вона показує, як різні учасники взаємодіють між собою, які дії вони виконують та в якому порядку, щоб досягти результату (рис. 2.4).

Команда розробників збирається для визначення загальної концепції гри та 3D світу. Проектний менеджер та дизайнер визначають основні вимоги та функціональні можливості 3D світу, включаючи ландшафт, об'єкти, освітлення та ефекти.

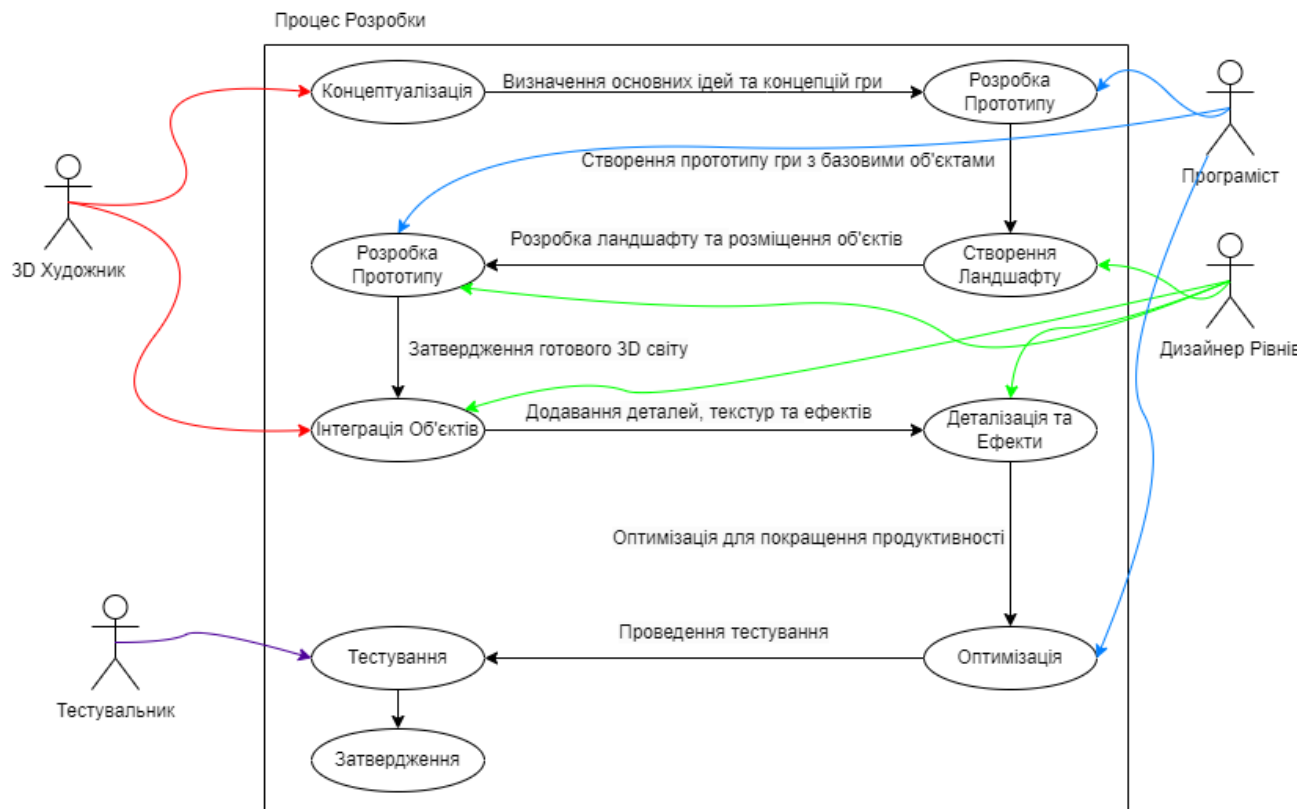


Рисунок 2.4 – Діаграма варіантів використання процесу розробки 3D світу для створення ігрового застосунку "Forbidden Forest"

Дизайнер розробляє концепцію світу, включаючи скетчі локацій, мапи рівнів та основні об'єкти. Художник 3D моделює базові об'єкти, такі як дерева, камені, будівлі та інші елементи, які потрібні для створення прототипу.

Художники додають деталі до ландшафту, такі як трава, кущі, водні елементи та інші природні об'єкти. Програмісти вбудовують реалістичні ефекти, такі як динамічна рослинність, анімація води, частинки пилу тощо.

Програмісти розробляють логіку для взаємодії з об'єктами на мапі, такими як взаємодія зі сховищами, дверима, рухомими об'єктами тощо. Художники створюють додаткові об'єкти, які розширюють інтерактивні можливості світу, такі як обладнання, зброя, предмети для збору.

Художники та програмісти спільно працюють над створенням шейдерів для покращення візуального враження, таких як реалістичне освітлення, тіні, водні ефекти тощо. Додаються спеціальні ефекти, такі як погода, частинки атмосфери, вогні, які допомагають створити атмосферу та настрій гри.

Команда тестувальників проводить тестування, перевіряючи функціональність, геймплей та взаємодію з 3D світом. Програмісти оптимізують світ для підвищення продуктивності та забезпечення стабільності гри.

Після успішного завершення тестування та оптимізації, 3D світ "Forbidden Forest" затверджується для включення до гри. Готується до випуску, де гравці зможуть насолоджуватись його іммерсивними можливостями у світі Roguelite.

Діаграма варіантів використання описує процес розробки 3D світу для ігрового застосунку "Forbidden Forest", залучаючи різні ролі в команді розробників. Основні кроки цього процесу включають концептуалізацію, розробку прототипу, створення ландшафту, інтеграцію об'єктів, деталізацію та додавання ефектів, оптимізацію, тестування та затвердження.

У діаграмі виділено такі ролі:

1. «3D Художник» відповідає за створення графічних об'єктів, текстур та моделей.
2. «Програміст» займається програмуванням логіки гри, взаємодіє з геймдвижком та робить інтеграцію об'єктів у середовище гри.
3. «Тестувальник» відповідає за перевірку функціональності та якості гри.
4. «Дизайнер Рівнів» відповідає за створення рівнів гри та їхню інтеграцію у 3D світ.

Процес починається з визначення основних концепцій та ідей для 3D світу, які визначаються командою розробників. Після цього переходить до розробки прототипу, де створюються базові об'єкти та моделі для подальшого використання. Далі йде процес створення ландшафту, додавання об'єктів та деталей, що виконується 3D художниками та дизайнерами рівнів.

Програмісти відповідають за інтеграцію цих елементів, створення логіки взаємодії об'єктів у грі та оптимізацію для забезпечення швидкої та стабільної роботи. Після завершення розробки відбувається тестування гри для перевірки її функціональності та якості. Після успішного тестування готовий 3D світ затверджується для включення до гри, що завершує процес розробки.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		29

Ця діаграма допомагає візуалізувати послідовність дій та взаємодію між різними ролями в команді розробників, що сприяє ефективній роботі та досягненню поставлених цілей у розробці ігрового застосунку.

2.2 Розробка інтерактивного 3D світу

Створення інтерактивного 3D світу для ігрового застосунку "Forbidden Forest" включає декілька етапів, алгоритм якого є:

Крок 1: Визначення концепції (рис. 2.1).

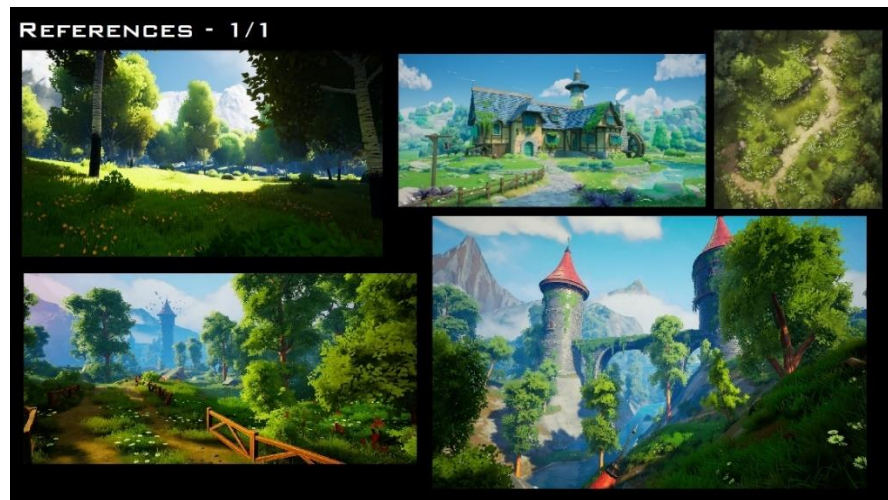


Рис. 2.1. Визначення загальної концепції рівня.

Зпочатку потрібно визначити загальну концепцію рівня. Це може бути пригода через ліс, дослідження вузьких гірських стежок, або взаємодія з місцевими традиціями і міфами. Ознайомлення з основними елементами, такими як гори, ліси, річки, і це є основне джерело натхнення.



Рисунок 2.2 – Референси для створення інтерактивного 3D світу.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		30

Далі шукаємо референси для створення інтерактивного 3D світу Roguelite гри (рис. 2.2). Описуючи процес пошуку референсів, можна розглянути такі кроки:

Онлайн-пошук (рис. 2.3): на цьому етапі використано пошукові системи та інтернет-ресурси для пошуку зображень, фотографій, ілюстрацій та інших візуальних матеріалів, які відповідають визначеній концепції гри. Ці матеріали служать вихідною точкою для подальшого створення ігрового світу та його елементів.

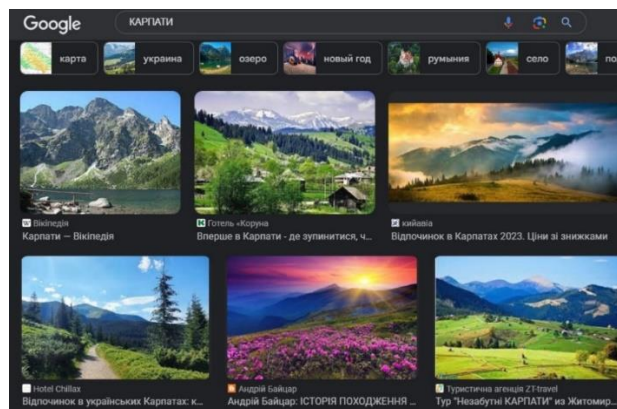


Рисунок 2.3 - Онлайн-пошук.

Книги та журнали (рис. 2.4): здійснювався пошук книг та журналів в онлайн-каталогах з метою знаходження джерел, присвячених тематиці ігрового застосунку "Forbidden Forest". Досліджувалися книги та журнали, які містять інформацію про фентезійні світи, інтерактивні додатки та інші теми, що стосуються розробки ігор.

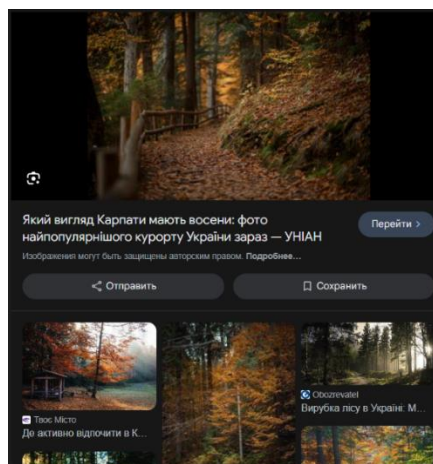


Рисунок 2.4 – Перегляд віртуальних каталогів.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		31

Фотографії та подорожі: фотографії природних ландшафтів, старовинних будівель, фауни та флори були використані для отримання ідей щодо вигляду та дизайну локацій. Наприклад, зображення старовинного замку може надихнути на створення руїн в грі.



Рисунок 2.5 – Пошук референсів в інтерактивних джерелах.

Інтерактивні джерела (рис. 2.5): ці джерела включають в себе різноманітні джерела даних та ресурсів, які використовуються для натхнення, дослідження та створення різноманітних елементів гри. Це використання інтерактивних карт, віртуальних турів, 360-градусних фотографій, а також використання різноманітних онлайн-інструментів для візуалізації інформації та вражень від подорожей.



Рисунок 2.6 – Пошук референсів в фільмах

Фільми та відеоматеріалів (рис. 2.6 - 2.7): цей етап включав в себе детальний аналіз різноманітних фільмів та відеоматеріалів, спрямованих на

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		32

створення відповідної атмосфери та настрою для фентезійного лісового світу гри. Використання відеозаписів дозволило точно відобразити природу та архітектурні особливості.

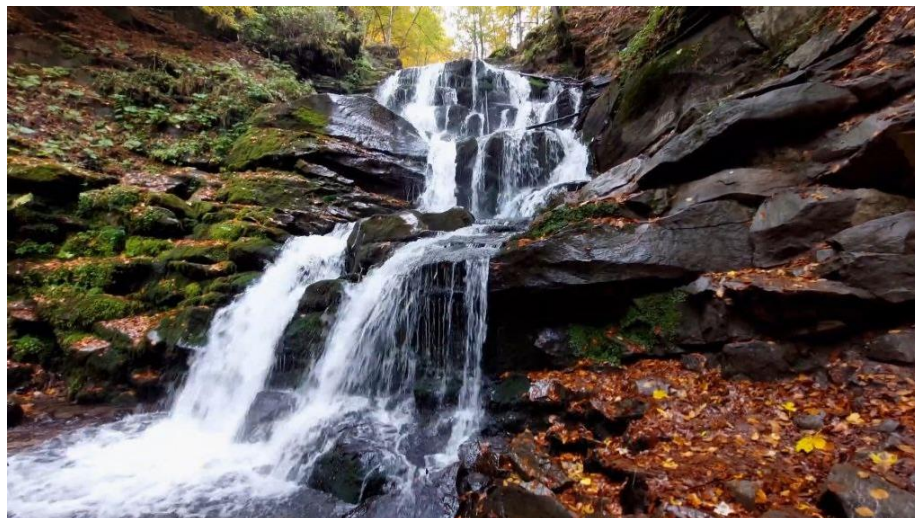


Рисунок 2.7 – Пошук референсів в відеоматеріалах

Комбінація цих джерел допомагло знайти різноманітні референси та ідеї для створення інтерактивного 3D світу для гри "Forbidden Forest". Важливо використовувати ці референси як натхнення, а не копіювання, і впроваджувати власні творчі ідеї в процес створення.

Крок 2: Створення мапи рівня (рис. 2.8).

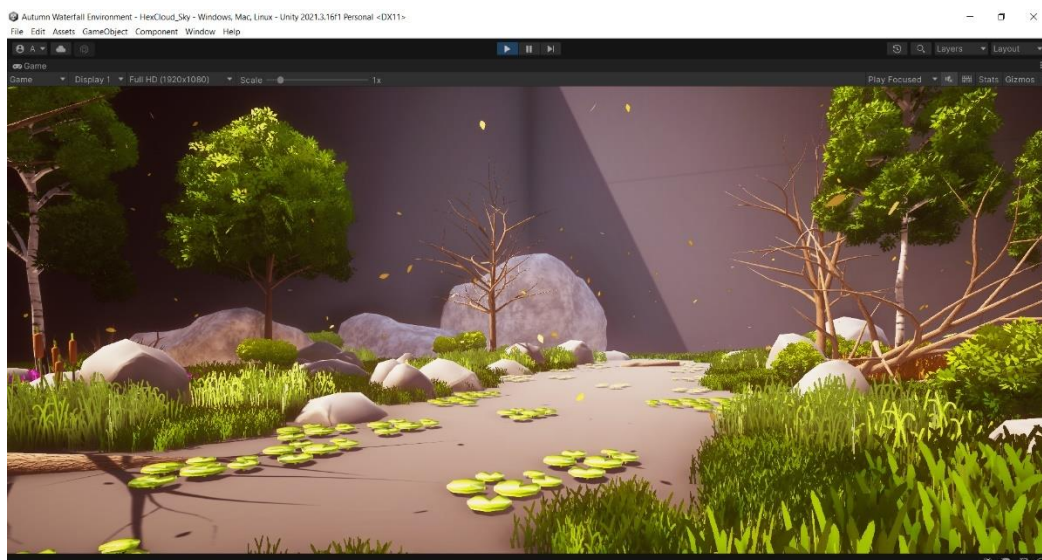


Рисунок 2.8 – Створення мапи рівня.

Під час виконання цього етапу потрібно визначити основні області, шляхи і перешкоди. Врахувати елементи, які є характерними для фентезійного світу,

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		33

грає ключову роль у створенні аутентичного та привабливого віртуального середовища.

Процес планування мапи рівня для Roguelite гри в фентезійному світі включає кілька кроків, що допомагає створити цікавий та затягуючий геймплей.

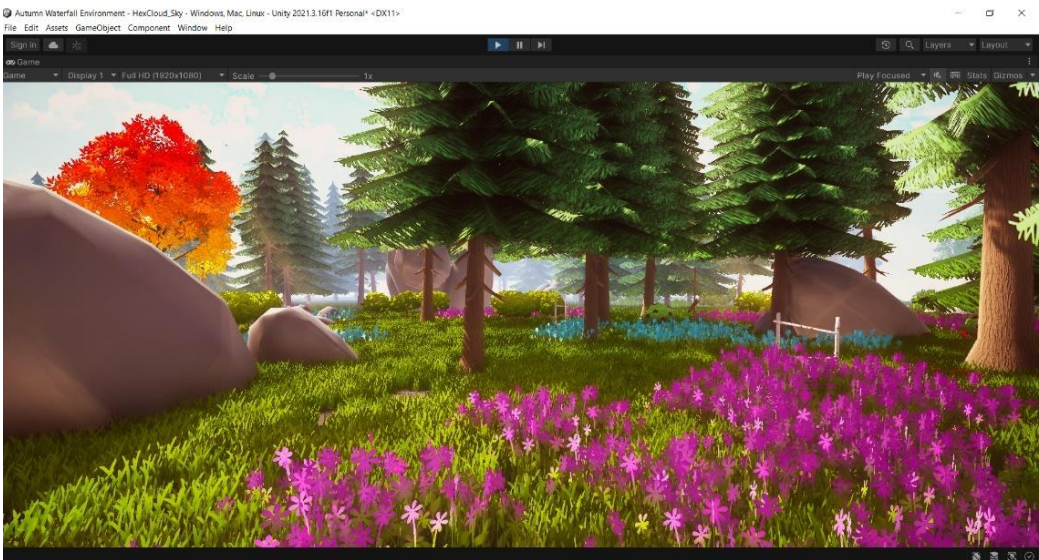


Рисунок 2.9 – Виклики та завдання по всій мапі рівня.

Спочатку потрібно створити загальну структуру рівня (рис. 2.9 - 2.10), враховуючи різні області та локації, які потрібно включити. Наприклад, це може бути лісова зона, старовинне селище, гірська річка та гори. Було встановлено логічні зв'язки між цими областями та шляхи для переміщення гравця.

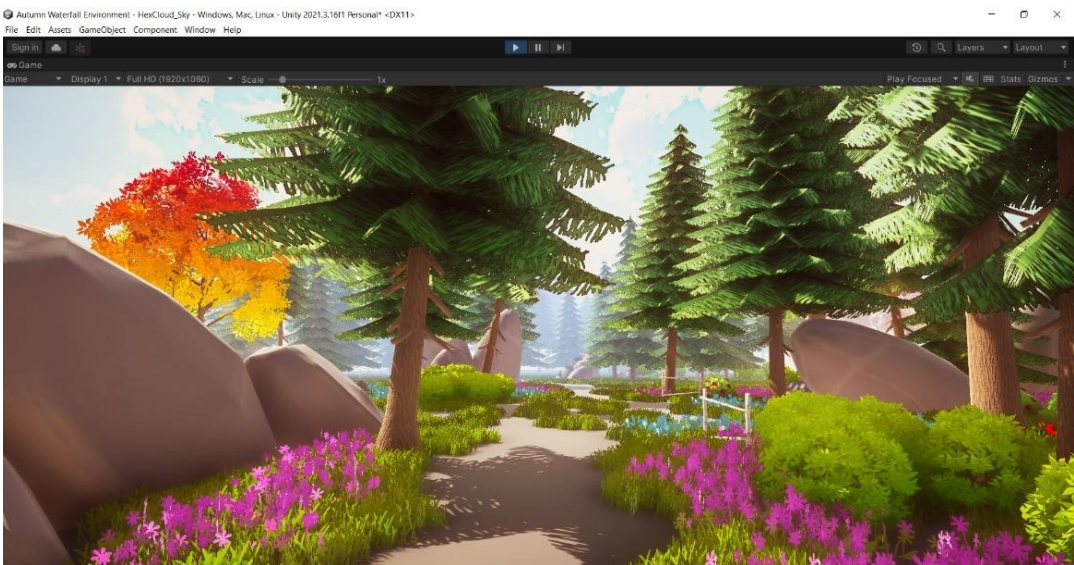


Рисунок 2.10 – Загальна структура рівня.

Далі розподілено виклики та завдання по всій мапі рівня (рис. 2.9). Це можуть бути головні завдання, які просувають сюжет, або додаткові завдання,

які розкривають додаткові деталі про світ гри. Враховано рівень складності завдань та їх залежність від розкладу мапи. Завдяки цьому підходу гравці отримають насичений та захоплюючий досвід, який поступово розкриває всі аспекти ігрового світу.

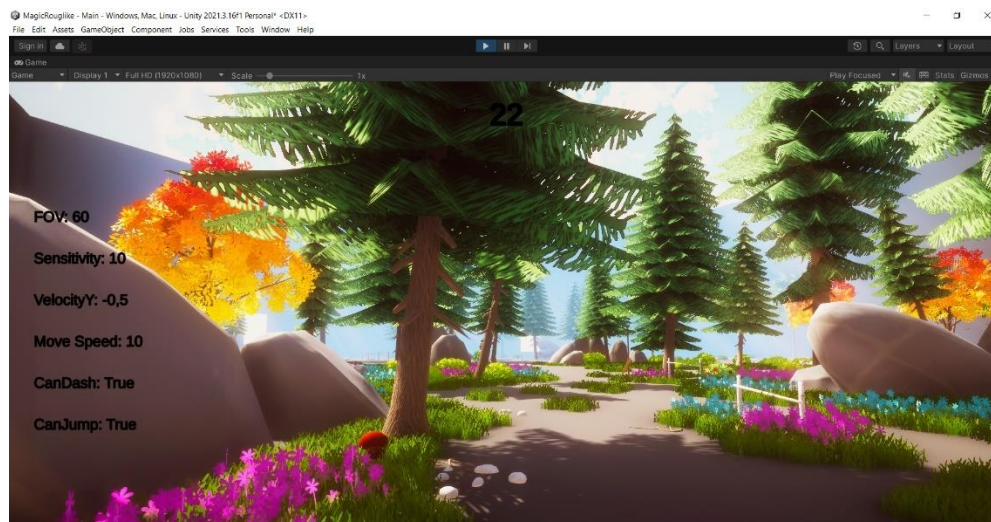


Рисунок 2.11 – Розміщення деталей та атмосферних об'єктів.

Наступним етапом є розміщення деталей та атмосферних об'єктів (рис. 2.11 - 2.12). Встановлено деталі, які допоможуть створити атмосферу фентезійного світу. Важливо врахувати кожен деталь, оскільки вони вирішуються на створення правдоподібного та захоплюючого віртуального світу. Додано рослинність, яка є характерною для фентезійного світу, таку як смереки, ялиці, та інші гірські рослини. Враховано сезонні зміни у рослинності, які можуть впливати на атмосферу (наприклад, ялинковий ліс).

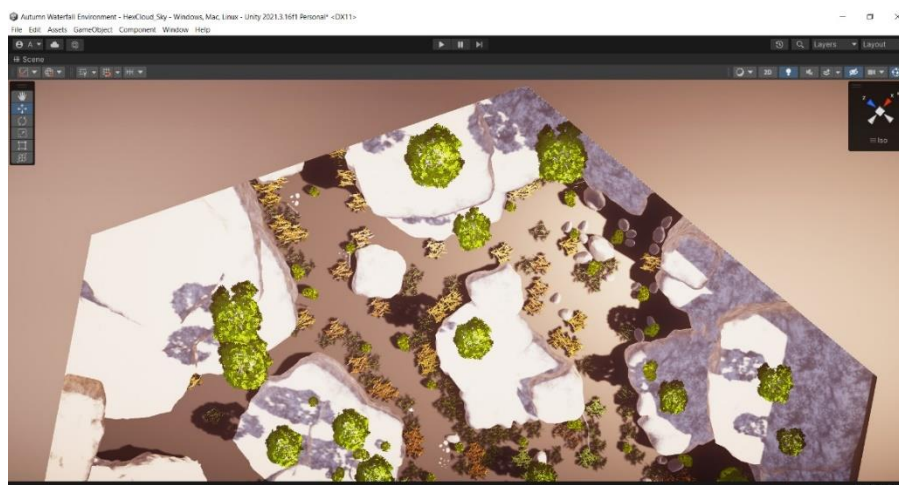


Рисунок 2.12 – Розміщення деталей та атмосферних об'єктів.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		35

Наступним кроком є розробка шляхів та переміщення (рис. 2.13). Вирішення, як гравець буде переміщатися по мапі рівня. Встановлено шляхи, лабіринти, схованки та інші елементи дизайну, що дозволять гравцю відчувати прогресію та досліджувати різні частини рівня. Забезпечено цікавість та різноманітність у переміщенні.

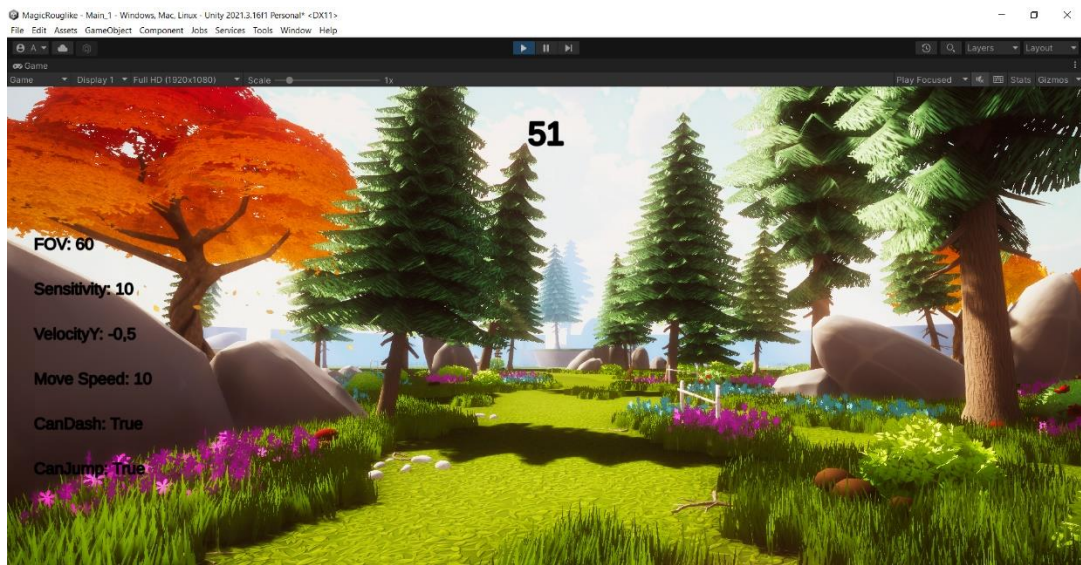


Рисунок 2.13 – Розробка шляхів та переміщення.

Урахування механік Roguelite гри: під час планування мапи рівня у фентезійному світі (рис. 2.14). Необхідно враховувати різноманітність областей і перешкод, які характеризуються випадковим розміщенням, що додає динаміку і виклик гравітації до гри. Шляхи і перешкоди повинні бути різноманітними для кожного геймплею, щоб забезпечити унікальний досвід кожного разу, коли гравець пройде рівень.

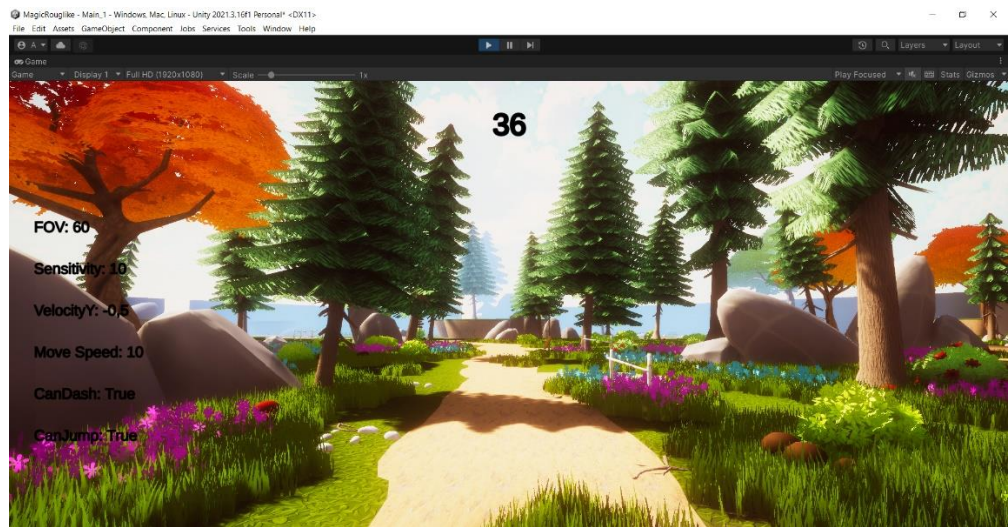


Рисунок 2.14 – Урахування механік Roguelite гри

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				36
Змн.	Арк.	№ докум.	Підпис	Дата		

Крім того, важливо передбачити випадкову генерацію скарбів, ворогів і пасток, що забезпечить непередбачуваність і постійну зацікавленість гравця. Застосування процедурної генерації гарантує, що кожне проходження рівня буде відрізнятися від попереднього, створюючи нові виклики та можливості для дослідження.

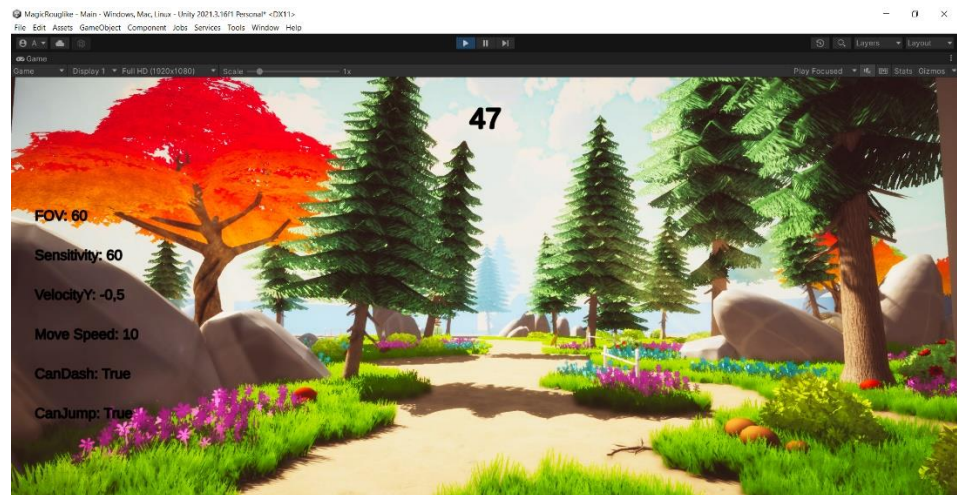


Рисунок 2.15 – Тестування та ітерації.

Наступним кроком є тестування та ітерації (рис. 2.15). Після створення першої версії мапи рівня проведено тестування. Приділено увагу щодо геймплею, потенційні проблеми з взаємодією, атмосферу та загальний досвід гравця. Здійснено необхідні корекції та ітерації, щоб забезпечити оптимальний дизайн рівня.

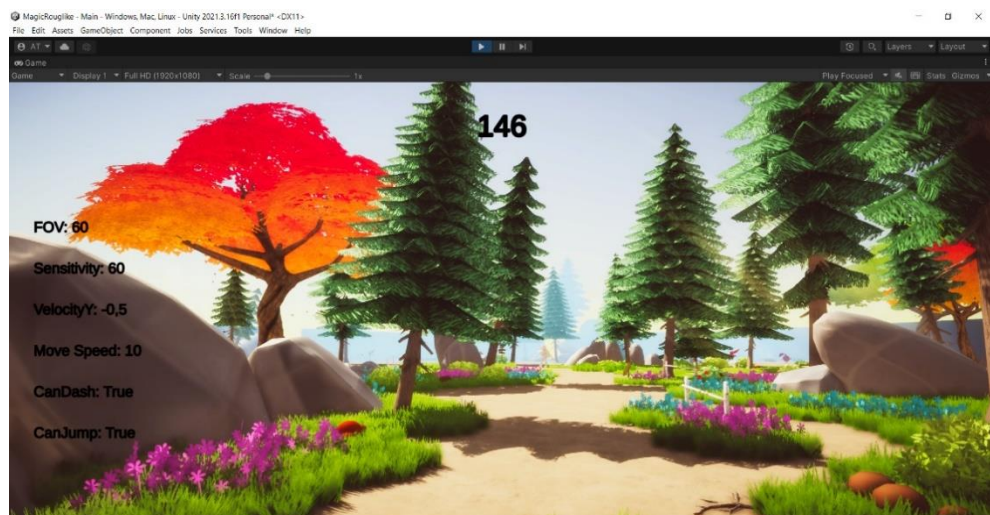


Рисунок 2.16 – Тестування та поліпшення.

Тестування та поліпшення: цей процес планування мапи рівня (рис. 2.16) допомагає виявити помилки, нестабільності та недоліки. Це дозволяє виправити

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		37

проблеми та забезпечити коректну роботу гри перед випуском для гравців. Поліпшення гри може включати в себе оптимізацію продуктивності, виправлення багів та недоліків, додавання нового контенту (такого як монстри, предмети, локації) для розширення геймплею, а також вдосконалення візуальних ефектів для створення більш іммерсивного досвіду гравця.

Робота з освітленням та постпроцесінгом є важливою складовою при розробці інтерактивного 3D світу на Unity. Освітлення (рис. 2.17) допомагає створити атмосферу та настрій, а постпроцесінг дозволяє покращити візуальну якість та ефекти. Ось кілька ключових аспектів роботи з освітленням та постпроцесінгом:

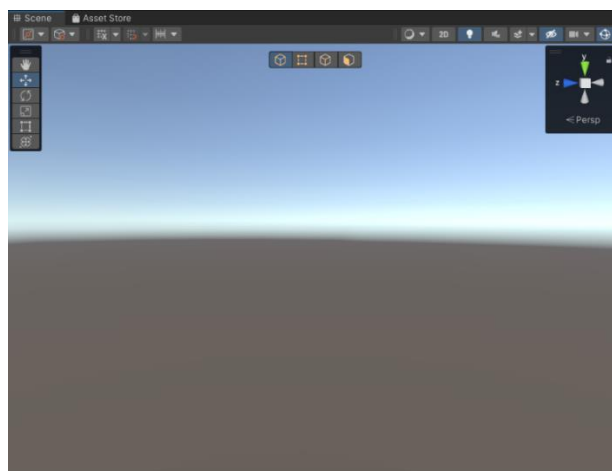


Рисунок 2.17 – Робота з освітленням.

Розміщення джерел світла: встановлення джерел світла (рис. 2.18), таких як точкові, напрямлені або площинні світильники, щоб створити потрібну освітленість у рівні.

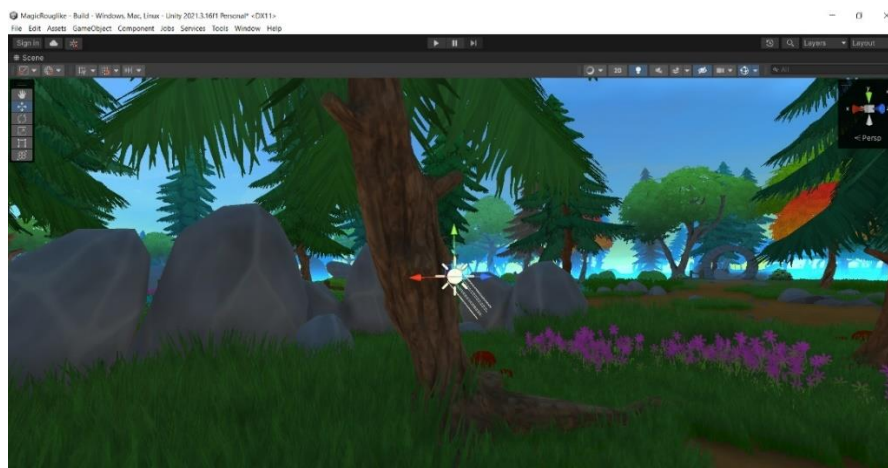


Рисунок 2.18 – Розміщення джерел світла

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		38

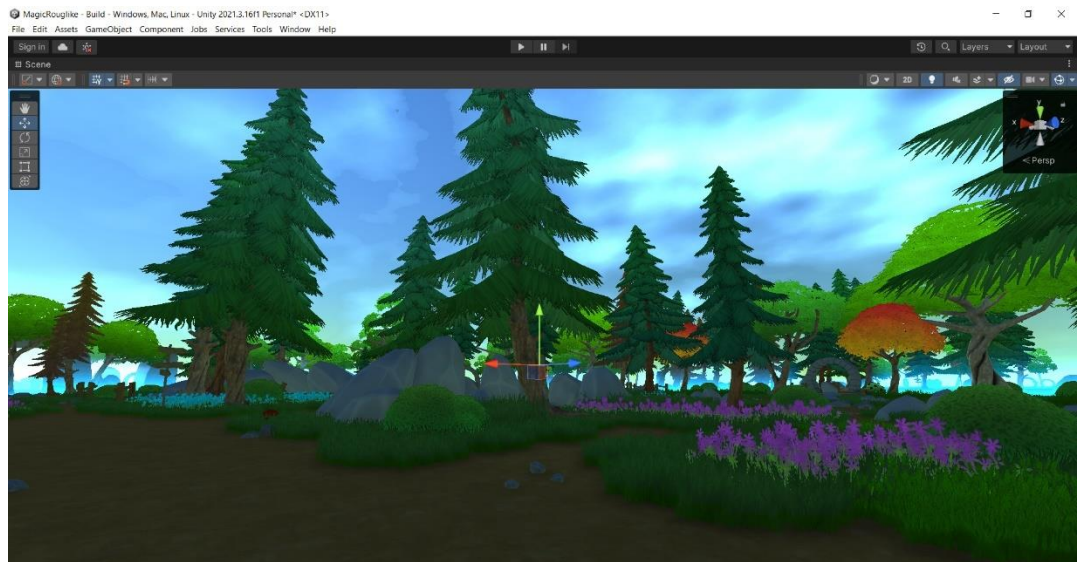


Рисунок 2.19 – Розміщення джерел світла на карті

Типи світла та їх параметри: Unity надає різні типи світла, такі як директорне, точкове, амбієнтне, спот, прожектор тощо. Було визначено потрібний тип світла для кожної сцени або об'єкта, а також налаштовано його параметри, такі як інтенсивність, колір, радіус освітлення тощо (рис. 2.19). Для створення реалістичної та атмосферної освітлення було використано різні комбінації цих типів світла, де кожен тип відповідає за свої особливості в освітленні сцени. Наприклад, директорне світло використовується для симуляції сонячного світла з певного напрямку, тоді як точкове світло створює відчуття джерела світла у вигляді точки.

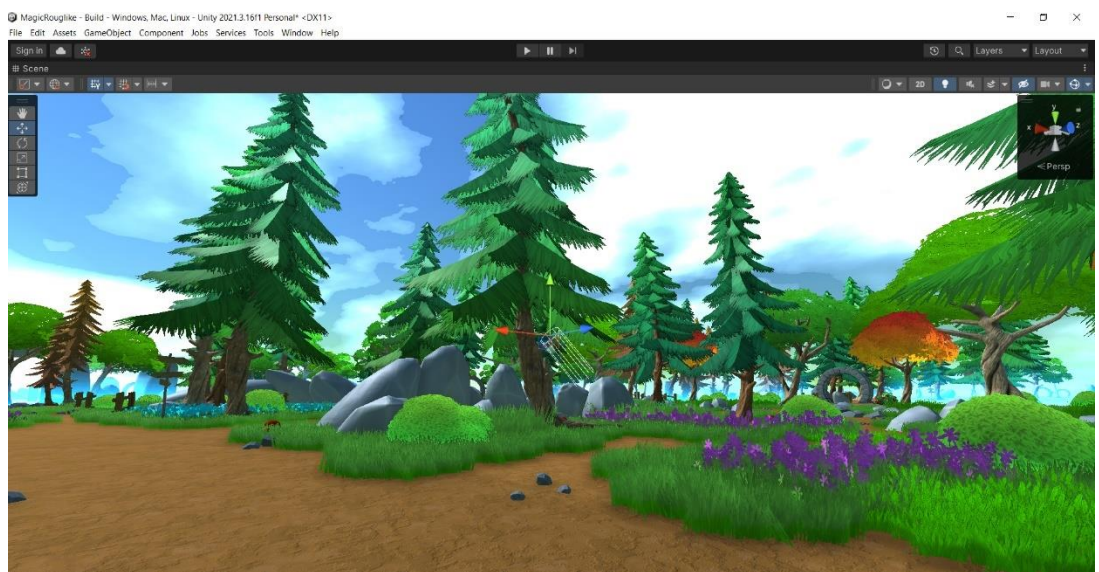


Рисунок 2.20 – Розміщення глобального освітлення без налаштувань

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		39

Реалістичне освітлення: Використано техніки, такі як блики, тіні, розсіяне освітлення, відблиски, щоб створити реалістичний вигляд освітлення для гравця. Враховано матеріали об'єктів та їх взаємодію зі світлом (рис. 2.20 - 21).

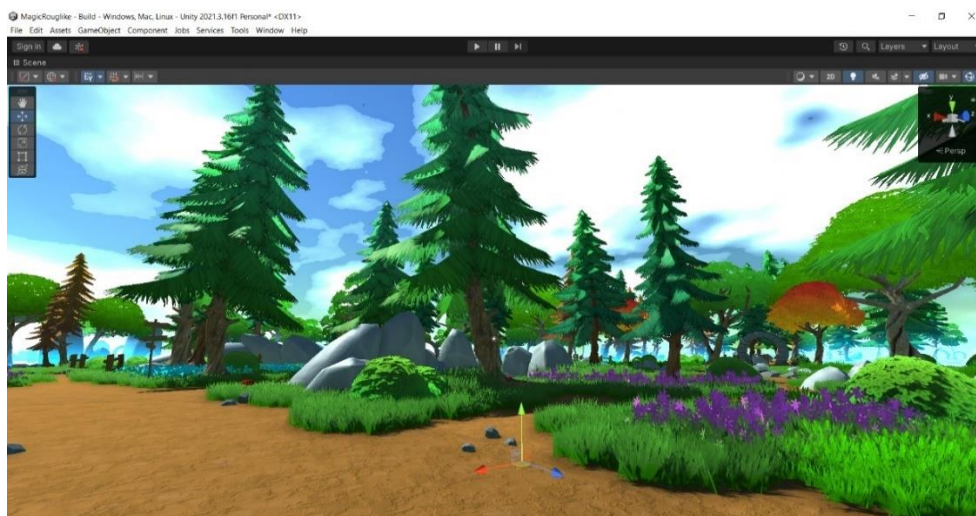


Рисунок 2.21 – Розміщення глобального освітлення на карті

Постпроцесінгові ефекти: Unity має вбудовані постпроцесінгові ефекти, такі як моушн-блюр, DOF (глибина різкості), кольорові фільтри, блики, розсіяння тощо. Використовуючи ці ефекти можливо покращити візуальну якість, настрій та ефекти.

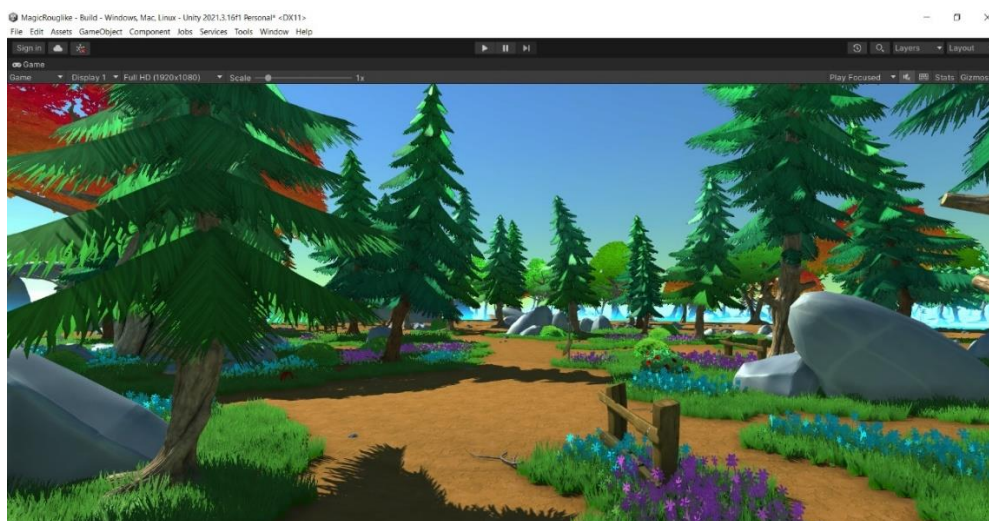


Рисунок 2.22 – Локація без використання постпроцесінга

Параметри постпроцесінгу: налаштовано параметри постпроцесінгу для досягнення бажаних ефектів. Використовано налаштування, такі як інтенсивність ефектів, розмиття, контраст, насиченість тощо, для досягнення потрібного вигляду та настрою (рис. 2.22).

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				40
Змн.	Арк.	№ докум.	Підпис	Дата		

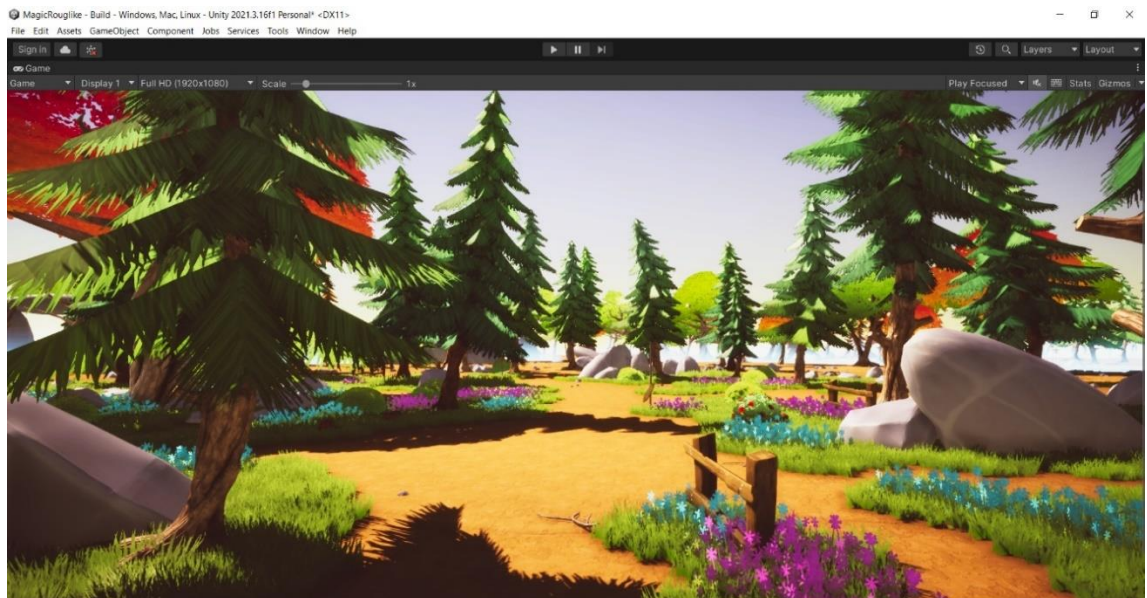


Рисунок 2.23 – Локація з використання постпроцесінга

Тестування та оптимізація: перевірено роботу освітлення та постпроцесінгу на різних пристроях, щоб переконатися, що вони працюють належним чином та не впливають на продуктивність. Оптимізовано налаштування та параметри для досягнення оптимальної продуктивності без втрати візуальної якості (рис. 2.23).

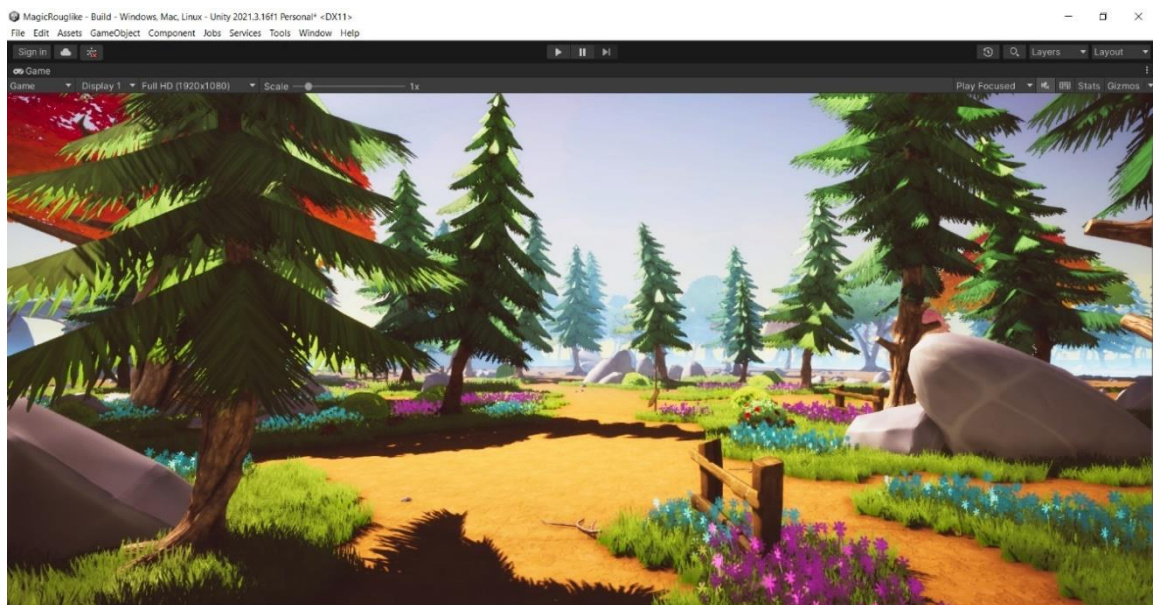


Рисунок 2.24 – Фінальний результат

Робота з освітленням та постпроцесінгом вимагає експериментів, тестування та творчого підходу. Відпрацьовано різні комбінації налаштувань та ефекти, щоб створити бажану атмосферу та візуальний стиль для Roguelite гри на Unity (рис. 2.24).

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		41

2.3 Створення візуальних ефектів для ігрового застосунку

Particle Effects - це потужний інструмент для створення різноманітних візуальних ефектів, таких як вибухи, вогонь, дощ, сніг та багато іншого. Вона ґрунтується на імітації поведінки великої кількості маленьких частинок, що дозволяє досягти реалістичних та динамічних результатів.

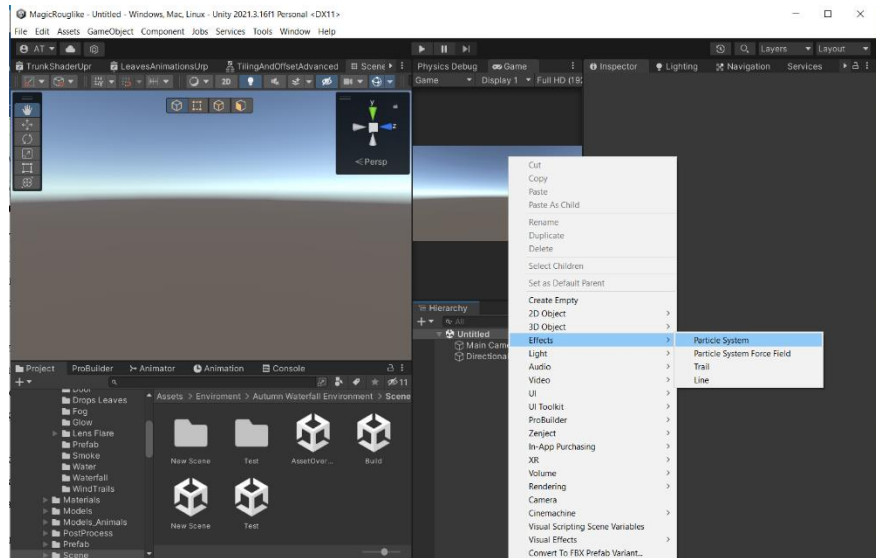


Рисунок 2.25 – Додавання компонента Particle System

Першим кроком є додавання компонента Particle System до об'єкта GameObject (рис. 2.25). Це можна зробити, перетягнувши префаб Particle System з Assets панелі на об'єкт GameObject в Hierarchy панелі, або ж за допомогою меню GameObject > Effects > Particle System або комбінації клавіш Shift + P.

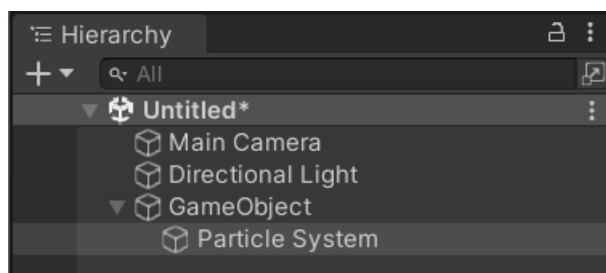


Рисунок 2.26 – Знаходження Particle System в панелі Inspector

Після додавання компонента Particle System він буде знаходитись в Inspector панелі (рис. 2.26). Кожен параметр Particle System може бути налаштований для досягнення потрібного візуального ефекту в грі. Далі, налаштування Particle System розпочинається з визначення (форми області викидання), яка включає в себе різні параметри:

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		42

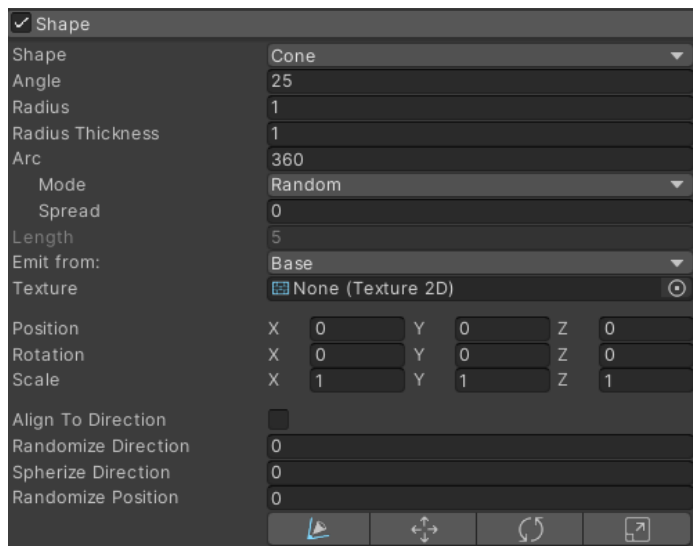


Рисунок 2.27 – Додавання компонента Shape (Форма)

Shape (Форма): спочатку, вибрано форму області викидання частинок. Наприклад, використано сферу для створення кулястого ефекту вибуху. Потім налашвано радіус сфери, щоб визначити область, з якої частинки будуть викидатися (рис. 2.27).

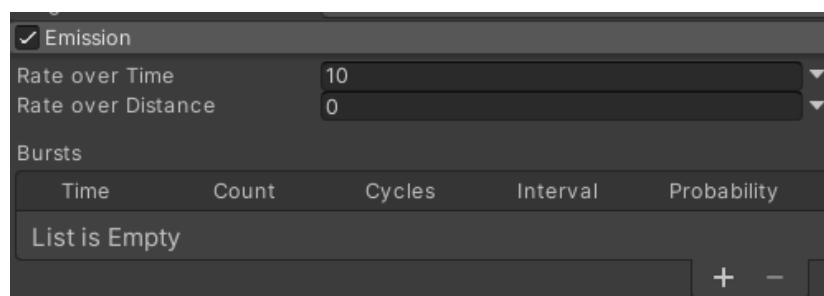


Рисунок 2.28 – Додавання компонента Emission (Викидання):

Emission (Викидання): потім, встановлено кількість та частоту викидання частинок. Наприклад, було встановлено 10 частинок кожні 0 секунди для створення постійного потоку частинок (рис. 2.28). Далі, налаштовано зміну частоти з часом, щоб змінювати інтенсивність ефекту.

Velocity over Lifetime (Швидкість протягом усього життя): далі, встановлено швидкість руху частинок. Було встановлено швидкість 1 м/сек, щоб частинки рухалися від центру сфери в усіх напрямках. Додано випадкову швидкість, щоб додати випадковий рух частинок (рис. 2.29). Крім того, налаштовано змінення швидкості протягом життя частинки, щоб створити більш природний рух, поступово зменшуючи швидкість до нуля.

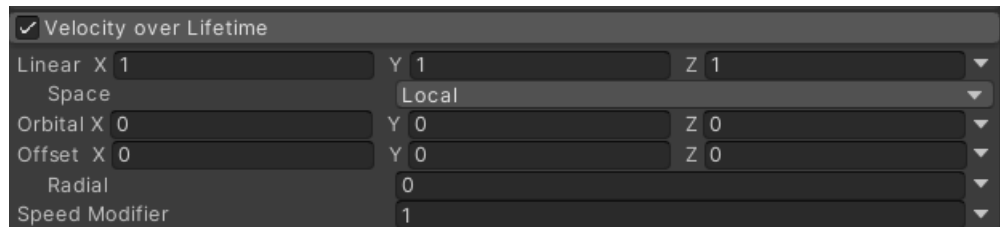


Рисунок 2.29 – Додавання компонента Velocity over Lifetime

Size over Lifetime (Розмір протягом усього життя): цей модуль дозволяє визначити криву, яка диктує, як розмір частинки змінюється протягом її життя (рис. 2.30).

Можна редагувати криву в вікні Inspector. Вісь X представляє нормалізований час життя частинки (0 - створення, 1 - знищення). Вісь Y представляє розмір відносно початкового розміру, встановленого в налаштуваннях системи частинок.

Постійно зростаюча крива змусить частинки рости з часом. Крива у формі дзвона змусить частинки спочатку рости, потім стискатися перед зникненням. Постійно спадна крива змусить частинки стискатися з часом.

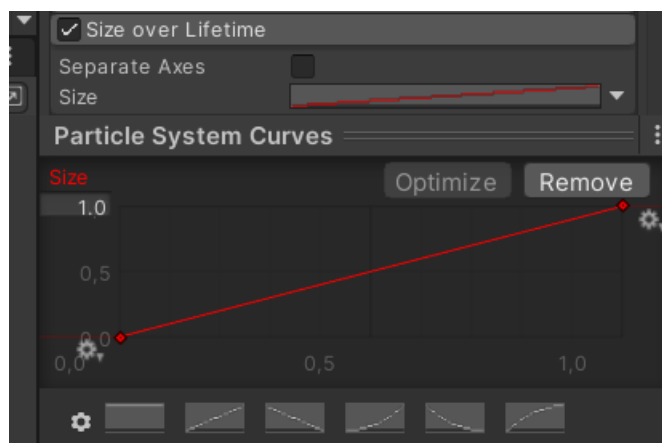


Рисунок 2.30 – Додавання компонента Size over Lifetime

Size by Speed (Розмір за швидкістю): цей модуль визначає зв'язок між швидкістю частинки та її розміром (рис. 2.31). З його допомогою можна налаштувати частинки таким чином, щоб їх розмір змінювався в залежності від швидкості руху. Наприклад, при більшій швидкості частинки можуть ставати більшими, а при зниженні швидкості — меншими. Це додає динаміки та реалістичності ефектам, роблячи їх більш правдоподібними та виразними.

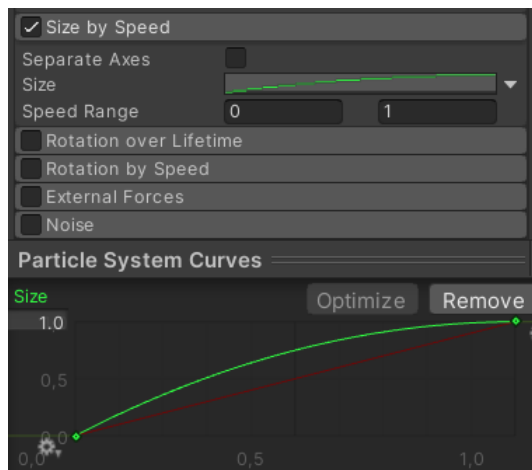


Рисунок 2.31 – Додавання компонента Size by Speed

Size: ця крива визначає, як розмір масштабується залежно від швидкості частинки. Вісь X представляє нормалізовану швидкість частинки (0 - нерухома, 1 - максимальна швидкість). Вісь Y представляє розмір відносно початкового розміру.

Діапазон швидкості: визначає діапазон швидкостей, що враховуються для масштабування розміру. Частинки з швидкостями, що виходять за межі цього діапазону, використовуватимуть кінцеві значення кривої.

Size over Lifetime пропонує більш точний контроль над зміною розміру протягом життя частинки, незалежно від швидкості. Size by Speed динамічно реагує на швидкість частинки. Було створено ефект, де швидші частинки здаються більшими, імітуючи ефект того, що більші об'єкти здаються, що рухаються повільніше.

Випадки використання: Size over Lifetime підходить для створення чітких шаблонів зростання або стискання для частинок. Size by Speed корисний для створення відчуття масштабу на основі руху частинок. Хоча ці модулі працюють незалежно, можна об'єднати їх для отримання складніших ефектів. Наприклад, можна використовувати Size over Lifetime для загальної зміни розміру, а потім використовувати Size by Speed для незначних варіацій розміру на основі швидкості.

Вибір між Size over Lifetime та Size by Speed залежить від бажаного ефекту. Якщо потрібно створити чіткі зміни розміру з часом, потрібно використовувати

Size over Lifetime. Якщо потрібно щоб частинки здавалися більшими на основі їх швидкості, то потрібно використовувати Size by Speed. Експериментування з їх об'єднанням дає можливість отримати більш складні візуальні ефекти.

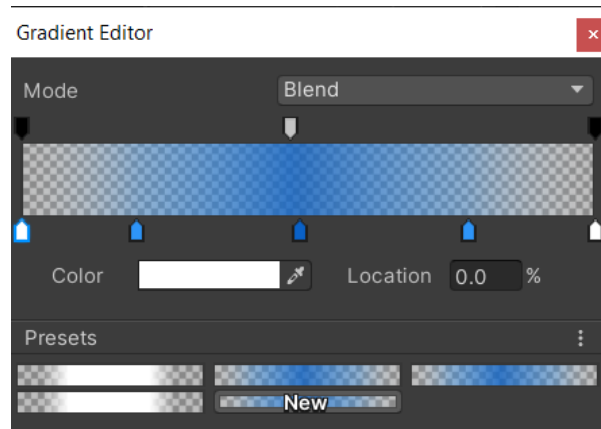


Рисунок 2.32 – Додавання компонента Color over Lifetime

Color over Lifetime (Колір протягом усього життя): Цей модуль дозволяє визначити, як колір частинок змінюється протягом її життя (рис. 2.32).

Можна редагувати криву в вікні Inspector. Вісь X представляє нормалізований час життя частинки (0 - створення, 1 - знищення).

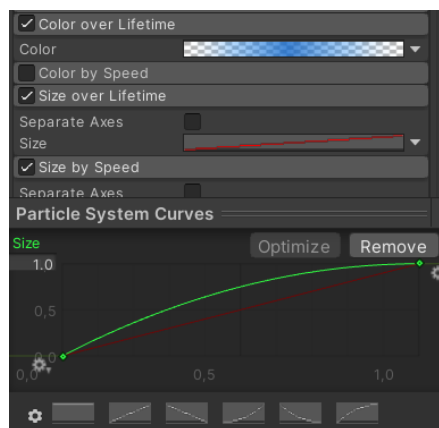


Рисунок 2.33 – Додавання компонента Color by Speed

Color by Speed (Колір за швидкістю): цей модуль визначає зв'язок між швидкістю частинки та її кольором (рис. 2.33).

Color (Колір): ця крива визначає, як колір змінюється залежно від швидкості частинки. Вісь X представляє нормалізовану швидкість частинки (0 - нерухома, 1 - максимальна швидкість). Вісь Y представляє колір, який буде використовуватися для частинки.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		46

Діапазон швидкості: визначає діапазон швидкостей, що враховуються для зміни кольору. Частинки з швидкостями, що виходять за межі цього діапазону, використовуватимуть кінцеві значення кривої. Створено ефект, де швидші частинки мають більш насичений колір, а повільніші - більш тьмянний.

Texture (Текстура): потім, вибрано текстуру для частинок. Наприклад, для світла обрано текстуру пилу або власне зображення для унікального ефекту. Додано вибір з вбудованих текстур або власних зображень для кожної частинки (рис. 2.34).

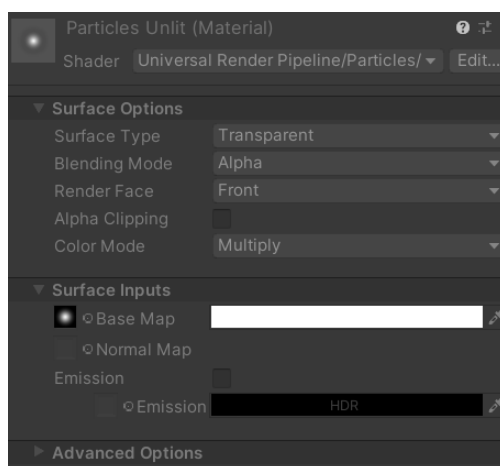


Рисунок 2.34 – Додавання Texture

Noise (Шум): використовується для створення випадкових або псевдовипадкових змін у властивостях частинок, таких як позиція, розмір, кут обертання та інші (рис. 2.35).

Цей процес можна описати наступним кодом:

```
using UnityEngine;

public class MagicLights : MonoBehaviour
{
    public ParticleSystem particles;

    private void Start()
    {
        var noiseModule = particles.noise;

        // Налаштування Noise для Position
        noiseModule.noise.type = NoiseModule.NoiseType.Position;
        noiseModule.noise.strength = 0.5f;
        noiseModule.noise.frequency = 5f;

        // Налаштування Noise для Color
        noiseModule.noise.type = NoiseModule.NoiseType.Color;
        noiseModule.noise.strength = 0.2f;
        noiseModule.noise.frequency = 2f;
    }
}
```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		47

Частота та Амплітуда (Frequency & Amplitude): спочатку, встановлюється частота та амплітуда шуму, які визначають, як часто та наскільки великі будуть зміни. Велика амплітуда може призвести до більших змін у властивостях частинок.

Просторова Швидкість (Spatial Speed): параметр, який визначає швидкість змін у властивостях частинок в просторі. Більша швидкість може призвести до більш швидких змін.

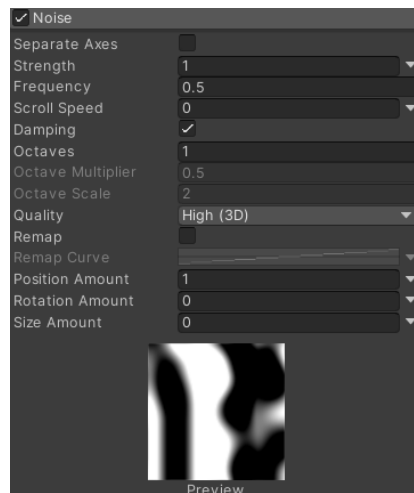


Рисунок 2.35 – Додавання компонента Noise

Часова Швидкість (Temporal Speed): цей параметр визначає швидкість змін у властивостях частинок з часом. Наприклад, можна налаштувати частинки так, щоб їхні властивості змінювалися зі збільшенням часу в грі.

Мінімальне та Максимальне Значення (Min & Max Value): ці параметри обмежують зміни властивостей частинок. Наприклад, якщо встановити мінімальне та максимальне значення розміру, розмір частинок буде змінюватися в межах цих значень.

Форма Шуму (Noise Shape): вибір форми шуму, такого як шум Перліна або Simplex, впливає на те, як генеруються зміни властивостей частинок. Різні форми шуму можуть мати різні характеристики змін.

Додаткові Параметри (Additional Parameters): вони впливають на вигляд шуму, такі як сід шуму, який контролює початковий стан шуму, або масштаб, який визначає масштаб змін.

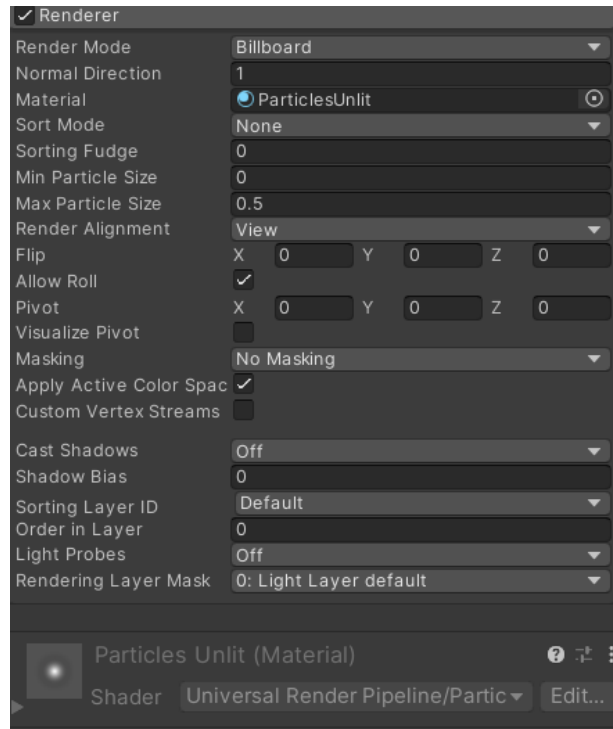


Рисунок 2.36 – Додавання компонента Renderer

В Unity Particle System параметри `Renderer` визначають спосіб візуалізації частинок і включають такі налаштування (рис. 2.36).

Render Mode (Режим візуалізації): спочатку вибирається тип візуалізації частинок, такий як «Billboard» для завжди звернутих до камери частинок, «Stretched Billboard» для текстур, що розтягнуті між початковою та кінцевою позиціями, «Mesh» для використання 3D меші та ін.

Material (Матеріал): потім обирається матеріал, який використовуватиметься для візуалізації частинок. Це може бути вбудований матеріал або створений вами власний.

Trail Material (Матеріал сліду): якщо використовується слід для частинок, то вибирається матеріал для візуалізації цього сліду.

Sorting Fudge (Вирівнювання): параметр, який визначає зміщення для сортування частинок у відображенні, щоб коректно відображати їх на сцені.

Sorting Layer (Шар сортування) і Order in Layer (Порядок у шарі сортування): визначається шар сортування та порядок частинок у цьому шарі для правильного відображення на сцені.

Min Particle Size (Мінімальний розмір частинок) і Max Particle Size (Максимальний розмір частинок): налаштовується мінімальний та максимальний розмір частинок у візуалізації.

Flip (Обертання): параметр, який дозволяє обертати текстуру частинок навколо осі.

Pivot (Центр): визначає точку, від якої відбуватиметься масштабування та обертання частинок.

Billboard (Білборд): частинки завжди обертаються обличчям до камери.

Stretch Billboard: частинки обертаються обличчям до камери, але можуть також розтягуватись уздовж напрямку руху.

Створення God Rays для 3D світу ігрового застосунку "Forbidden Forest" в Unity Particle System.

God rays - це візуальний ефект, який використовується для імітації променів світла, що пробиваються крізь атмосферу або інші об'єкти. Цей ефект часто використовується в іграх та фільмах для створення драматичної атмосфери або підкреслення важливих елементів сцени (рис. 2.37).



Рисунок 2.37 – Створення God Rays в Unity Particle System.

Створення God rays в Unity - це порівняно простий процес, який виконано за допомогою системи Particle System. Завдяки правильним налаштуванням і використанню освітлення, можна створити вражаючий ефект God rays, який додає глибини і атмосферності будь-якій ігровій сцені.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		50

Створено новий Particle System. Переходимо до GameObject > Effects > Particle System або просто натискання клавіш Shift + P, щоб створити новий Particle System (рис. 2.38).

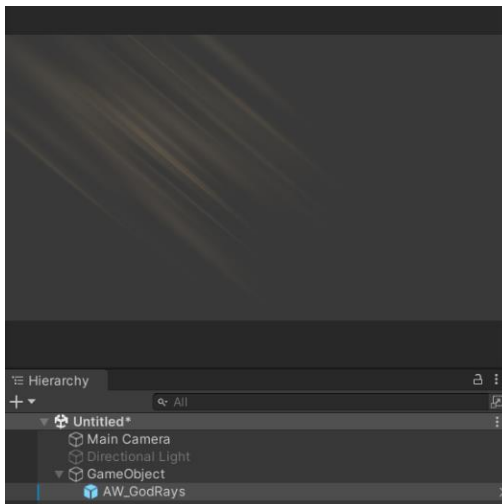


Рисунок 2.38 – Додавання компонента Particle System

Налаштування Particle System (рис. 2.39):

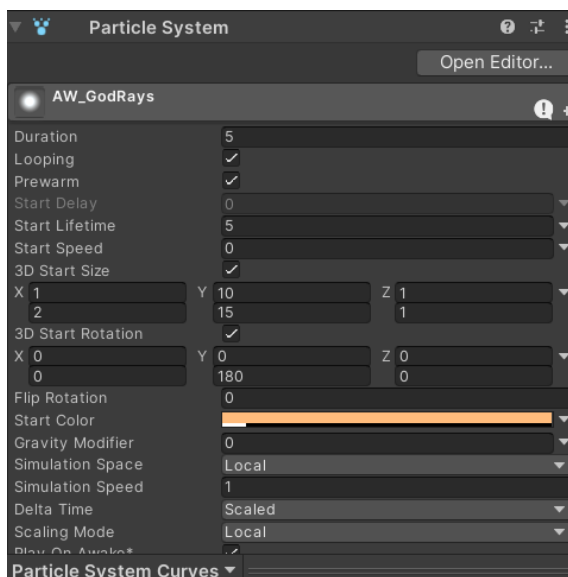


Рисунок 2.39 – Налаштування Particle System

1. Тривалість: 5. Це означає, що кожна частинка буде існувати протягом 5 секунд, перш ніж зникнути.
2. Швидкість: 0. Це означає, що частинки не будуть мати початкової швидкості.
3. Розмір: Y Z1, X1 10. Це означає, що частинки будуть мати початковий розмір по осі Y і Z, який буде визначатися випадковим значенням між 1 і Z1, а початковий розмір по осі X буде дорівнювати 10.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		51

4. Поворот: 2 15 1, 0 YO ZO, 180 0. Це означає, що частинки будуть мати початковий поворот по осі X, який буде визначатися випадковим значенням між 2 і 15, початковий поворот по осі Y буде дорівнювати 0, а початковий поворот по осі Z буде дорівнювати 180.
5. Колір: Білий. Це означає, що частинки будуть мати початковий білий колір.
6. Гравітаційний модифікатор: Local 0. Це означає, що на частинки буде впливати сила тяжіння, але сила тяжіння буде розраховуватися локально, а не глобально.
7. Швидкість симуляції: Scaled 1. Це означає, що швидкість симуляції буде масштабуватися відповідно до частоти кадрів.

Додавання Renderer: Переходимо до Component> Renderers> Sprite Renderer або Mesh Renderer. Це додає візуальний компонент до Particle System (рис. 2.40).

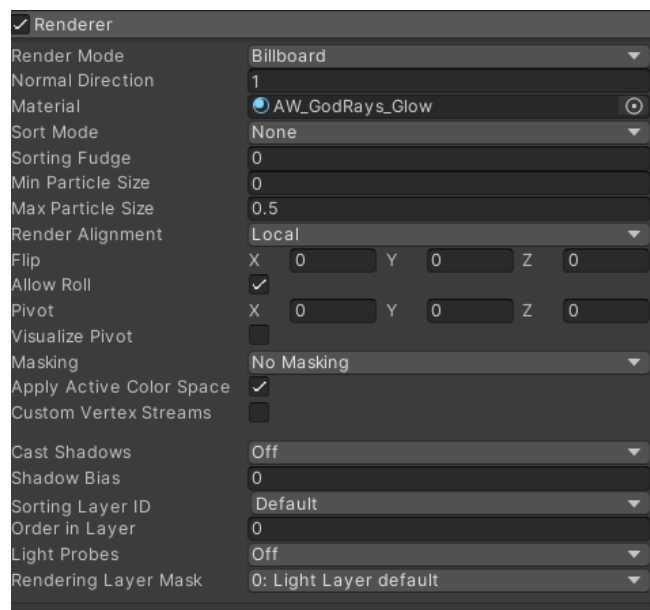


Рисунок 2.40 – Налаштування Renderer

Далі вибираємо матеріал і встановлюємо Additive або Soft Additive режим блендингу. Це дозволить променям світла пробиватися через інші об'єкти (рис. 2.41). Additive блендинг додає кольори частинок до кольорів фону, створюючи яскраві та насичені світлові промені.

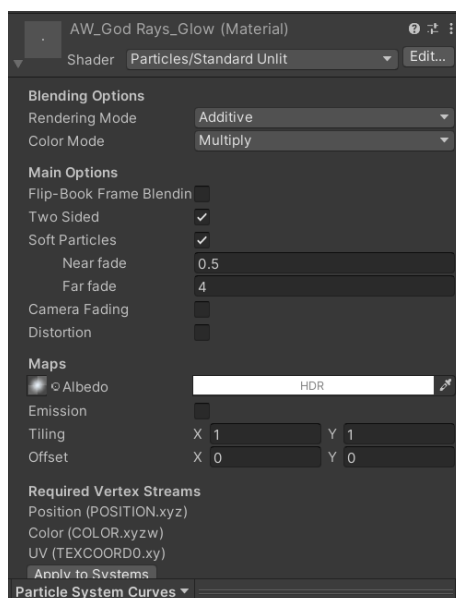


Рисунок 2.41 – Налаштування параметрів матеріалу

Додавання параметра Maps:

1. Albedo: HDR - Основна текстура матеріалу.
2. Emission: Вимкнено - Емісійна карта для створення світла від частинок.
3. Tiling: X 1, Y 1 - Плитка текстури.
4. Offset: X 0, Y 0 - Зсув текстури.

Додавання Noise: Переходимо до Component > Add Component > Noise. Це додасть ефект розмиття та ореолу до променів світла (рис. 2.42).

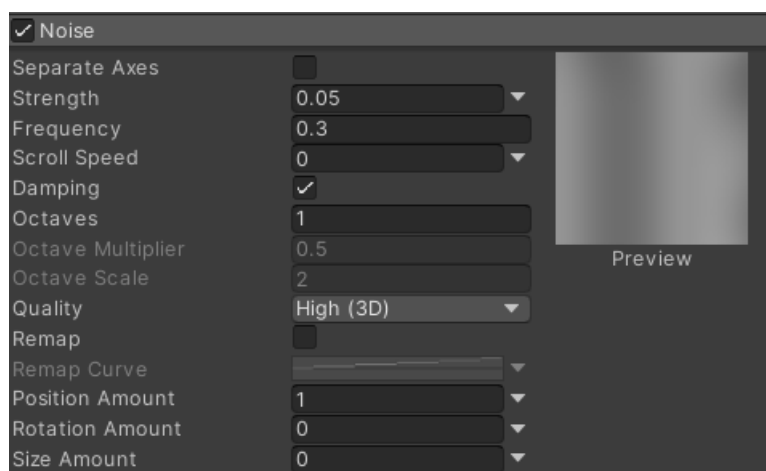


Рисунок 2.42 – Додавання компонента Noise

- Strength: контролює інтенсивність шумового ефекту.
- Frequency: визначає частоту повторення шумового шаблону.
- Damping: контролює швидкість згасання шуму з часом.

- Octaves: контролює кількість шарів шуму, які використовуються для створення загального ефекту.
- Quality (Якість): контролює якість шумового ефекту. Шум вищої якості виглядатиме реалістичніше, але його відтворення буде дорожчим.

Додавання Color over Lifetime: Переходимо до Component > Add Component > Noise. Це додасть ефект розмиття та ореолу до променів світла (рис. 2.43).

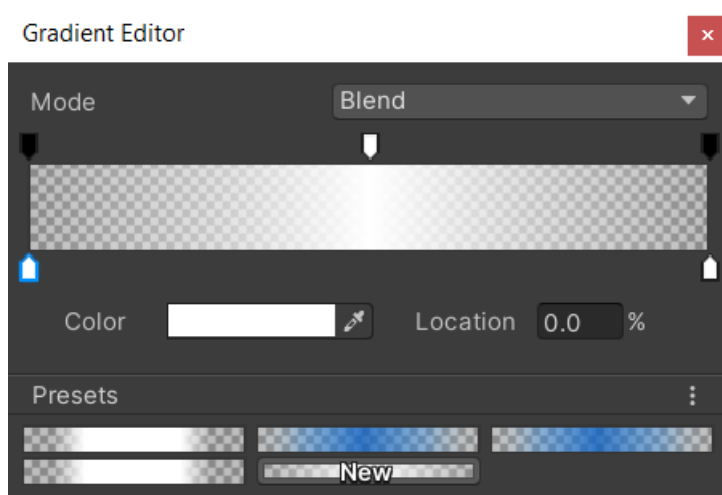


Рисунок 2.43 – Налаштування параметрів Color over Lifetime

2.4 Реалізація програмного коду для створення шейдерів

Shader Graph - це візуальний інструмент для створення шейдерів в Unity, який дозволяє розробникам без глибоких знань мови шейдерів (HLSL або Cg) створювати складні та красиві візуальні ефекти. Замість написання коду шейдера, Shader Graph використовує графічний інтерфейс з нодами, де кожна нода відповідає за певну операцію або функцію.

Переваги використання Shader Graph:

1. Простота використання: Shader Graph не потребує знань мови шейдерів, роблячи його доступним для ширшого кола розробників.
2. Інтуїтивно зрозумілий інтерфейс: Візуальне представлення шейдерів за допомогою нод робить їх більш зрозумілими та легкими для редагування.

3. Швидке прототипування: Shader Graph дозволяє швидко створювати та тестувати різні візуальні ефекти, економлячи час розробки.
4. Гнучкість: Shader Graph пропонує широкий спектр нод та функцій, що дає можливість створювати складні та реалістичні шейдери.
5. Візуалізація: Shader Graph дозволяє візуалізувати результат шейдера в режимі реального часу, що полегшує його налаштування.

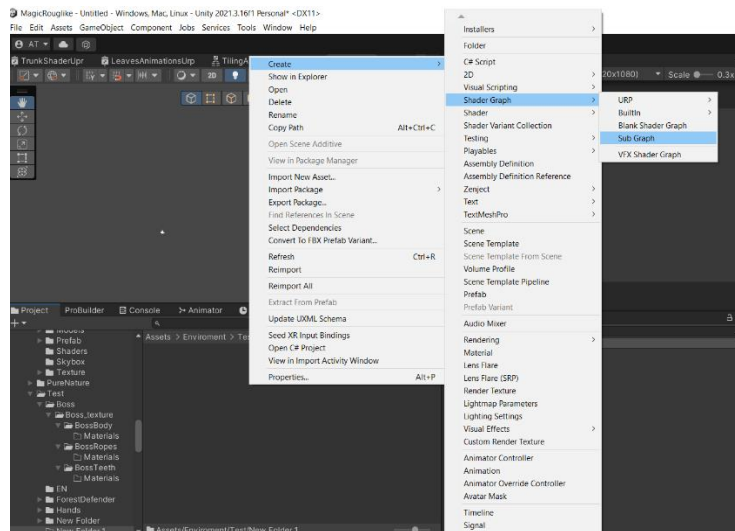


Рисунок 2.44 – Додавання Shader Graph

Створення шейдера починається зі створення нового матеріалу в Unity та вибираємо Shader Graph як тип шейдера для цього матеріалу (рис. 2.44). Для цього потрібно перейти в Create меню та обрати Shader. Обираємо PBR Graph для створення шейдера фізично-коректного рендерингу (PBR), або Unlit Graph для створення простого шейдера без освітлення. Називаємо свій Shader Graph та зберігаємо його в потрібній папці.

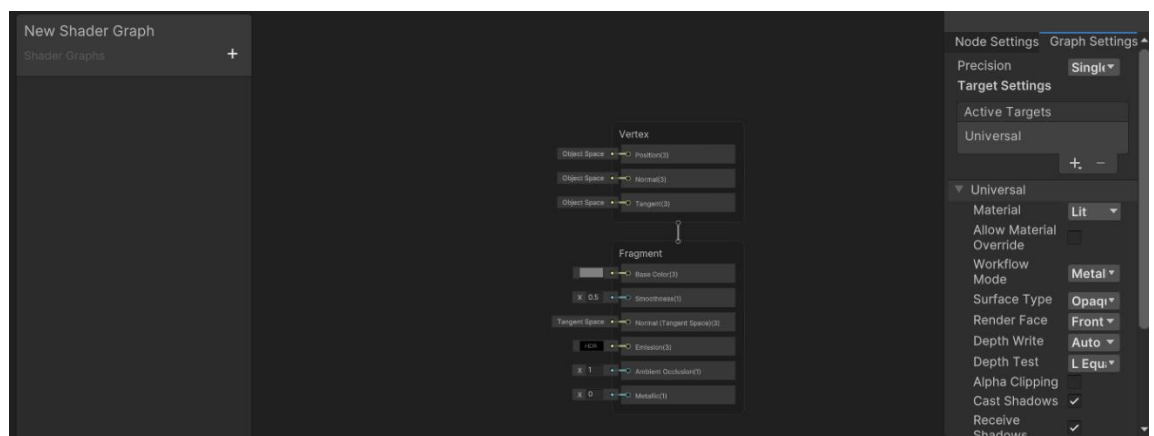


Рисунок 2.45 – Вікно в Graph View

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		
						55

Перетягуємо ноду Color з панелі Nodes на Graph View (рис. 2.45). Ця нода буде використовуватися для встановлення нового кольору з якою ми будемо змінювати колір. Додаємо вузли кольорів, натиснувши правою кнопкою миші на графіці і вибравши «Create Node» (Створити вузол) > «Input» (Введення) > «Color» (Колір).

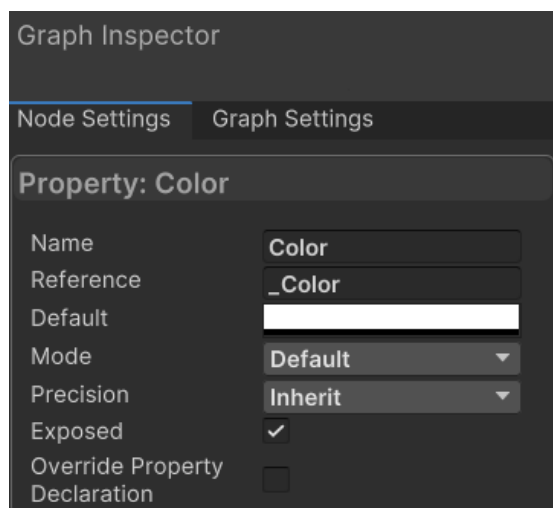


Рисунок 2.46 – Налаштування властивостей Color

Налаштування властивостей (рис. 2.46):

Змінюємо значення Color, щоб встановити бажаний колір. Можна використовувати Color Picker, щоб вибрати колір візуально. Далі, перетягаємо ноди з панелі Nodes на Graph View.

Існує багато типів нод, таких як ноди для математичних операцій, доступу до текстур, генерування освітлення та ін. Використовуємо пошук, щоб швидко знайти потрібні ноди. Це дозволяє зручно організувати робоче середовище та швидко знаходити необхідні інструменти для реалізації різноманітних ефектів і операцій в графічному редакторі.

Потім підключаємо вузли разом, щоб створити бажаний ефект. Наприклад, вузол «Texture 2D» до вузла «Albedo» для встановлення текстури для частинок, а вузол «Noise» з'єднаємо до вузла "Position" для додавання шуму до руху частинок.

Підключаємо вузли кольорів до основних вузлів шейдера. Наприклад, підключаємо вузол колірного введення до вузла «Base Color» (Основний колір) в графіку PBR шейдера (рис. 2.47).

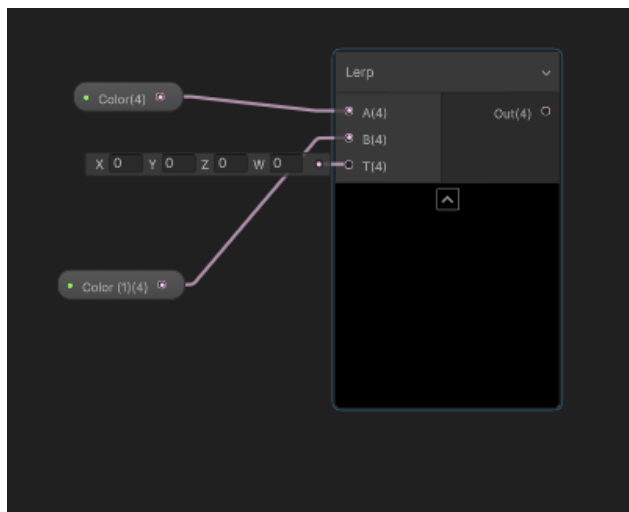


Рисунок 2.47 – Підключення вузлів в Shader Graph

Для того щоб з'єднати ноди потрібно:

1. Вибрати вихід одного нода.
2. Перетягнути лінію до входу іншого нода.
3. З'єднання визначають потік даних в шейдері.

В Shader Graph можна додавати різні вузли, які представляють різні ефекти шейдера. Наприклад, вузол «Texture 2D» для вибору текстури, «Color» для вибору кольору частинок, «Noise» для додавання шуму до руху частинок і т.д.

Shader Graph також дозволяє додавати додаткові ефекти, такі як освітлення, зміна кольорів в залежності від життєвого циклу частинки, різні взаємодії з середовищем та інші ефекти за допомогою відповідних вузлів.

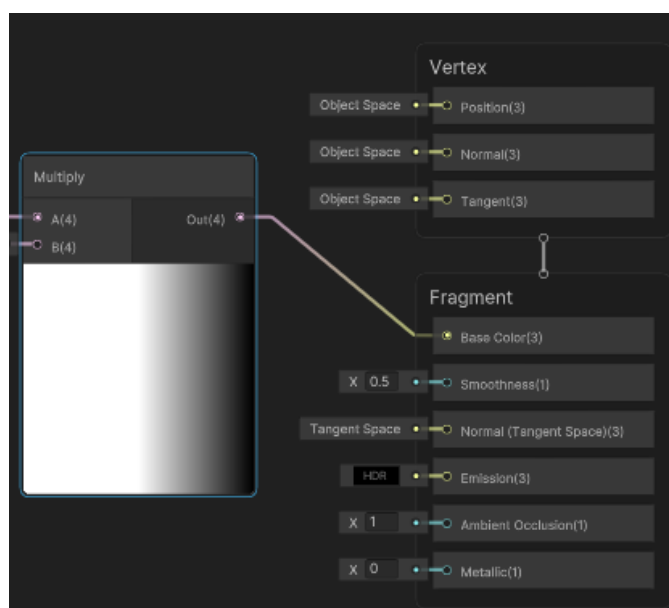


Рисунок 2.48 – Додавання ноди Multiply

Далі додаємо ноду для зміни кольорів. Для цього потрібно:

Перетягнути ноду Multiply з панелі Nodes на Graph View (рис. 2.48). З'єднати вихід RGB ноди Sample Texture з входом А ноди Multiply. Цей зв'язок передає колір текстури, який буде змінюватися, до ноди Multiply. З'єднати вихід Color ноди Set Color з входом В ноди Multiply. Цей зв'язок передає колір, який буде використовуватися для множення на колір текстури. Ця конфігурація дозволить змінювати колір за допомогою множення.

Налаштовуємо властивості ноди Multiply, двічі клацаємо на ноді Multiply, щоб відкрити її Properties Panel, міняємо значення А для контролю інтенсивності зміни кольору. Значення 1.0 буде зберігати оригінальний колір, а значення менше 1.0 буде затемнювати його.

Параметр А контролює перший множник. У цьому випадку він буде використовуватися для множення кольору текстури.

Параметр В контролює другий множник. У цьому випадку він буде використовуватися для множення кольору, заданого нодою Set Color.

Коли два кольори множаться, кожен компонент RGB (червоний, зелений, синій) множиться окремо.

Наприклад, якщо колір текстури RGB (0.5, 0.7, 0.3) множиться на колір RGB (2, 1, 0.5), то результат буде RGB (1, 0.7, 0.15).

Це призведе до затемнення червоного та зеленого компонентів текстури, а синій компонент стане тьмянішим.

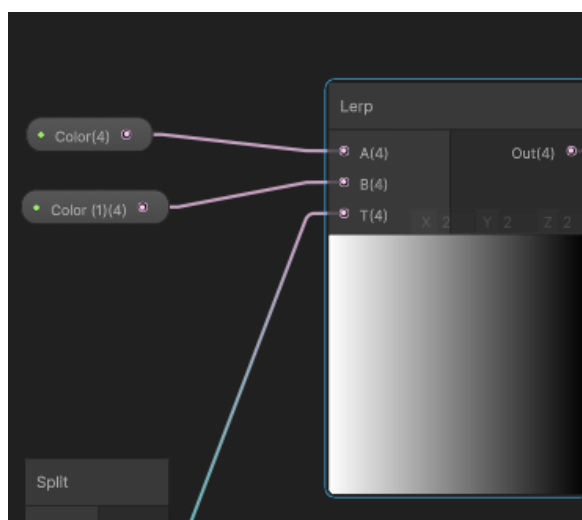


Рисунок 2.49 – Додавання ноди Lerp

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		58

Перетягуємо ноду Lerp з панелі Nodes на Graph View (рис. 2.49). Ця нода буде використовуватися для плавного переходу між двома кольорами, що дозволить створювати більш плавну інверсію.

З'єднуємо RGB ноди Sample Texture з входом А ноди Lerp. Цей зв'язок передає колір текстури, який буде інвертовано, до ноди Lerp. З'єднаємо фіксований вихід 1.0 з входом В ноди Lerp. Ця взаємодія буде використовуватися як базовий колір для інверсії. З'єднаємо вихід Color ноди Invert з входом Т ноди Lerp. Воно передає інвертований колір текстури до ноди Lerp.

Налаштування властивостей ноди Lerp:

Двічі клацаємо на ноді Lerp щоб відкрити її Properties Panel.

Amount: Цей параметр контролює плавність переходу між двома кольорами. Значення 0.0 буде використовувати базовий колір (1.0), а значення 1.0 буде застосовувати інвертований колір.

Можна налаштувати цей параметр, щоб отримати бажану плавність інверсії.

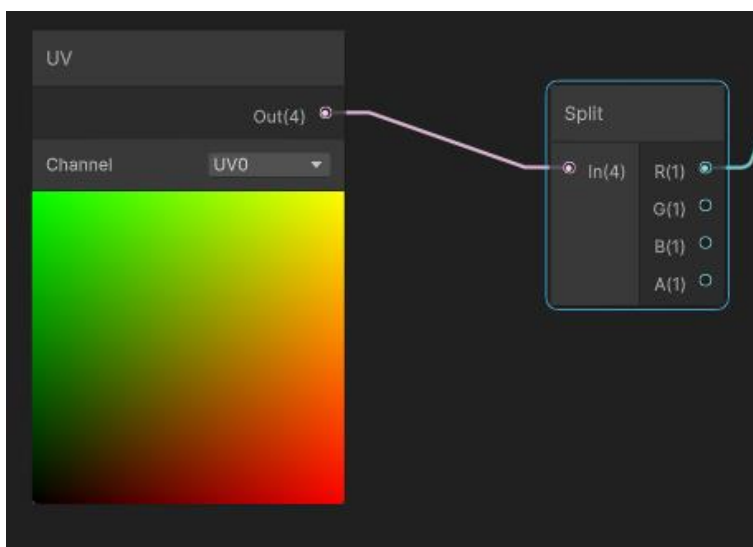


Рисунок 2.50 – Додавання ноди UV

Перетягаємо ноду uv з панелі Nodes на Graph View (рис. 2.50). Ця нода генерує координати текстури (UV) для поточного пікселя. UV-координати - це значення між 0.0 і 1.0, які визначають, де на текстурі знаходиться піксель. Перетягаємо ноду Split з панелі Nodes на Graph View. Ця нода розділяє вектор

UV на чотири окремі компоненти: R, G, B та A. З'єднаємо вихід ноди uv з входом ноди Split. Це передасть UV-координати до ноди Split для розділення.

Можна використовувати виходи R, G, B та A ноди Split для різних цілей. З'єднаємо один або декілька виходів ноди Split з входом ноди Sample Texture. Це дозволить використовувати UV-координати для вибірки кольорів з текстури динамічно. Наприклад, можна використовувати компонент R для вибору червоного каналу текстури, а компонент G для вибору зеленого каналу.

Дозволяється використовувати компоненти UV для створення різних візерунків на основі їх значень. Наприклад компонент G для створення смугастого візерунка, де значення 0.0 відповідає чорній смугі, а значення 1.0 відповідає білій смугі.

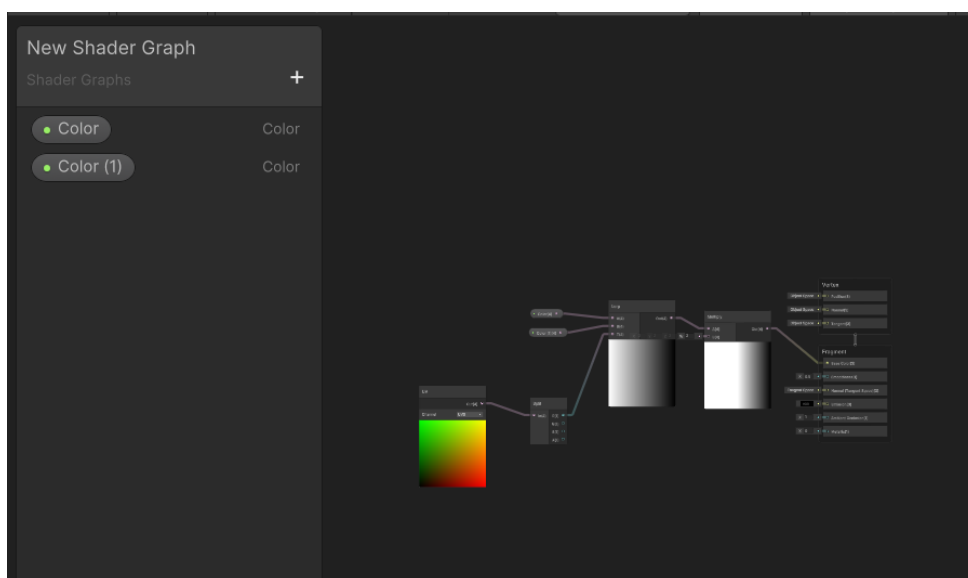


Рисунок 2.51 – Результат підключання вузлів Multiply та Lerp

Через використання ноди Multiply з фіксованим значенням, таким як 0.5, то результат буде затемнення текстури на 50%. Використання ноди Lerp з параметром Amount, який змінюється з часом, то він дає результат з плавним переходом від оригінального кольору текстури до чорного (рис. 2.51).

Використовуючи Set Color з фіксованим кольором, таким як червоний, то результат буде зміною відтінку текстури на червоний.

Натискаємо кнопку Play в Inspector панелі (рис. 2.52).

Shader Graph буде візуалізувати результат шейдера в режимі реального часу. Можна обертати та масштабувати камеру, щоб покращити результат.

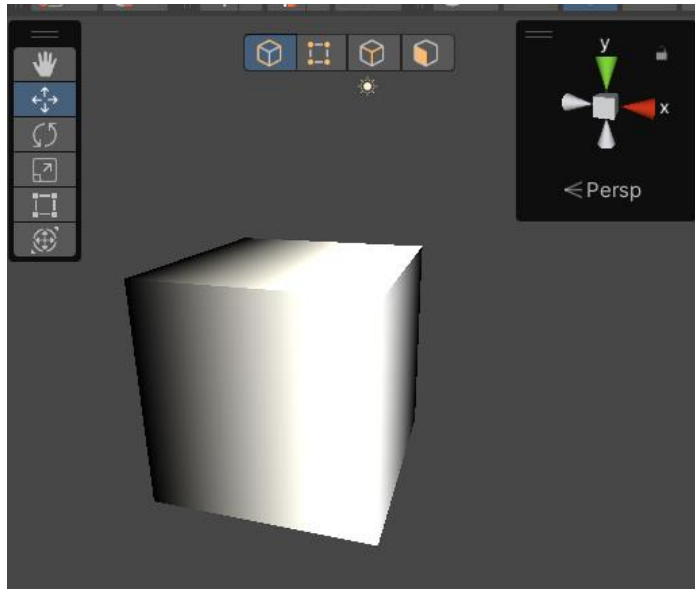


Рисунок 2.52 – Результат використання шейдера на сцені.

Після створення шейдера у Shader Graph, застосовуємо його до об'єктів у нашій сцені, щоб побачити результати в режимі реального часу.

Збереження та використання Shader Graph:

Збереження та використання Shader Graph може бути досить простим процесом. Після створення графа шейдерів і задання всіх необхідних параметрів, ми зберігаємо наш граф Shader Graph, щоб мати можливість повторно використовувати його у майбутньому. Для цього переходимо до панелі Materials, де ми можемо створити новий матеріал або вибрати існуючий. При цьому обираємо збережений раніше граф Shader Graph для використання в обраному матеріалі. Створюємо новий матеріал або обираємо існуючий.

У Shader dropdown меню, знаходимо та обираємо «ColorChangeShader». Можна налаштувати властивості матеріалу, щоб керувати Shader Graph, наприклад, змінювати колір, інтенсивність зміни або інверсії (рис. 2.53).

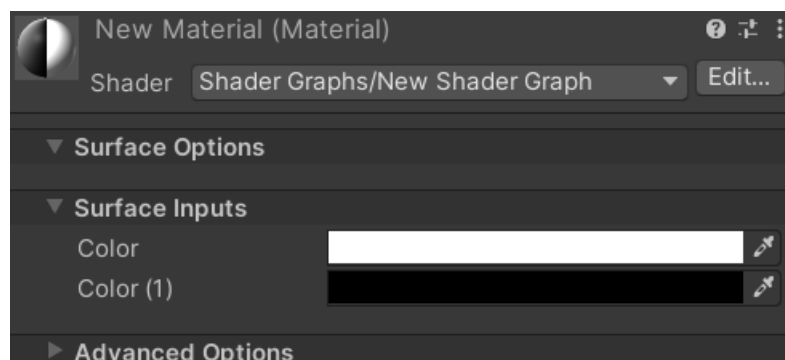


Рисунок 2.53 – Налаштування Шейдера.

В результаті отримано шейдер з можливістю зміни кольорів на заданому об'єкті на сцені.

Приклад використання зміна кольорів на прикладі створеного шейдера для хмар для ігрового застосунку "Forbidden Forest" (рис. 2.54).



Рисунок 2.54 – Шейдер хмар

Хмари в ігровому застосунку "Forbidden Forest" можуть динамічно змінювати свій колір, створюючи більш реалістичну та атмосферу (рис. 2.55).



Рисунок 2.55 – Результат використання шейдера для зміни кольору хмар
Переваги:

- Цей метод дозволяє динамічно змінювати кольори хмар без необхідності писати код.
- Shader Graph візуально простий у використанні, що робить його доступним для розробників без досвіду роботи з шейдерами.
- Це може покращити візуальну якість ігрового застосунку "Forbidden Forest".

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		62

Щоб програма ініціювала використовується наступний код:

```
Shader "Custom/NewSurfaceShader"
{
    Properties
    {
        _Color ("Color", Color) = (1,1,1,1)
        _MainTex ("Albedo (RGB)", 2D) = "white" {}
        _Glossiness ("Smoothness", Range(0,1)) = 0.5
        _Metallic ("Metallic", Range(0,1)) = 0.0
    }
    SubShader
    {
        Tags { "RenderType"="Opaque" }
        LOD 200

        CGPROGRAM
        // Physically based Standard lighting model, and enable shadows on all light
types
        #pragma surface surf Standard fullforwardshadows

        // Use shader model 3.0 target, to get nicer looking lighting
        #pragma target 3.0

        sampler2D _MainTex;

        struct Input
        {
            float2 uv_MainTex;
        };

        half _Glossiness;
        half _Metallic;
        fixed4 _Color;

        // Add instancing support for this shader. You need to check 'Enable Instancing'
on materials that use the shader.
        // See https://docs.unity3d.com/Manual/GPUInstancing.html for more information
about instancing.
        // #pragma instancing_options assumeuniformscaling
        UNITY_INSTANCING_BUFFER_START(Props)
            // put more per-instance properties here
        UNITY_INSTANCING_BUFFER_END(Props)

        void surf (Input IN, inout SurfaceOutputStandard o)
        {
            // Albedo comes from a texture tinted by color
            fixed4 c = tex2D (_MainTex, IN.uv_MainTex) * _Color;
            o.Albedo = c.rgb;
            // Metallic and smoothness come from slider variables
            o.Metallic = _Metallic;
            o.Smoothness = _Glossiness;
            o.Alpha = c.a;
        }
        ENDCG
    }
    FallBack "Diffuse"
}
```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		63

Висновки до другого розділу

Було продемонстровано аналіз процесу розробки концепції інтерактивного 3D світу для ігрового застосунку через діаграми, що ілюструють цей процес. Детально описано кроки розробки локації та створення візуальних ефектів для підвищення реалістичності ігрового досвіду. Розроблено власний шейдер, який значно покращив візуалізацію ігрового застосунку.

Додатково, проведено аналіз ефективності використання ресурсів під час розробки, що дозволило оптимізувати процес та забезпечити високу продуктивність гри.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		64

РОЗДІЛ 3. ОПИС РОБОТИ З ІГРОВИМ ЗАСТОСУНКУ ТА ЙОГО ТЕСТУВАННЯ

3.1 Опис роботи з ігровим додатком

Рівень переносить гравця в фентезійний світ, де кожен куточок наповнений чарами та пригодами. Основною локацією є трактир, який поділений на кілька різних зон. Деталі добре промальовані, а освітлення ефективно створює теплу та приємну атмосферу. Трактир, знаходиться в середньовічному фентезійному світі. Трактир прикрашений різноманітними предметами, що надають йому сільського та середньовічного вигляду (рис. 3.1).

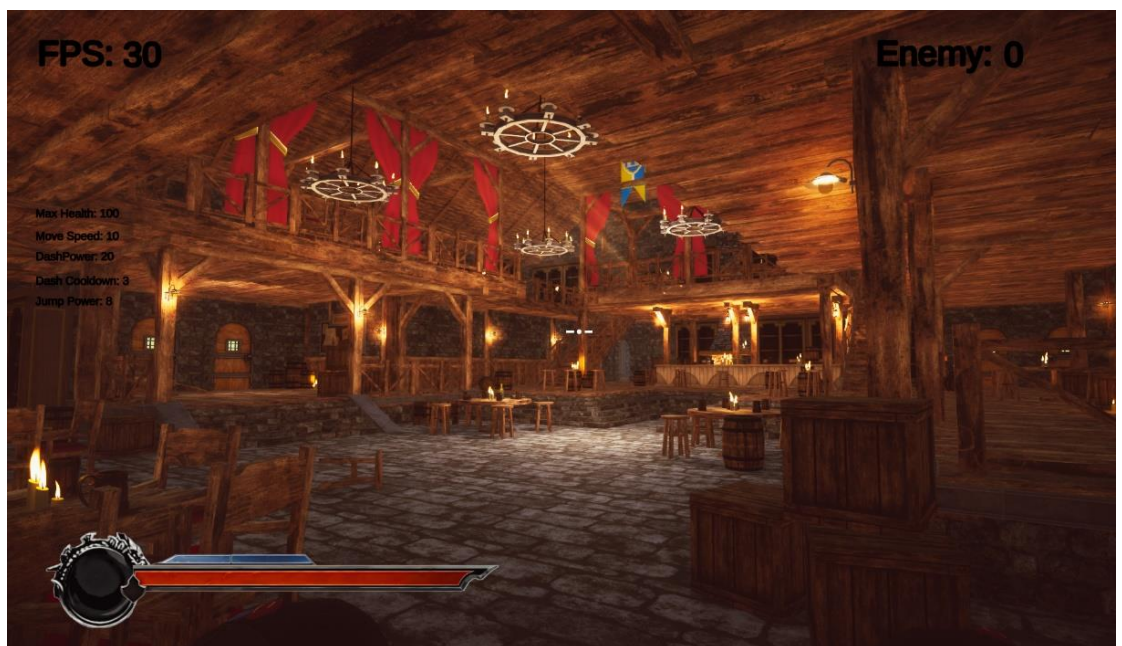


Рисунок 3.1 – Початок рівня

У передній частині розташований просторий бар з кількома високими табуретами, де відвідувачі можуть замовити напої. За барною стійкою можна побачити полиці, заставлені пляшками вина та іншими алкогольними напоями, які створюють атмосферу затишку та гостинності.

У центрі трактира розташовані численні столи та стільці, де відвідувачі можуть насолоджуватися їжею та напоями, обговорюючи свої пригоди. Вдалині, у задній частині трактира, видніється великий камін, що випромінює тепле світло, запрошуючи гостей зігрітися після довгого дня (рис. 3.2).

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		65

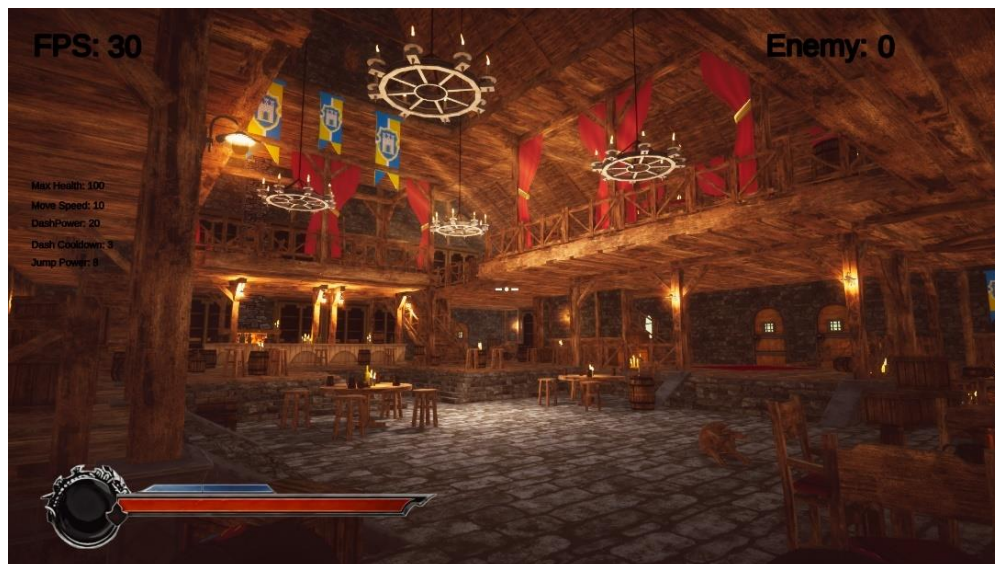


Рисунок 3.2 – Трактир

Декор трактира доповнюють різноманітні предмети, що створюють відчуття стародавнього середньовічного світу. На стінах висять зброя, обладунки, а з стелі звисають різнокольорові прапори та стяги. Кожен куточок цього місця наповнений атмосферою пригод та стародавніх історій (рис. 3.3).

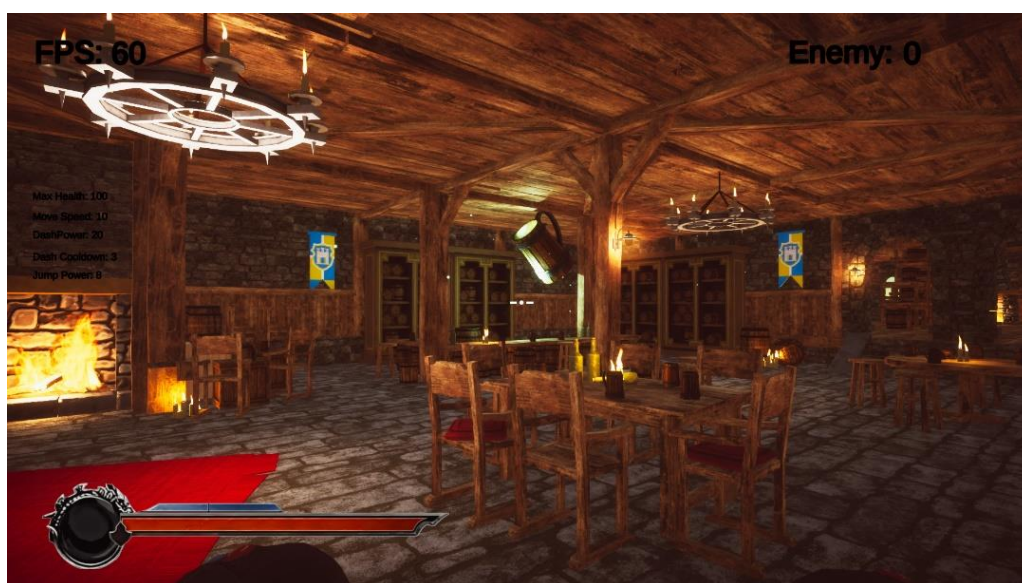


Рисунок 3.3 – Просторий бар

Відвідувачі трактира – це різноманітна група людей. Тут можна побачити пригодників у блискучих обладунках, місцевих жителів у простому одязі та навіть кількох монстрів, що спокійно сидять за столами. Вони зайняті різними справами: хтось п'є ель, хтось грає в азартні ігри, а хтось розповідає про свої подвиги.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				66
Змн.	Арк.	№ докум.	Підпис	Дата		



Рисунок 3.4 – Кімната в трактирі

Зображення показує інтер'єр великої багатофункціональної кімнати, яка функціонує як фентезійний трактир і склад одночасно (рис. 3.4).

Кілька дерев'яних столів і стільців розкидані по кімнаті, що вказує на місце для зібрань та спілкування відвідувачів. Зліва на зображенні розташований великий камін, що створює теплу та затишну атмосферу. Праворуч на зображенні є прилавок із кількома табуретами, що утворюють барну зону. На столах можна побачити залишки їжі та напоїв, які підказують, що це місце часто відвідується людьми. Дерев'яні меблі та аксесуари додають приміщенню сільського шарму, підкреслюючи загальний стиль інтер'єру.

Бочки, ящики та мішки високо складені вздовж задньої стіни та в кутах кімнати, що вказує на зберігання запасів та товарів. На стінах також є кілька полиць із різними предметами, зброєю, обладунками або іншим спорядженням.

Кімната слабо освітлена, з теплим світлом, що виходить від каміна та деяких ліхтарів, які висять зі стелі. Це створює атмосферне та, можливо, затишне відчуття, незважаючи на присутність складських запасів. Тепле освітлення створює гру тіней, додаючи приміщенню загадковості та глибини. На зображенні немає видимих людей, але наявність стільців і натяк на трактир припускають, що працівники можуть часто використовувати цей простір для відпочинку та розваг. Атмосфера кімнати свідчить про те, що це місце часто служить прихистком від зовнішніх негараздів, пропонуючи спокій та комфорт.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		67



Рисунок 3.5 – Зруйнована вартова вежа

Сцена зображує інтер'єр зруйнованої вартової вежі. Уламки та пошкодження по всій вежі свідчать про те, що вона пережила битву або інше руйнування. Стіни вартової вежі зроблені з грубо обробленого каменю. Великі тріщини та зламані балки розкидані по підлозі, натякаючи на руйнування. Можливо, в даху або верхніх стінах є дірки, через які світло проникає всередину (рис. 3.5).

На одній з внутрішніх стін висить велика дерев'яна дошка. Ця дошка завдань, ймовірно, слугує центром квестів для гравців, надаючи деталі та цілі для їхніх персонажів. Сама дошка може бути зношеною або пошарпаною, що відповідає пошкодженням вежі. На ній, ймовірно, розміщені різні деталі квестів, включаючи:

У вартовій вежі можуть бути й інші розкидані меблі або обладнання, знищені або покинуті через пошкодження. Шматочки паперу, сувої або маленькі торбинки можуть бути розкидані на підлозі навколо дошки завдань, представляючи потенційні підказки або додаткову інформацію про квести, яку гравці можуть знайти. Гравці, ретельно досліджуючи ці знахідки, можуть отримати цінні підказки, які допоможуть їм виконати завдання. Зібравши необхідну інформацію і підготувавшись до майбутніх випробувань, гравець отримує зброю, яка стає невід'ємною частиною його подорожі.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		68

Озброївшись, гравець виходить з вежі і опиняється в процедурно створеному лісі. Цей ліс, сповнений таємниць і небезпек, змінюється кожного разу, коли гравець вступає на його територію, забезпечуючи унікальний і непередбачуваний досвід. Гравець повинен бути готовий до зустрічі з різноманітними ворогами і перешкодами, які чекають на нього в цьому загадковому і непередбачуваному середовищі (рис. 3.6).



Рисунок 3.6 – Локація ліс

Рівень складається з кількох захоплюючих локацій, які об'єднують в собі захоплюючі елементи від ялинкового до березового лісу. Гравець, опиняється в прекрасному березовому лісі. Цей ліс вражає своєю витонченою красою та надає враження таємничості. Стовбури берез високі та стрункі, і вони, наче художні стовпи, співзвучні та виткані один навколо одного (рис. 3.7).

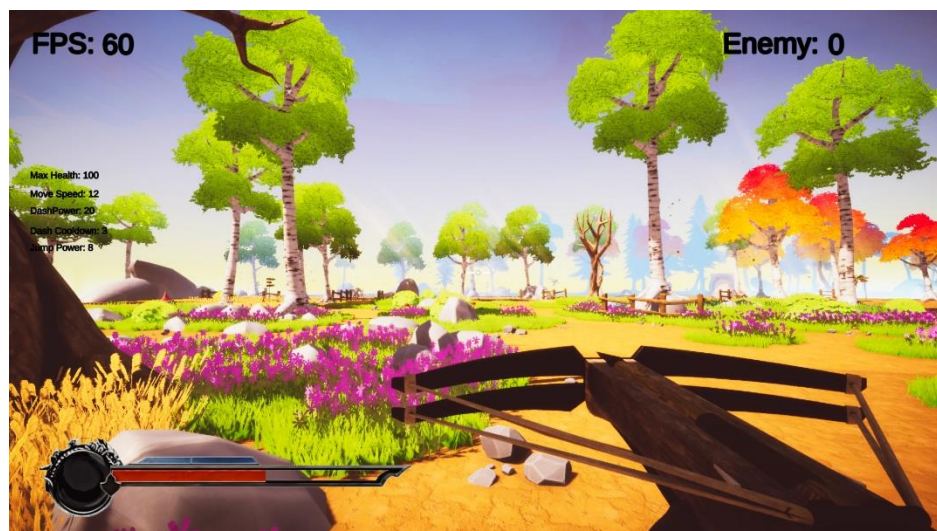


Рисунок 3.7 – Березовий ліс

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		69



Рисунок 3.8 – Березовий ліс

Звивиста ґрунтова стежка проходить через центр зображення. Шлях добре витоптаний, що свідчить про часте використання. Шлях веде до густого лісу з численними деревами. Сонячне світло проникає крізь листя дерев, кидаючи плямисті тіні на шлях і трав'янисту ділянку.

Загальний стиль фону схиляється до більш реалістичної естетики. Використання світла і тіні для створення глибини та вимірів дуже реалістичне. Особливо приємний дотик - те, як сонячне світло фільтрується крізь листя та створює плямисті тіні.

Ґрунтова стежка, трав'яниста ділянка та лісова підстилка мають текстурований вигляд, що робить їх більш реалістичними. Зображені дерева та рослини нагадують реальні види, такі як сосни та звичайні зелені кущі.

Використання освітлення, текстур і знайомої флори створює правдоподібне середовище. Хоча наявність кольорових квітів може натякати на фентезі, це не відволікає від загальної реалістичної атмосфери (рис. 3.9).

Додатково, розкидані по землі камені та коріння дерев додають природності й деталізації до сцени. Усе це створює враження, ніби гравець знаходиться у справжньому лісі. Анімації листя, що ворухиться під легким вітром, та тіні, що змінюються залежно від положення сонця, додають динаміки і життя до середовища

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		70



Рис. 3.9 Ялинковий ліс

Гравець, вирушає в ялинковий ліс який розгортається як справжнє природне чудо. Цей ліс наповнений не лише красою величних ялин, але й веселими білками, створюючи атмосферу спокою та взаємодії.



Рис. 3.10 Ялинковий ліс

Високі та величні ялинки стоять, наче стражі природи, створюючи неповторний гірський ландшафт. Білочки літають з гілки на гілку, збираючи жолуді та горіхи. Їх жвавість та шустра діяльність роблять ліс ще живим та цікавим для гравця (рис. 3.10).

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		71

3.2 Порядок встановлення, налаштування та тестування застосунку в ігровому рушії

Процес імпорту локації в Unity можна розбити на кілька кроків, які включають підготовку файлів, сам імпорт та подальшу обробку і налаштування локації у проекті. Після імпорту важливо провести подальшу обробку і налаштування локації у проекті, щоб вона відповідала потребам і концепції проекту.

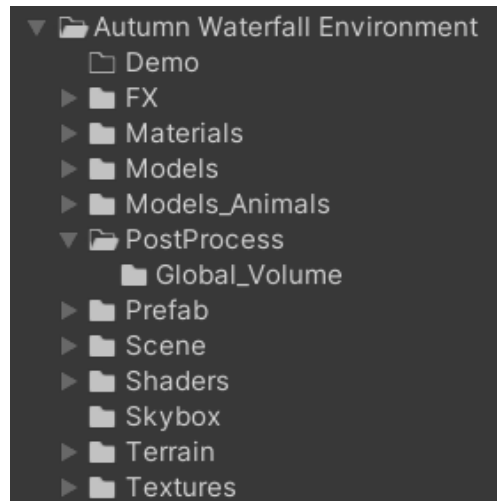


Рисунок 3.11 – Директорія де зберігаються всі файли локації

Отже, створимо папку або директорію, в яку зберігаємо всі файли локації. Потім перевіряємо формати файлів, які плануються імпортувати в Unity (рис. 3.11).

Unity підтримує такі формати:

1. Моделі: .fbx, .obj, .blend (якщо ви використовуєте Blender).
2. Текстури: .png, .jpg, .tga, .psd (також можуть бути використані формати з компресією для оптимізації розміру файлів).
3. Аудіо: .mp3, .wav, .ogg.
4. Відео: .mp4, .avi, .mov. Імпорт у Unity:

Зберігаємо всі файли локації у зручному для Unity форматі. Це може бути папка з текстурами, моделями 3D об'єктів, анімаціями, аудіо та іншими ресурсами.

Далі збираємо всі необхідні файли та налаштування в Unity Package, щоб зручно імпортувати їх у інші проекти та надіслати іншим користувачам.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		72

Потім у Unity вибираємо папку зі своїми ресурсами та файли, які потрібно додати до пакету, та вибираємо опцію «Export Package» (Експорт пакету) для створення Unity Package (рис. 3.12).

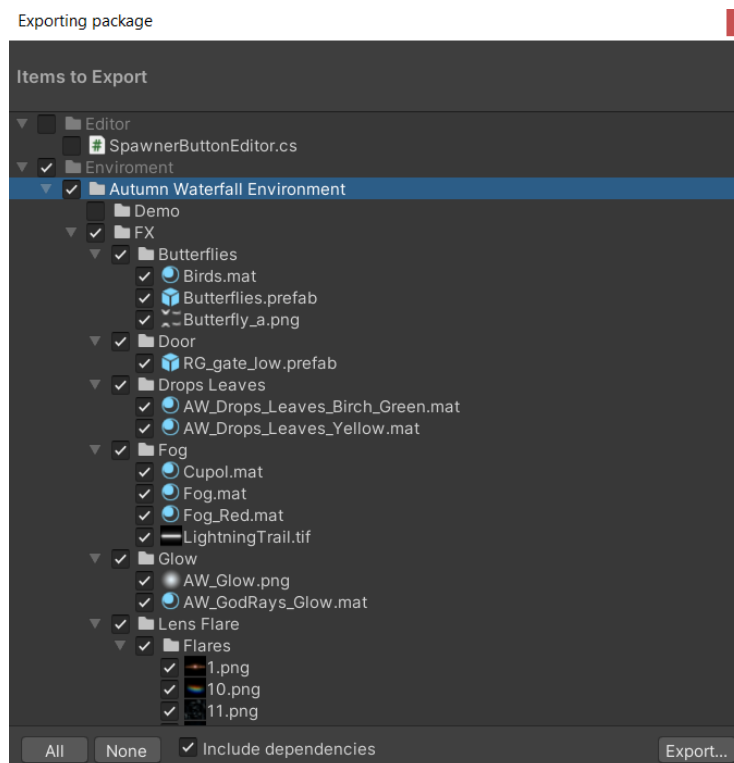


Рисунок 3.12 – Налаштування Export Package

Unity Package - це формат файлу, який використовується в Unity для зберігання інформації та ресурсів проекту, щоб їх можна було легко імпортувати, експортувати та обмінюватися між різними проектами або користувачами.

Unity Package дозволяє зберігати різні ресурси проекту, такі як сцени, моделі, текстури, аудіо, анімації, шейдери, скрипти, налаштування, плагіни та інше. Всі ці ресурси зберігаються у форматі, який зручно використовувати для перенесення між різними проектами або для ділитися з іншими користувачами.

Unity Package дозволяє експортувати весь або частину вашого проекту у вигляді пакету, який можна потім імпортувати в інші проекти Unity. Це дозволяє переносити окремі елементи, наприклад, створені скрипти, об'єкти зі сцен, налаштування або цілі папки з ресурсами між проектами без необхідності копіювання та вставляння кожного елемента окремо.

Unity Package є зручним способом обмінюватися ресурсами, кодом та налаштуваннями між різними членами команди або з іншими користувачами

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		73

Unity. Можливо створювати пакети з певними функціями, рівнями, об'єктами тощо, які можна додати до інших проектів або використовувати як додатковий контент для вашої гри чи застосунку.

Unity Package дозволяє зберігати версії різних частин проекту, що полегшує версіонування і керування різними варіантами ресурсів. Також це може бути зручним для резервного копіювання ключових елементів вашого проекту, щоб запобігти втраті даних у випадку нещасного випадку або помилки

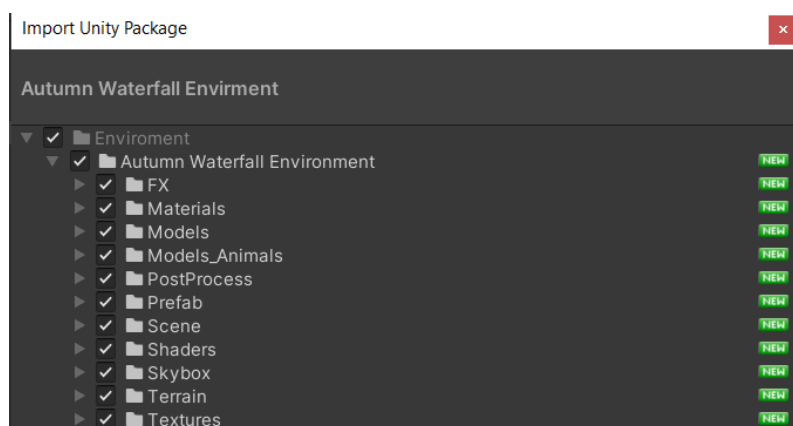


Рисунок 3.13 – Процес імпорту проекту

Відкриємо свій проект в Unity та перейдемо у вікно Project (Проект). Створимо нову папку для локації, імпортуємо усі файли, які потрібні, у цю папку (рис. 3.13).

Для цього в Unity існує вбудоване меню «Import», де можна вибрати файли для імпорту та налаштувати параметри імпорту, такі як масштабування, обрізання, налаштування колізій тощо. Преретягуємо файли зі свого комп'ютера до вікна Project (Проект) Unity, або використовуємо опцію Import (Імпорт) у меню Project (Проект) Unity, щоб вибрати файли для імпорту.

Після імпорту можна побачити попередній перегляд усіх імпортованих файлів у вікні Project (Проект) Unity (рис. 3.14). Для моделей можна додавати матеріали, налаштовувати анімацію, створювати колізії, додавати світло та ефекти. Текстури можна налаштовувати за допомогою Shader Graph або вбудованих налаштувань. Переконаємось, що всі файли імпортувалися коректно і виглядають як очікувалося.



Рисунок 3.14 – Попередній перегляд усіх імпортованих файлів

Потім для кожного типу файлу (моделі, текстури, аудіо) потрібно виконати налаштування та оптимізацію.

Для моделей: встановлення колізій, щоб вони взаємодіяли з іншими об'єктами у грі без зайвих перешкод. Оптимізація полігонів для зменшення навантаження на рендеринг та збільшення продуктивності гри. Додавання анімацій та анімаційних керувань для живлення моделей.

Для текстур: мінімізація розміру текстур для зменшення обсягу пам'яті та прискорення завантаження гри. Налаштування міп-карт для підвищення якості текстур при збільшенні та зменшенні віддаленості.

Для ефектів: додавання частинкових ефектів (Particle Systems) для створення різноманітних візуальних ефектів, таких як вибухи, дим, вогонь тощо. Налаштування параметрів ефектів, таких як швидкість, кількість частинок, розмір, кольори, щоб дати потрібний вигляд ефектам.

Для шейдерів: створення і редагування шейдерів за допомогою Shader Graph або покового підходу для реалізації різних візуальних ефектів та стилів. Оптимізація шейдерів для зменшення використання ресурсів та підвищення продуктивності гри.

Після обробки файлів та налаштування їхніх властивостей можна почати розміщувати об'єкти у локації. Спочатку перетягуємо об'єкти (моделі, текстури, аудіо) з вікна Project (Проект) Unity у нашу сцену. Розміщаємо їх у відповідних місцях нашої локації, налаштуємо освітлення, колізії, анімацію та інші деталі (рис. 3.15).

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		75

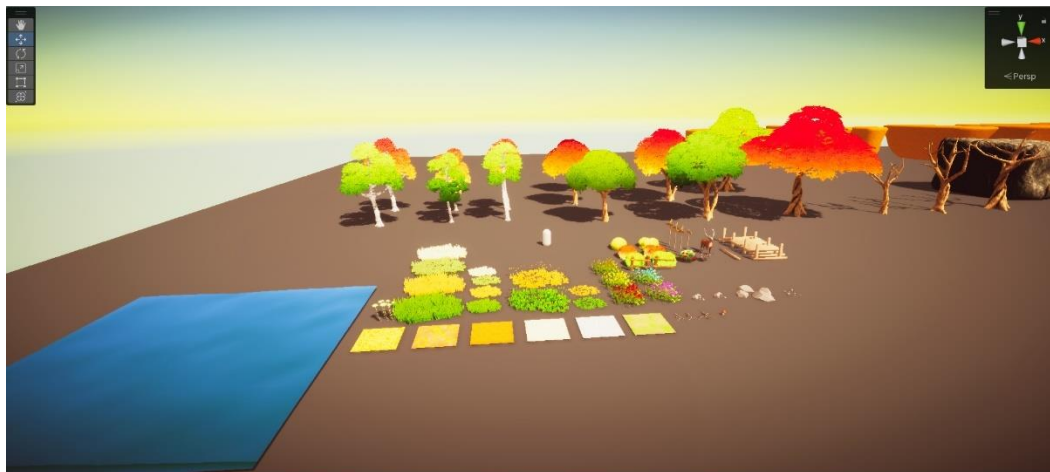


Рисунок 3.15 – Перевірка всіх моделей з налаштуванням

Наступним кроком тестування є переведення проекту в версію Universal Render Pipeline (URP) в Unity. Це є важливим кроком для поліпшення графіки та продуктивності інтерактивного застосунку (рис. 3.16).

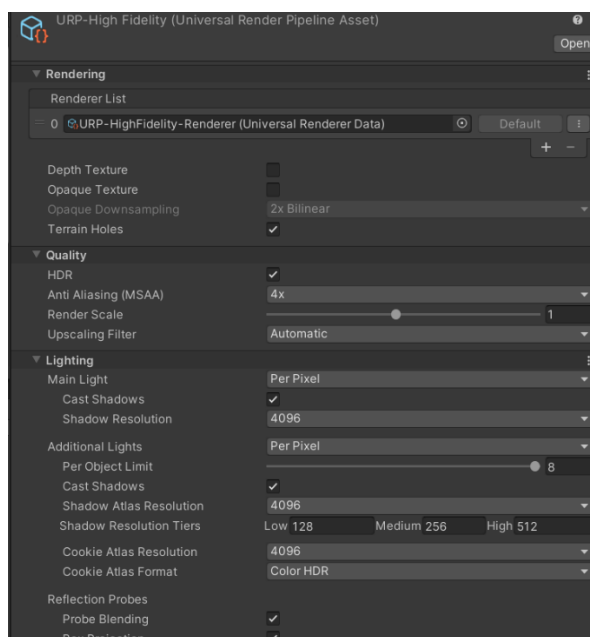


Рисунок 3.16 – Процес переведення проекту в версію Universal Render Pipeline

Unity Universal Render Pipeline (URP) - це легкий і гнучкий рендерний пайплайн, призначений для створення високоякісних графічних ефектів у проектах Unity. Він дозволяє розробникам створювати стильні та динамічні візуальні ефекти з низьким використанням ресурсів, що робить його ідеальним вибором для мобільних платформ, веб-ігор та інших проектів, де важлива продуктивність.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				76
Змн.	Арк.	№ докум.	Підпис	Дата		

Рендерний пайплайн URP пропонує наступні можливості:

Lightweight Render Pipeline (LWRP): у поєднанні з попередніми версіями Unity, URP раніше відомий як LWRP. Це забезпечує швидке та ефективне рендерінг сцени з високим рівнем оптимізації для мобільних та інших пристроїв.

URP підтримує Shader Graph, що дозволяє легко створювати та налаштовувати шейдери без програмування, що спрощує процес розробки та експериментування з візуальними ефектами.

Підтримка Post-Processing ефектів: URP має вбудовану підтримку Post-Processing ефектів, таких як розмиття, кольорові корекції, ефекти камери тощо, що додає різноманітність та глибину візуальним ефектам.

URP дозволяє налаштовувати різні аспекти рендерингу, включаючи освітлення, тіні, матеріали та інші параметри, щоб досягти потрібного вигляду та продуктивності гри. Cross-Platform підтримка URP розроблений для роботи на різних платформах, включаючи мобільні, десктопні та веб-платформи, що робить його універсальним рішенням для різноманітних проектів

Після налаштування імпортованих об'єктів важливо перевірити їх у грі для виявлення будь-яких проблем або несумісностей. Тестування допоможе впевнитися, що імпортовані об'єкти працюють коректно та відповідають вимогам проекту. Для цього запустимо гру для тестування локації. Перевіримо, чи все виглядає та працює належним чином, та вносимо необхідні корективи (рис. 3.17).

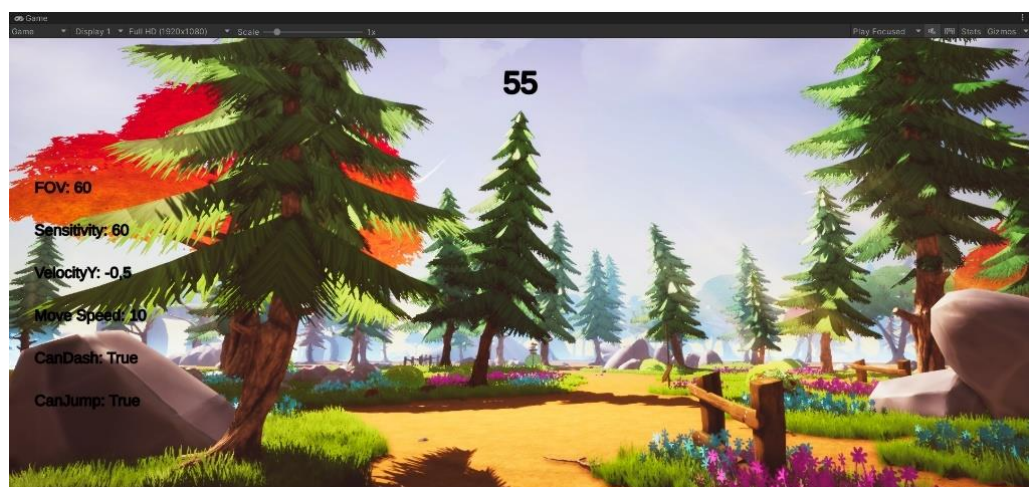


Рисунок 3.17 – Перевірка об'єктів у грі

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				77
Змн.	Арк.	№ докум.	Підпис	Дата		

Висновки до третього розділу

Було продемонстровано інтер'єр трактиру в фентезійному світі, де кожен елемент, від бару до каміна, пропонує відвідувачам глибоку атмосферу історій та пригод. У вартівній вежі було продемонстровано пошкодження та руйнування, що створюють атмосферу загадковості і пригод. Розкидані предмети та дошка завдань надають гравцю можливість дослідження і поглиблення у групову взаємодію. Лісова локація, включаючи ялинковий і березовий ліси, створює враження живого та змінюваного середовища. Взаємодія з різноманітними перешкодами та ворогами у кожній локації надає гравцю великий діапазон гри та досвіду.

Проведено аналіз основних етапів імпорту та обробки локацій та ресурсів у Unity, включаючи підготовку файлів, їх імпорт, обробку та налаштування у проєкті. Показано процес важливості дотримання форматів файлів, підтримуваних Unity, для успішного імпорту різних типів ресурсів. Продемонстровано процес використання Unity Package як зручного засобу для зберігання та обміну ресурсами між проєктами. Тестування локації після імпорту та налаштування ресурсів допомагає виявити проблеми та забезпечити коректну роботу об'єктів у грі.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		78

Висновки

Під час виконання кваліфікаційної роботи була успішно розроблена концепція та технічна база для комп'ютерної гри у стилі Roguelite. Успішно встановлено ціль та сформульовано завдання для розробки ігрового застосунку "Forbidden Forest". Проведено детальний аналіз трьох відомих ігор (Borderlands, City of Brass, The Cycle), що допомогло виявити їх сильні та слабкі сторони. Визначено необхідну візуалізацію для створення інтерактивного 3D світу, а також обґрунтовано вибір інструментарію (Unity) для розробки.

Проведено аналіз процесу розробки концепції інтерактивного 3D світу через діаграми та детально описано кроки розробки локацій та візуальних ефектів. Розроблено власний шейдер для покращення візуалізації ігрового застосунку. Проведено аналіз ефективності використання ресурсів під час розробки, що дозволило оптимізувати процес та забезпечити високу продуктивність гри.

Продемонстровано інтер'єр трактиру в фентезійному світі та описано враження від різних локацій і елементів середовища. Проведено детальний аналіз основних етапів імпорту та обробки локацій та ресурсів у Unity, включаючи важливість дотримання форматів файлів та використання Unity Package. Зазначено важливість тестування гри після імпорту та налаштування ресурсів для виявлення проблем та забезпечення коректної роботи гри.

Таким чином, розроблена концептуальна та технічна база гри в жанрі Roguelite виявилася не лише успішною, але й високоефективною з точки зору оптимізації ресурсів та якості геймплею. Виявлення та усунення можливих проблем з продуктивністю дозволили досягти високої швидкодії та ефективності використання ресурсів, що є критичним для забезпечення задоволення користувачів та збереження репутації гри.

Вище вказані результати роблять підґрунтя для подальшого розвитку та успішного випуску гри на ринок ігрової індустрії.

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		79

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Borderlands [Електронний ресурс]:- Режим доступу:
<https://borderlandsthegame.com>
2. City of Brass [Електронний ресурс]:- Режим доступу:
<https://www.cityofbrassgame.com>
3. The Cycle [Електронний ресурс]:- Режим доступу:
<https://thecycle.game>
4. Офіційний сайт Unity [Електронний ресурс]:- Режим доступу:
<https://unity3d.com/ua/unity>
5. Game Interface Elements. GUI Scripting Guide [Електронний ресурс]:-
Режим доступу:
<http://docs.unity3d.ua/Manual/Game%20Interface%20Elements.html>
6. Unity Manual - Particle Systems [Електронний ресурс]:- Режим
доступу: <https://docs.unity3d.com/Manual/ParticleSystem.html>
7. Unity Shader Graph [Електронний ресурс]:- Режим доступу:
<https://docs.unity3d.com/Manual/ShaderGraph.html>
8. Unity VFX Graph [Електронний ресурс]:- Режим доступу:
<https://docs.unity3d.com/Manual/VFXGraph.html>
9. Unity Asset Store [Електронний ресурс]:- Режим доступу:
<https://assetstore.unity.com/>
10. Blender - офіційний сайт [Електронний ресурс]:- Режим доступу:
<https://www.blender.org/>
11. Unity URP [Електронний ресурс]:- Режим доступу:
<https://docs.unity3d.com/Packages/com.unity.render-pipelines.universal@10.2/manual/index.html>
12. Unity LOD System [Електронний ресурс]:- Режим доступу:
<https://docs.unity3d.com/Manual/class-LODGroup.html>
13. Unity Profiler [Електронний ресурс]:- Режим доступу:
<https://docs.unity3d.com/Manual/Profiler.html>

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				80
Змн.	Арк.	№ докум.	Підпис	Дата		

14. Unity Visual Effect Graph Documentation [Електронний ресурс]:- Режим доступу: <https://docs.unity3d.com/Packages/com.unity.visualeffectgraph@7.3/manual/index.html>
15. Unity Importing Models [Електронний ресурс]:- Режим доступу: <https://docs.unity3d.com/Manual/HOWTO-importObject.html>
16. Unity Particle System [Електронний ресурс]:- Режим доступу: <https://docs.unity3d.com/Manual/ParticleSystem.html>
17. Unity Package Manager [Електронний ресурс]:- Режим доступу: <https://docs.unity3d.com/Manual/upm-ui.html>
18. Unity Asset Import Settings [Електронний ресурс]:- Режим доступу: <https://docs.unity3d.com/Manual/FBXImporter-Model.html>
19. Unity Terrain Tools [Електронний ресурс]:- Режим доступу: <https://docs.unity3d.com/Manual/terrain-Tools.html>
20. "The Art of Game Design: A Book of Lenses" by Jesse Schell [Електронний ресурс]:- Режим доступу: <https://www.amazon.com/Art-Game-Design-Book-Lenses/dp/0123694965>
21. "Level Up! The Guide to Great Video Game Design" by Scott Rogers [Електронний ресурс]:- Режим доступу: <https://www.amazon.com/Level-Guide-Great-Video-Design/dp/1118877160>
22. "Real-Time Rendering" by Tomas Akenine-Möller, Eric Haines, and Naty Hoffman [Електронний ресурс]:- Режим доступу: <https://www.amazon.com/Real-Time-Rendering-Third-Tomas-Akenine-Moller/dp/1568814240>
23. "Digital Painting Techniques: Practical Techniques of Digital Art Masters" by 3dtotal Publishing [Електронний ресурс]:- Режим доступу: <https://www.amazon.com/Digital-Painting-Techniques-Practical-Masters/dp/0955153077>
24. "Rules of Play: Game Design Fundamentals" by Katie Salen Tekinbas and Eric Zimmerman [Електронний ресурс]:- Режим доступу: <https://www.amazon.com/Rules-Play-Game-Design-Fundamentals/dp/0262240459>

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		81

25. "Introduction to Game Design, Prototyping, and Development" by Jeremy Gibson Bond [Електронний ресурс]:- Режим доступу: <https://www.amazon.com/Introduction-Design-Prototyping-Development-Second/dp/0134659864>

26. "The Indie Game Developer Handbook" by Richard Hill-Whittall [Електронний ресурс]:- Режим доступу: <https://www.amazon.com/Indie-Game-Developer-Handbook/dp/1138828420>

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				82
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				83
Змн.	Арк.	№ докум.	Підпис	Дата		

Clouds.shader – шейдер створення хмар, їх руху, зміни кольорів тощо

```

Shader "Custom/NewSurfaceShader" {
Properties {
    _Color ("Color", Color) = (1.000000,1.000000,1.000000,1.000000)
    _MainTex ("Albedo (RGB)", 2D) = "white" { }
    _Glossiness ("Smoothness", Range(0.000000,1.000000)) = 0.500000
    _Metallic ("Metallic", Range(0.000000,1.000000)) = 0.000000
}
SubShader {
    LOD 200
    Tags { "RenderType"="Opaque" }

    // Stats for Vertex shader:
    //      d3d11: 27 avg math (16..50)
    // Stats for Fragment shader:
    //      d3d11: 155 avg math (133..177), 4 avg texture (2..6), 7 avg branch (6..8)
    Pass {
        Name "FORWARD"
        Tags { "LIGHTMODE"="FORWARD" "SHADOWSUPPORT"="true" "RenderType"="Opaque" }
        ///////////////////////////////////
        //                               //
        //      Compiled programs      //
        //                               //
        ///////////////////////////////////
        ///////////////////////////////////
    Keywords: DIRECTIONAL
    -- Hardware tier variant: Tier 1
    -- Vertex shader for "d3d11":
    // Stats: 16 math, 2 temp registers
    Uses vertex data channel "Vertex"
    Uses vertex data channel "Normal"
    Uses vertex data channel "TexCoord0"

    Constant Buffer "$Globals" (112 bytes) on slot 0 {
        Vector4 _MainTex_ST at 96
    }
    Constant Buffer "UnityPerDraw" (176 bytes) on slot 1 {
        Matrix4x4 unity_ObjectToWorld at 0
        Matrix4x4 unity_WorldToObject at 64
    }
    Constant Buffer "UnityPerFrame" (368 bytes) on slot 2 {
        Matrix4x4 unity_MatrixVP at 272
    }
}

```

Shader Disassembly:

```

//
// Generated by Microsoft (R) D3D Shader Disassembler
//
//
// Input signature:
//
// Name                Index  Mask Register SysValue  Format  Used
// -----
// POSITION              0    xyzw      0      NONE   float  xyzw
// TANGENT              0    xyzw      1      NONE   float
// NORMAL              0    xyz       2      NONE   float  xyz
// TEXCOORD             0    xyzw      3      NONE   float  xy
// TEXCOORD             1    xyzw      4      NONE   float
// TEXCOORD             2    xyzw      5      NONE   float
// TEXCOORD             3    xyzw      6      NONE   float
// COLOR               0    xyzw      7      NONE   float

```

```
//
//
// Output signature:
//
// Name Index Mask Register SysValue Format Used
// -----
// SV_POSITION 0 xyzw 0 POS float xyzw
// TEXCOORD 0 xy 1 NONE float xy
// TEXCOORD 1 xyz 2 NONE float xyz
// TEXCOORD 2 xyz 3 NONE float xyz
// TEXCOORD 5 xyzw 4 NONE float xyzw
// TEXCOORD 6 xyzw 5 NONE float xyzw
//
```

```
vs_4_0
dcl_constantbuffer CB0[7], immediateIndexed
dcl_constantbuffer CB1[7], immediateIndexed
dcl_constantbuffer CB2[21], immediateIndexed
dcl_input v0.xyzw
dcl_input v2.xyz
dcl_input v3.xy
dcl_output_siv o0.xyzw, position
dcl_output o1.xy
dcl_output o2.xyz
dcl_output o3.xyz
dcl_output o4.xyzw
dcl_output o5.xyzw
dcl_temps 2
0: mul r0.xyzw, v0.yyyy, cb1[1].xyzw
1: mad r0.xyzw, cb1[0].xyzw, v0.xxxx, r0.xyzw
2: mad r0.xyzw, cb1[2].xyzw, v0.zzzz, r0.xyzw
3: add r1.xyzw, r0.xyzw, cb1[3].xyzw
4: mad o3.xyz, cb1[3].xyzx, v0.wwww, r0.xyzx
5: mul r0.xyzw, r1.yyyy, cb2[18].xyzw
6: mad r0.xyzw, cb2[17].xyzw, r1.xxxx, r0.xyzw
7: mad r0.xyzw, cb2[19].xyzw, r1.zzzz, r0.xyzw
8: mad o0.xyzw, cb2[20].xyzw, r1.wwww, r0.xyzw
9: mad o1.xy, v3.xyxx, cb0[6].xyxx, cb0[6].zwzz
10: dp3 r0.x, v2.xyzx, cb1[4].xyzx
11: dp3 r0.y, v2.xyzx, cb1[5].xyzx
12: dp3 r0.z, v2.xyzx, cb1[6].xyzx
13: dp3 r0.w, r0.xyzx, r0.xyzx
14: rsq r0.w, r0.w
15: mul o2.xyz, r0.wwww, r0.xyzx
16: mov o4.xyzw, l(0,0,0,0)
17: mov o5.xyzw, l(0,0,0,0)
18: ret
```

// Approximately 0 instruction slots used

-- Hardware tier variant: Tier 1
-- Fragment shader for "d3d11":

// Stats: 133 math, 12 temp registers, 2 textures, 6 branches
Set 2D Texture "_MainTex" to slot 0 sampler slot 2
Set CUBE Texture "unity_SpecCube0" to slot 1 sampler slot 0
Set CUBE Texture "unity_SpecCube1" to slot 2 sampler slot -1
Set 3D Texture "unity_ProbeVolumeSH" to slot 3 sampler slot 1

Constant Buffer "\$Globals" (112 bytes) on slot 0 {
Vector4 _LightColor0 at 32
Float _Glossiness at 64
Float _Metallic at 68
Vector4 _Color at 80
}

Constant Buffer "UnityPerCamera" (144 bytes) on slot 1 {
Vector3 _WorldSpaceCameraPos at 64

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		85

```

}
Constant Buffer "UnityLighting" (768 bytes) on slot 2 {
    Vector4 _WorldSpaceLightPos0 at 0
    Vector4 unity_OcclusionMaskSelector at 736
}
Constant Buffer "UnityReflectionProbes" (128 bytes) on slot 3 {
    Vector4 unity_SpecCube0_BoxMax at 0
    Vector4 unity_SpecCube0_BoxMin at 16
    Vector4 unity_SpecCube0_ProbePosition at 32
    Vector4 unity_SpecCube0_HDR at 48
    Vector4 unity_SpecCube1_BoxMax at 64
    Vector4 unity_SpecCube1_BoxMin at 80
    Vector4 unity_SpecCube1_ProbePosition at 96
    Vector4 unity_SpecCube1_HDR at 112
}
Constant Buffer "UnityProbeVolume" (112 bytes) on slot 4 {
    Matrix4x4 unity_ProbeVolumeWorldToObject at 16
    Vector4 unity_ProbeVolumeParams at 0
    Vector3 unity_ProbeVolumeSizeInv at 80
    Vector3 unity_ProbeVolumeMin at 96
}
Keywords: DIRECTIONAL LIGHTPROBE_SH
-- Hardware tier variant: Tier 1
-- Vertex shader for "d3d11":
// Stats: 23 math, 2 temp registers
Uses vertex data channel "Vertex"
Uses vertex data channel "Normal"
Uses vertex data channel "TexCoord0"

Constant Buffer "$Globals" (112 bytes) on slot 0 {
    Vector4 _MainTex_ST at 96
}
Constant Buffer "UnityLighting" (768 bytes) on slot 1 {
    Vector4 unity_SHBr at 672
    Vector4 unity_SHBg at 688
    Vector4 unity_SHBb at 704
    Vector4 unity_SHC at 720
}
Constant Buffer "UnityPerDraw" (176 bytes) on slot 2 {
    Matrix4x4 unity_ObjectToWorld at 0
    Matrix4x4 unity_WorldToObject at 64
}
Constant Buffer "UnityPerFrame" (368 bytes) on slot 3 {
    Matrix4x4 unity_MatrixVP at 272
}
// Stats: 160 math, 13 temp registers, 5 textures, 8 branches
Set 2D Texture "_MainTex" to slot 0 sampler slot 2
Set CUBE Texture "unity_SpecCube0" to slot 1 sampler slot 0
Set CUBE Texture "unity_SpecCube1" to slot 2 sampler slot -1
Set 3D Texture "unity_ProbeVolumeSH" to slot 3 sampler slot 1

Constant Buffer "$Globals" (112 bytes) on slot 0 {
    Vector4 _LightColor0 at 32
    Float _Glossiness at 64
    Float _Metallic at 68
    Vector4 _Color at 80
}
Constant Buffer "UnityPerCamera" (144 bytes) on slot 1 {
    Vector3 _WorldSpaceCameraPos at 64
}
Constant Buffer "UnityLighting" (768 bytes) on slot 2 {
    Vector4 _WorldSpaceLightPos0 at 0
    Vector4 unity_SHAr at 624
    Vector4 unity_SHAg at 640
    Vector4 unity_SHAb at 656

```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		86

```

    Vector4 unity_OcclusionMaskSelector at 736
}
Constant Buffer "UnityReflectionProbes" (128 bytes) on slot 3 {
    Vector4 unity_SpecCube0_BoxMax at 0
    Vector4 unity_SpecCube0_BoxMin at 16
    Vector4 unity_SpecCube0_ProbePosition at 32
    Vector4 unity_SpecCube0_HDR at 48
    Vector4 unity_SpecCube1_BoxMax at 64
    Vector4 unity_SpecCube1_BoxMin at 80
    Vector4 unity_SpecCube1_ProbePosition at 96
    Vector4 unity_SpecCube1_HDR at 112
}
Constant Buffer "UnityProbeVolume" (112 bytes) on slot 4 {
    Matrix4x4 unity_ProbeVolumeWorldToObject at 16
    Vector4 unity_ProbeVolumeParams at 0
    Vector3 unity_ProbeVolumeSizeInv at 80
    Vector3 unity_ProbeVolumeMin at 96
}

```

Keywords: DIRECTIONAL SHADOWS_SCREEN

-- Hardware tier variant: Tier 1

-- Vertex shader for "d3d11":

// Stats: 19 math, 2 temp registers

Uses vertex data channel "Vertex"

Uses vertex data channel "Normal"

Uses vertex data channel "TexCoord0"

```

Constant Buffer "$Globals" (112 bytes) on slot 0 {
    Vector4 _MainTex_ST at 96
}
Constant Buffer "UnityPerCamera" (144 bytes) on slot 1 {
    Vector4 _ProjectionParams at 80
}
Constant Buffer "UnityPerDraw" (176 bytes) on slot 2 {
    Matrix4x4 unity_ObjectToWorld at 0
    Matrix4x4 unity_WorldToObject at 64
}
Constant Buffer "UnityPerFrame" (368 bytes) on slot 3 {
    Matrix4x4 unity_MatrixVP at 272
}

```

-- Hardware tier variant: Tier 1

-- Fragment shader for "d3d11":

// Stats: 143 math, 12 temp registers, 3 textures, 6 branches

Set 2D Texture "_MainTex" to slot 0 sampler slot 3

Set 2D Texture "_ShadowMapTexture" to slot 1 sampler slot 2

Set CUBE Texture "unity_SpecCube0" to slot 2 sampler slot 0

Set CUBE Texture "unity_SpecCube1" to slot 3 sampler slot -1

Set 3D Texture "unity_ProbeVolumeSH" to slot 4 sampler slot 1

```

Constant Buffer "$Globals" (112 bytes) on slot 0 {
    Vector4 _LightColor0 at 32
    Float _Glossiness at 64
    Float _Metallic at 68
    Vector4 _Color at 80
}
Constant Buffer "UnityPerCamera" (144 bytes) on slot 1 {
    Vector3 _WorldSpaceCameraPos at 64
}
Constant Buffer "UnityLighting" (768 bytes) on slot 2 {
    Vector4 _WorldSpaceLightPos0 at 0
    Vector4 unity_OcclusionMaskSelector at 736
}
Constant Buffer "UnityShadows" (416 bytes) on slot 3 {
    Vector4 _LightShadowData at 384
    Vector4 unity_ShadowFadeCenterAndType at 400
}

```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				87
Змн.	Арк.	№ докум.	Підпис	Дата		

```

}
Constant Buffer "UnityPerFrame" (368 bytes) on slot 4 {
    Matrix4x4 unity_MatrixV at 144
}
Constant Buffer "UnityReflectionProbes" (128 bytes) on slot 5 {
    Vector4 unity_SpecCube0_BoxMax at 0
    Vector4 unity_SpecCube0_BoxMin at 16
    Vector4 unity_SpecCube0_ProbePosition at 32
    Vector4 unity_SpecCube0_HDR at 48
    Vector4 unity_SpecCube1_BoxMax at 64
    Vector4 unity_SpecCube1_BoxMin at 80
    Vector4 unity_SpecCube1_ProbePosition at 96
    Vector4 unity_SpecCube1_HDR at 112
}
Constant Buffer "UnityProbeVolume" (112 bytes) on slot 6 {
    Matrix4x4 unity_ProbeVolumeWorldToObject at 16
    Vector4 unity_ProbeVolumeParams at 0
    Vector3 unity_ProbeVolumeSizeInv at 80
    Vector3 unity_ProbeVolumeMin at 96
}

```

Keywords: DIRECTIONAL LIGHTPROBE_SH SHADOWS_SCREEN

-- Hardware tier variant: Tier 1

-- Vertex shader for "d3d11":

// Stats: 26 math, 3 temp registers

Uses vertex data channel "Vertex"

Uses vertex data channel "Normal"

Uses vertex data channel "TexCoord0"

```

Constant Buffer "$Globals" (112 bytes) on slot 0 {
    Vector4 _MainTex_ST at 96
}

```

```

Constant Buffer "UnityPerCamera" (144 bytes) on slot 1 {
    Vector4 _ProjectionParams at 80
}

```

```

Constant Buffer "UnityLighting" (768 bytes) on slot 2 {
    Vector4 unity_SHBr at 672
    Vector4 unity_SHBg at 688
    Vector4 unity_SHBb at 704
    Vector4 unity_SHC at 720
}

```

```

Constant Buffer "UnityPerDraw" (176 bytes) on slot 3 {
    Matrix4x4 unity_ObjectToWorld at 0
    Matrix4x4 unity_WorldToObject at 64
}

```

```

Constant Buffer "UnityPerFrame" (368 bytes) on slot 4 {
    Matrix4x4 unity_MatrixVP at 272
}

```

-- Hardware tier variant: Tier 1

-- Fragment shader for "d3d11":

// Stats: 170 math, 13 temp registers, 6 textures, 8 branches

Set 2D Texture "_MainTex" to slot 0 sampler slot 3

Set 2D Texture "_ShadowMapTexture" to slot 1 sampler slot 2

Set CUBE Texture "unity_SpecCube0" to slot 2 sampler slot 0

Set CUBE Texture "unity_SpecCube1" to slot 3 sampler slot -1

Set 3D Texture "unity_ProbeVolumeSH" to slot 4 sampler slot 1

```

Constant Buffer "$Globals" (112 bytes) on slot 0 {
    Vector4 _LightColor0 at 32
    Float _Glossiness at 64
    Float _Metallic at 68
    Vector4 _Color at 80
}

```

```

Constant Buffer "UnityPerCamera" (144 bytes) on slot 1 {
    Vector3 _WorldSpaceCameraPos at 64
}

```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		88

```

Constant Buffer "UnityLighting" (768 bytes) on slot 2 {
    Vector4 _WorldSpaceLightPos0 at 0
    Vector4 unity_SHAr at 624
    Vector4 unity_SHAg at 640
    Vector4 unity_SHAb at 656
    Vector4 unity_OcclusionMaskSelector at 736
}
Constant Buffer "UnityShadows" (416 bytes) on slot 3 {
    Vector4 _LightShadowData at 384
    Vector4 unity_ShadowFadeCenterAndType at 400
}
Constant Buffer "UnityPerFrame" (368 bytes) on slot 4 {
    Matrix4x4 unity_MatrixV at 144
}
Constant Buffer "UnityReflectionProbes" (128 bytes) on slot 5 {
    Vector4 unity_SpecCube0_BoxMax at 0
    Vector4 unity_SpecCube0_BoxMin at 16
    Vector4 unity_SpecCube0_ProbePosition at 32
    Vector4 unity_SpecCube0_HDR at 48
    Vector4 unity_SpecCube1_BoxMax at 64
    Vector4 unity_SpecCube1_BoxMin at 80
    Vector4 unity_SpecCube1_ProbePosition at 96
    Vector4 unity_SpecCube1_HDR at 112
}
Constant Buffer "UnityProbeVolume" (112 bytes) on slot 6 {
    Matrix4x4 unity_ProbeVolumeWorldToObject at 16
    Vector4 unity_ProbeVolumeParams at 0
    Vector3 unity_ProbeVolumeSizeInv at 80
    Vector3 unity_ProbeVolumeMin at 96
}
Keywords: DIRECTIONAL FOG_LINEAR
-- Vertex shader for "d3d11":
// No shader variant for this keyword set. The closest match will be used instead.

-- Hardware tier variant: Tier 1
-- Fragment shader for "d3d11":
// Stats: 140 math, 12 temp registers, 2 textures, 6 branches
Set 2D Texture "_MainTex" to slot 0 sampler slot 2
Set CUBE Texture "unity_SpecCube0" to slot 1 sampler slot 0
Set CUBE Texture "unity_SpecCube1" to slot 2 sampler slot -1
Set 3D Texture "unity_ProbeVolumeSH" to slot 3 sampler slot 1

Constant Buffer "$Globals" (112 bytes) on slot 0 {
    Vector4 _LightColor0 at 32
    Float _Glossiness at 64
    Float _Metallic at 68
    Vector4 _Color at 80
}
Constant Buffer "UnityPerCamera" (144 bytes) on slot 1 {
    Vector3 _WorldSpaceCameraPos at 64
    Vector4 _ProjectionParams at 80
}
Constant Buffer "UnityLighting" (768 bytes) on slot 2 {
    Vector4 _WorldSpaceLightPos0 at 0
    Vector4 unity_OcclusionMaskSelector at 736
}
Constant Buffer "UnityFog" (32 bytes) on slot 3 {
    Vector4 unity_FogColor at 0
    Vector4 unity_FogParams at 16
}
Constant Buffer "UnityReflectionProbes" (128 bytes) on slot 4 {
    Vector4 unity_SpecCube0_BoxMax at 0
    Vector4 unity_SpecCube0_BoxMin at 16
    Vector4 unity_SpecCube0_ProbePosition at 32
    Vector4 unity_SpecCube0_HDR at 48

```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		89

```

Vector4 unity_SpecCube1_BoxMax at 64
Vector4 unity_SpecCube1_BoxMin at 80
Vector4 unity_SpecCube1_ProbePosition at 96
Vector4 unity_SpecCube1_HDR at 112
}
Constant Buffer "UnityProbeVolume" (112 bytes) on slot 5 {
    Matrix4x4 unity_ProbeVolumeWorldToObject at 16
    Vector4 unity_ProbeVolumeParams at 0
    Vector3 unity_ProbeVolumeSizeInv at 80
    Vector3 unity_ProbeVolumeMin at 96
}

Keywords: DIRECTIONAL FOG_LINEAR LIGHTPROBE_SH
-- Vertex shader for "d3d11":
// No shader variant for this keyword set. The closest match will be used instead.

-- Hardware tier variant: Tier 1
-- Fragment shader for "d3d11":
// Stats: 167 math, 13 temp registers, 5 textures, 8 branches
Set 2D Texture "_MainTex" to slot 0 sampler slot 2
Set CUBE Texture "unity_SpecCube0" to slot 1 sampler slot 0
Set CUBE Texture "unity_SpecCube1" to slot 2 sampler slot -1
Set 3D Texture "unity_ProbeVolumeSH" to slot 3 sampler slot 1

Constant Buffer "$Globals" (112 bytes) on slot 0 {
    Vector4 _LightColor0 at 32
    Float _Glossiness at 64
    Float _Metallic at 68
    Vector4 _Color at 80
}
Constant Buffer "UnityPerCamera" (144 bytes) on slot 1 {
    Vector3 _WorldSpaceCameraPos at 64
    Vector4 _ProjectionParams at 80
}
Constant Buffer "UnityLighting" (768 bytes) on slot 2 {
    Vector4 _WorldSpaceLightPos0 at 0
    Vector4 unity_SHAr at 624
    Vector4 unity_SHAg at 640
    Vector4 unity_SHAb at 656
    Vector4 unity_OcclusionMaskSelector at 736
}
Constant Buffer "UnityFog" (32 bytes) on slot 3 {
    Vector4 unity_FogColor at 0
    Vector4 unity_FogParams at 16
}
Constant Buffer "UnityReflectionProbes" (128 bytes) on slot 4 {
    Vector4 unity_SpecCube0_BoxMax at 0
    Vector4 unity_SpecCube0_BoxMin at 16
    Vector4 unity_SpecCube0_ProbePosition at 32
    Vector4 unity_SpecCube0_HDR at 48
    Vector4 unity_SpecCube1_BoxMax at 64
    Vector4 unity_SpecCube1_BoxMin at 80
    Vector4 unity_SpecCube1_ProbePosition at 96
    Vector4 unity_SpecCube1_HDR at 112
}
Constant Buffer "UnityProbeVolume" (112 bytes) on slot 5 {
    Matrix4x4 unity_ProbeVolumeWorldToObject at 16
    Vector4 unity_ProbeVolumeParams at 0
    Vector3 unity_ProbeVolumeSizeInv at 80
    Vector3 unity_ProbeVolumeMin at 96
}

Keywords: DIRECTIONAL FOG_LINEAR SHADOWS_SCREEN
-- Vertex shader for "d3d11":
// No shader variant for this keyword set. The closest match will be used instead.

```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				90
Змн.	Арк.	№ докум.	Підпис	Дата		

```

-- Hardware tier variant: Tier 1
-- Fragment shader for "d3d11":
// Stats: 150 math, 12 temp registers, 3 textures, 6 branches
Set 2D Texture "_MainTex" to slot 0 sampler slot 3
Set 2D Texture "_ShadowMapTexture" to slot 1 sampler slot 2
Set CUBE Texture "unity_SpecCube0" to slot 2 sampler slot 0
Set CUBE Texture "unity_SpecCube1" to slot 3 sampler slot -1
Set 3D Texture "unity_ProbeVolumeSH" to slot 4 sampler slot 1

Constant Buffer "$Globals" (112 bytes) on slot 0 {
    Vector4 _LightColor0 at 32
    Float _Glossiness at 64
    Float _Metallic at 68
    Vector4 _Color at 80
}
Constant Buffer "UnityPerCamera" (144 bytes) on slot 1 {
    Vector3 _WorldSpaceCameraPos at 64
    Vector4 _ProjectionParams at 80
}
Constant Buffer "UnityLighting" (768 bytes) on slot 2 {
    Vector4 _WorldSpaceLightPos0 at 0
    Vector4 unity_OcclusionMaskSelector at 736
}
Constant Buffer "UnityShadows" (416 bytes) on slot 3 {
    Vector4 _LightShadowData at 384
    Vector4 unity_ShadowFadeCenterAndType at 400
}
Constant Buffer "UnityPerFrame" (368 bytes) on slot 4 {
    Matrix4x4 unity_MatrixV at 144
}
Constant Buffer "UnityFog" (32 bytes) on slot 5 {
    Vector4 unity_FogColor at 0
    Vector4 unity_FogParams at 16
}
Constant Buffer "UnityReflectionProbes" (128 bytes) on slot 6 {
    Vector4 unity_SpecCube0_BoxMax at 0
    Vector4 unity_SpecCube0_BoxMin at 16
    Vector4 unity_SpecCube0_ProbePosition at 32
    Vector4 unity_SpecCube0_HDR at 48
    Vector4 unity_SpecCube1_BoxMax at 64
    Vector4 unity_SpecCube1_BoxMin at 80
    Vector4 unity_SpecCube1_ProbePosition at 96
    Vector4 unity_SpecCube1_HDR at 112
}
Constant Buffer "UnityProbeVolume" (112 bytes) on slot 7 {
    Matrix4x4 unity_ProbeVolumeWorldToObject at 16
    Vector4 unity_ProbeVolumeParams at 0
    Vector3 unity_ProbeVolumeSizeInv at 80
    Vector3 unity_ProbeVolumeMin at 96
}

Keywords: DIRECTIONAL FOG_LINEAR LIGHTPROBE_SH SHADOWS_SCREEN
-- Vertex shader for "d3d11":
// No shader variant for this keyword set. The closest match will be used instead.

-- Hardware tier variant: Tier 1
-- Fragment shader for "d3d11":
// Stats: 177 math, 13 temp registers, 6 textures, 8 branches
Set 2D Texture "_MainTex" to slot 0 sampler slot 3
Set 2D Texture "_ShadowMapTexture" to slot 1 sampler slot 2
Set CUBE Texture "unity_SpecCube0" to slot 2 sampler slot 0
Set CUBE Texture "unity_SpecCube1" to slot 3 sampler slot -1
Set 3D Texture "unity_ProbeVolumeSH" to slot 4 sampler slot 1

Constant Buffer "$Globals" (112 bytes) on slot 0 {
    Vector4 _LightColor0 at 32

```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		91

```

Float _Glossiness at 64
Float _Metallic at 68
Vector4 _Color at 80
}
Constant Buffer "UnityPerCamera" (144 bytes) on slot 1 {
    Vector3 _WorldSpaceCameraPos at 64
    Vector4 _ProjectionParams at 80
}
Constant Buffer "UnityLighting" (768 bytes) on slot 2 {
    Vector4 _WorldSpaceLightPos0 at 0
    Vector4 unity_SHAr at 624
    Vector4 unity_SHAg at 640
    Vector4 unity_SHAb at 656
    Vector4 unity_OcclusionMaskSelector at 736
}
Constant Buffer "UnityShadows" (416 bytes) on slot 3 {
    Vector4 _LightShadowData at 384
    Vector4 unity_ShadowFadeCenterAndType at 400
}
Constant Buffer "UnityPerFrame" (368 bytes) on slot 4 {
    Matrix4x4 unity_MatrixV at 144
}
Constant Buffer "UnityFog" (32 bytes) on slot 5 {
    Vector4 unity_FogColor at 0
    Vector4 unity_FogParams at 16
}
Constant Buffer "UnityReflectionProbes" (128 bytes) on slot 6 {
    Vector4 unity_SpecCube0_BoxMax at 0
    Vector4 unity_SpecCube0_BoxMin at 16
    Vector4 unity_SpecCube0_ProbePosition at 32
    Vector4 unity_SpecCube0_HDR at 48
    Vector4 unity_SpecCube1_BoxMax at 64
    Vector4 unity_SpecCube1_BoxMin at 80
    Vector4 unity_SpecCube1_ProbePosition at 96
    Vector4 unity_SpecCube1_HDR at 112
}
Constant Buffer "UnityProbeVolume" (112 bytes) on slot 7 {
    Matrix4x4 unity_ProbeVolumeWorldToObject at 16
    Vector4 unity_ProbeVolumeParams at 0
    Vector3 unity_ProbeVolumeSizeInv at 80
    Vector3 unity_ProbeVolumeMin at 96
}

```

Keywords: DIRECTIONAL

-- Vertex shader for "d3d11":

// No shader variant for this keyword set. The closest match will be used instead.

-- Hardware tier variant: Tier 1

-- Fragment shader for "metal":

Set 2D Texture "_MainTex" to slot 0 sampler slot 2

Set CUBE Texture "unity_SpecCube0" to slot 1 sampler slot 0

Set CUBE Texture "unity_SpecCube1" to slot 2 sampler slot -1

Set 3D Texture "unity_ProbeVolumeSH" to slot 3 sampler slot 1

Constant Buffer "FGlobals" (336 bytes) on slot 0 {

Matrix4x4 unity_ProbeVolumeWorldToObject at 192

Vector3 _WorldSpaceCameraPos at 0

Vector4 _WorldSpaceLightPos0 at 16

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		92

```

Vector4 unity_OcclusionMaskSelector at 32
Vector4 unity_SpecCube0_BoxMax at 48
Vector4 unity_SpecCube0_BoxMin at 64
Vector4 unity_SpecCube0_ProbePosition at 80
Vector4 unity_SpecCube0_HDR at 96
Vector4 unity_SpecCube1_BoxMax at 112
Vector4 unity_SpecCube1_BoxMin at 128
Vector4 unity_SpecCube1_ProbePosition at 144
Vector4 unity_SpecCube1_HDR at 160
Vector4 unity_ProbeVolumeParams at 176
Vector3 unity_ProbeVolumeSizeInv at 256
Vector3 unity_ProbeVolumeMin at 272
Vector4 _LightColor0 at 288
Float _Glossiness at 304
Float _Metallic at 308
Vector4 _Color at 320
}

```

Shader Disassembly:

```

#include <metal_stdlib>
#include <metal_texture>
using namespace metal;
#ifdef XLT_REMAP_O
    #define XLT_REMAP_O {0, 1, 2, 3, 4, 5, 6, 7}
#endif
constexpr constant uint xlt_remap_o[] = XLT_REMAP_O;
struct FGlobals_Type
{
    float3 _WorldSpaceCameraPos;
    float4 _WorldSpaceLightPos0;
    float4 unity_OcclusionMaskSelector;
    float4 unity_SpecCube0_BoxMax;
    float4 unity_SpecCube0_BoxMin;
    float4 unity_SpecCube0_ProbePosition;
    float4 unity_SpecCube0_HDR;
    float4 unity_SpecCube1_BoxMax;
    float4 unity_SpecCube1_BoxMin;
    float4 unity_SpecCube1_ProbePosition;
    float4 unity_SpecCube1_HDR;
    float4 unity_ProbeVolumeParams;
    float4 hlscc_mtx4x4unity_ProbeVolumeWorldToObject[4];
    float3 unity_ProbeVolumeSizeInv;
    float3 unity_ProbeVolumeMin;
    float4 _LightColor0;
}

```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				
Змн.	Арк.	№ докум.	Підпис	Дата		93

```

float _Glossiness;
float _Metallic;
float4 _Color;
};

struct Mtl_FragmentIn
{
    float2 TEXCOORD0 [[ user(TEXCOORD0) ]] ;
    float3 TEXCOORD1 [[ user(TEXCOORD1) ]] ;
    float3 TEXCOORD2 [[ user(TEXCOORD2) ]] ;
};

struct Mtl_FragmentOut
{
    float4 SV_Target0 [[ color(xlt_remap_o[0]) ]];
};

fragment Mtl_FragmentOut xlatMtlMain(
    constant FGlobals_Type& FGlobals [[ buffer(0) ]],
    sampler samplerunity_SpecCube0 [[ sampler (0) ]],
    sampler samplerunity_ProbeVolumeSH [[ sampler (1) ]],
    sampler sampler_MainTex [[ sampler (2) ]],
    texture2d<float, access::sample > _MainTex [[ texture(0) ]] ,
    texturecube<float, access::sample > unity_SpecCube0 [[ texture(1) ]] ,
    texturecube<float, access::sample > unity_SpecCube1 [[ texture(2) ]] ,
    texture3d<float, access::sample > unity_ProbeVolumeSH [[ texture(3) ]] ,
    Mtl_FragmentIn input [[ stage_in ]])
{
    Mtl_FragmentOut output;
    float4 u_xlat0;
    float4 u_xlat1;
    bool u_xlatb1;
    float3 u_xlat2;
    float3 u_xlat3;
    float4 u_xlat4;
    float3 u_xlat5;
    float4 u_xlat6;
    float3 u_xlat7;
    float3 u_xlat8;
    float3 u_xlat9;
    bool3 u_xlatb9;
    float3 u_xlat10;
    bool3 u_xlatb11;
    float3 u_xlat12;

```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				94
Змн.	Арк.	№ докум.	Підпис	Дата		

```

float u_xlat13;
float u_xlat24;
float u_xlat25;
float u_xlat36;
float u_xlat37;
bool u_xlatb37;
float u_xlat38;
float u_xlat39;
float u_xlat40;
bool u_xlatb40;
u_xlat0.xyz = (-input.TEXCOORD2.xyz) + FGlobals._WorldSpaceCameraPos.xyzx.xyz;
u_xlat36 = dot(u_xlat0.xyz, u_xlat0.xyz);
u_xlat36 = rsqrt(u_xlat36);
u_xlat1.xyz = float3(u_xlat36) * u_xlat0.xyz;
u_xlat2.xyz = _MainTex.sample(sampler_MainTex, input.TEXCOORD0.xy).xyz;
u_xlat3.xyz = u_xlat2.xyz * FGlobals._Color.xyz;
u_xlatb37 = FGlobals.unity_ProbeVolumeParams.x==1.0;
if(u_xlatb37){
    u_xlatb37 = FGlobals.unity_ProbeVolumeParams.y==1.0;
    u_xlat4.xyz = input.TEXCOORD2.yyy *
FGlobals.hlslcc_mtx4x4unity_ProbeVolumeWorldToObject[1].xyz;
    u_xlat4.xyz = fma(FGlobals.hlslcc_mtx4x4unity_ProbeVolumeWorldToObject[0].xyz,
input.TEXCOORD2.xxx, u_xlat4.xyz);
    u_xlat4.xyz = fma(FGlobals.hlslcc_mtx4x4unity_ProbeVolumeWorldToObject[2].xyz,
input.TEXCOORD2.zzz, u_xlat4.xyz);
    u_xlat4.xyz = u_xlat4.xyz +
FGlobals.hlslcc_mtx4x4unity_ProbeVolumeWorldToObject[3].xyz;
    u_xlat4.xyz = (bool(u_xlatb37)) ? u_xlat4.xyz : input.TEXCOORD2.xyz;
    u_xlat4.xyz = u_xlat4.xyz + (-FGlobals.unity_ProbeVolumeMin.xyzx.xyz);
    u_xlat4.yzw = u_xlat4.xyz * FGlobals.unity_ProbeVolumeSizeInv.xyzx.xyz;
    u_xlat37 = fma(u_xlat4.y, 0.25, 0.75);
    u_xlat38 = fma(FGlobals.unity_ProbeVolumeParams.z, 0.5, 0.75);
    u_xlat4.x = max(u_xlat37, u_xlat38);
    u_xlat4 = unity_ProbeVolumeSH.sample(samplerunity_ProbeVolumeSH, u_xlat4.xzw);
} else {
    u_xlat4.x = float(1.0);
    u_xlat4.y = float(1.0);
    u_xlat4.z = float(1.0);
    u_xlat4.w = float(1.0);
}
u_xlat37 = dot(u_xlat4, FGlobals.unity_OcclusionMaskSelector);
u_xlat37 = clamp(u_xlat37, 0.0f, 1.0f);
u_xlat38 = (-FGlobals._Glossiness) + 1.0;
u_xlat39 = dot((-u_xlat1.xyz), input.TEXCOORD1.xyz);

```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				95
Змн.	Арк.	№ докум.	Підпис	Дата		

```

u_xlat39 = u_xlat39 + u_xlat39;
u_xlat4.xyz = fma(input.TEXCOORD1.xyz, (-float3(u_xlat39)), (-u_xlat1.xyz));
u_xlat5.xyz = float3(u_xlat37) * FGlobals._LightColor0.xyz;
u_xlatb37 = 0.0 < FGlobals.unity_SpecCube0_ProbePosition.w;
if(u_xlatb37){
    u_xlat37 = dot(u_xlat4.xyz, u_xlat4.xyz);
    u_xlat37 = rsqrt(u_xlat37);
    u_xlat6.xyz = float3(u_xlat37) * u_xlat4.xyz;
    u_xlat7.xyz = (-input.TEXCOORD2.xyz) + FGlobals.unity_SpecCube0_BoxMax.xyz;
    u_xlat7.xyz = u_xlat7.xyz / u_xlat6.xyz;
    u_xlat8.xyz = (-input.TEXCOORD2.xyz) + FGlobals.unity_SpecCube0_BoxMin.xyz;
    u_xlat8.xyz = u_xlat8.xyz / u_xlat6.xyz;
    u_xlatb9.xyz = (float3(0.0, 0.0, 0.0) < u_xlat6.xyz);
    {
        float3 hlsicc_movcTemp = u_xlat7;
        hlsicc_movcTemp.x = (u_xlatb9.x) ? u_xlat7.x : u_xlat8.x;
        hlsicc_movcTemp.y = (u_xlatb9.y) ? u_xlat7.y : u_xlat8.y;
        hlsicc_movcTemp.z = (u_xlatb9.z) ? u_xlat7.z : u_xlat8.z;
        u_xlat7 = hlsicc_movcTemp;
    }
    u_xlat37 = min(u_xlat7.y, u_xlat7.x);
    u_xlat37 = min(u_xlat7.z, u_xlat37);
    u_xlat7.xyz = input.TEXCOORD2.xyz + (-FGlobals.unity_SpecCube0_ProbePosition.xyz);
    u_xlat6.xyz = fma(u_xlat6.xyz, float3(u_xlat37), u_xlat7.xyz);
} else {
    u_xlat6.xyz = u_xlat4.xyz;
}
u_xlat37 = fma((-u_xlat38), 0.699999988, 1.70000005);
u_xlat37 = u_xlat37 * u_xlat38;
u_xlat37 = u_xlat37 * 6.0;
u_xlat6 = unity_SpecCube0.sample(samplerunity_SpecCube0, u_xlat6.xyz, level(u_xlat37));
u_xlat39 = u_xlat6.w + -1.0;
u_xlat39 = fma(FGlobals.unity_SpecCube0_HDR.w, u_xlat39, 1.0);
u_xlat39 = u_xlat39 * FGlobals.unity_SpecCube0_HDR.x;
u_xlat7.xyz = u_xlat6.xyz * float3(u_xlat39);
u_xlatb40 = FGlobals.unity_SpecCube0_BoxMin.w < 0.999989986;
if(u_xlatb40){
    u_xlatb40 = 0.0 < FGlobals.unity_SpecCube1_ProbePosition.w;
    if(u_xlatb40){
        u_xlat40 = dot(u_xlat4.xyz, u_xlat4.xyz);
        u_xlat40 = rsqrt(u_xlat40);
        u_xlat8.xyz = float3(u_xlat40) * u_xlat4.xyz;
        u_xlat9.xyz = (-input.TEXCOORD2.xyz) + FGlobals.unity_SpecCube1_BoxMax.xyz;
        u_xlat9.xyz = u_xlat9.xyz / u_xlat8.xyz;

```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				96
Змн.	Арк.	№ докум.	Підпис	Дата		

```

u_xlat10.xyz = (-input.TEXCOORD2.xyz) + FGlobals.unity_SpecCube1_BoxMin.xyz;
u_xlat10.xyz = u_xlat10.xyz / u_xlat8.xyz;
u_xlatb11.xyz = (float3(0.0, 0.0, 0.0)<u_xlat8.xyz);
{
    float3 hlsicc_movcTemp = u_xlat9;
    hlsicc_movcTemp.x = (u_xlatb11.x) ? u_xlat9.x : u_xlat10.x;
    hlsicc_movcTemp.y = (u_xlatb11.y) ? u_xlat9.y : u_xlat10.y;
    hlsicc_movcTemp.z = (u_xlatb11.z) ? u_xlat9.z : u_xlat10.z;
    u_xlat9 = hlsicc_movcTemp;
}
u_xlat40 = min(u_xlat9.y, u_xlat9.x);
u_xlat40 = min(u_xlat9.z, u_xlat40);
u_xlat9.xyz = input.TEXCOORD2.xyz + (-
FGlobals.unity_SpecCube1_ProbePosition.xyz);
u_xlat4.xyz = fma(u_xlat8.xyz, float3(u_xlat40), u_xlat9.xyz);
}
u_xlat4 = unity_SpecCube1.sample(samplerunity_SpecCube0, u_xlat4.xyz,
level(u_xlat37));
u_xlat37 = u_xlat4.w + -1.0;
u_xlat37 = fma(FGlobals.unity_SpecCube1_HDR.w, u_xlat37, 1.0);
u_xlat37 = u_xlat37 * FGlobals.unity_SpecCube1_HDR.x;
u_xlat4.xyz = u_xlat4.xyz * float3(u_xlat37);
u_xlat6.xyz = fma(float3(u_xlat39), u_xlat6.xyz, (-u_xlat4.xyz));
u_xlat7.xyz = fma(FGlobals.unity_SpecCube0_BoxMin.www, u_xlat6.xyz, u_xlat4.xyz);
}
u_xlat37 = dot(input.TEXCOORD1.xyz, input.TEXCOORD1.xyz);
u_xlat37 = rsqrt(u_xlat37);
u_xlat4.xyz = float3(u_xlat37) * input.TEXCOORD1.xyz;
u_xlat2.xyz = fma(u_xlat2.xyz, FGlobals._Color.xyz, float3(-0.220916301, -0.220916301, -
0.220916301));
u_xlat2.xyz = fma(float3(FGlobals._Metallic), u_xlat2.xyz, float3(0.220916301,
0.220916301, 0.220916301));
u_xlat37 = fma((-FGlobals._Metallic), 0.779083729, 0.779083729);
u_xlat3.xyz = float3(u_xlat37) * u_xlat3.xyz;
u_xlat0.xyz = fma(u_xlat0.xyz, float3(u_xlat36), FGlobals._WorldSpaceLightPos0.xyz);
u_xlat36 = dot(u_xlat0.xyz, u_xlat0.xyz);
u_xlat36 = max(u_xlat36, 0.00100000005);
u_xlat36 = rsqrt(u_xlat36);
u_xlat0.xyz = float3(u_xlat36) * u_xlat0.xyz;
u_xlat36 = dot(u_xlat4.xyz, u_xlat1.xyz);
u_xlat1.x = dot(u_xlat4.xyz, FGlobals._WorldSpaceLightPos0.xyz);
u_xlat1.x = clamp(u_xlat1.x, 0.0f, 1.0f);
u_xlat13 = dot(u_xlat4.xyz, u_xlat0.xyz);
u_xlat13 = clamp(u_xlat13, 0.0f, 1.0f);

```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				97
Змн.	Арк.	№ докум.	Підпис	Дата		

```

u_xlat0.x = dot(FGlobals._WorldSpaceLightPos0.xyz, u_xlat0.xyz);
u_xlat0.x = clamp(u_xlat0.x, 0.0f, 1.0f);
u_xlat12.x = u_xlat0.x * u_xlat0.x;
u_xlat12.x = dot(u_xlat12.xx, float2(u_xlat38));
u_xlat12.x = u_xlat12.x + -0.5;
u_xlat24 = (-u_xlat1.x) + 1.0;
u_xlat25 = u_xlat24 * u_xlat24;
u_xlat25 = u_xlat25 * u_xlat25;
u_xlat24 = u_xlat24 * u_xlat25;
u_xlat24 = fma(u_xlat12.x, u_xlat24, 1.0);
u_xlat25 = -abs(u_xlat36) + 1.0;
u_xlat39 = u_xlat25 * u_xlat25;
u_xlat39 = u_xlat39 * u_xlat39;
u_xlat25 = u_xlat25 * u_xlat39;
u_xlat12.x = fma(u_xlat12.x, u_xlat25, 1.0);
u_xlat12.x = u_xlat12.x * u_xlat24;
u_xlat24 = u_xlat38 * u_xlat38;
u_xlat24 = max(u_xlat24, 0.00200000009);
u_xlat39 = (-u_xlat24) + 1.0;
u_xlat4.x = fma(abs(u_xlat36), u_xlat39, u_xlat24);
u_xlat39 = fma(u_xlat1.x, u_xlat39, u_xlat24);
u_xlat36 = abs(u_xlat36) * u_xlat39;
u_xlat36 = fma(u_xlat1.x, u_xlat4.x, u_xlat36);
u_xlat36 = u_xlat36 + 9.99999975e-06;
u_xlat36 = 0.5 / u_xlat36;
u_xlat39 = u_xlat24 * u_xlat24;
u_xlat4.x = fma(u_xlat13, u_xlat39, (-u_xlat13));
u_xlat13 = fma(u_xlat4.x, u_xlat13, 1.0);
u_xlat39 = u_xlat39 * 0.318309873;
u_xlat13 = fma(u_xlat13, u_xlat13, 1.00000001e-07);
u_xlat13 = u_xlat39 / u_xlat13;
u_xlat36 = u_xlat36 * u_xlat13;
u_xlat36 = u_xlat36 * 3.14159274;
u_xlat36 = max(u_xlat36, 9.99999975e-05);
u_xlat12.z = sqrt(u_xlat36);
u_xlat12.xz = u_xlat1.xx * u_xlat12.xz;
u_xlat24 = u_xlat24 * u_xlat38;
u_xlat24 = fma((-u_xlat24), 0.280000001, 1.0);
u_xlat1.x = dot(u_xlat2.xyz, u_xlat2.xyz);
u_xlatb1 = u_xlat1.x!=0.0;
u_xlat1.x = u_xlatb1 ? 1.0 : float(0.0);
u_xlat36 = u_xlat12.z * u_xlat1.x;
u_xlat1.x = (-u_xlat37) + FGlobals._Glossiness;
u_xlat1.x = u_xlat1.x + 1.0;

```

		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				98
Змн.	Арк.	№ докум.	Підпис	Дата		

```

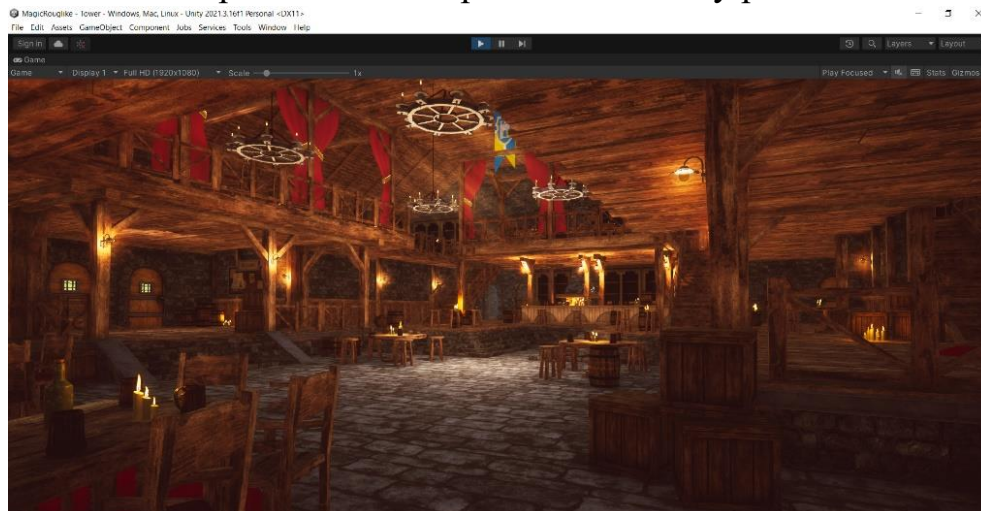
u_xlat1.x = clamp(u_xlat1.x, 0.0f, 1.0f);
u_xlat4.xyz = u_xlat12.xxx * u_xlat5.xyz;
u_xlat5.xyz = u_xlat5.xyz * float3(u_xlat36);
u_xlat0.x = (-u_xlat0.x) + 1.0;
u_xlat12.x = u_xlat0.x * u_xlat0.x;
u_xlat12.x = u_xlat12.x * u_xlat12.x;
u_xlat0.x = u_xlat0.x * u_xlat12.x;
u_xlat6.xyz = (-u_xlat2.xyz) + float3(1.0, 1.0, 1.0);
u_xlat0.xyw = fma(u_xlat6.xyz, u_xlat0.xxx, u_xlat2.xyz);
u_xlat0.xyw = u_xlat0.xyw * u_xlat5.xyz;
u_xlat0.xyw = fma(u_xlat3.xyz, u_xlat4.xyz, u_xlat0.xyw);
u_xlat3.xyz = u_xlat7.xyz * float3(u_xlat24);
u_xlat1.xyw = (-u_xlat2.xyz) + u_xlat1.xxx;
u_xlat1.xyz = fma(float3(u_xlat25), u_xlat1.xyw, u_xlat2.xyz);
output.SV_Target0.xyz = fma(u_xlat3.xyz, u_xlat1.xyz, u_xlat0.xyw);
output.SV_Target0.w = 1.0;
return output;
}

```

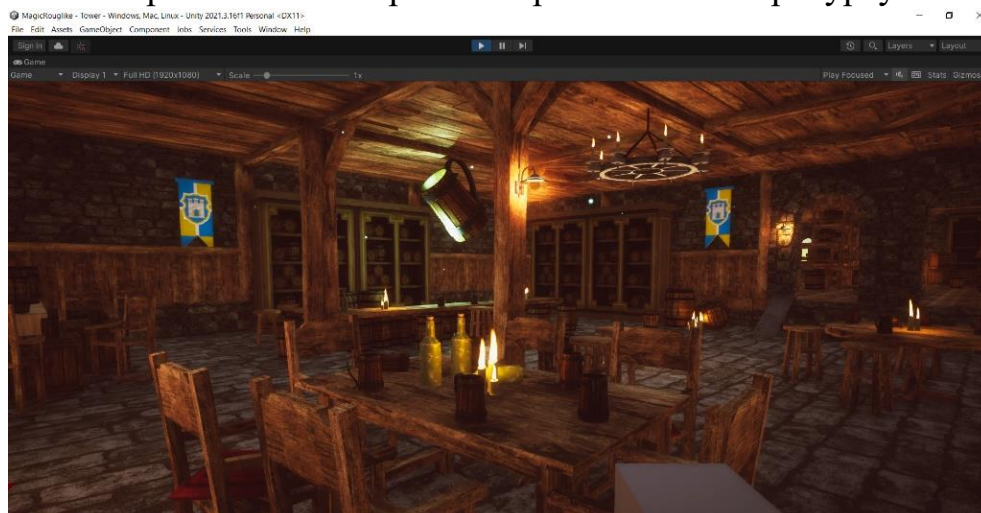
		<i>А. О. ТУРЛІЙ</i>			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		<i>О. В. КУЗЬМЕНКО</i>				
<i>Змн.</i>	<i>Арк.</i>	<i>№ докум.</i>	<i>Підпис</i>	<i>Дата</i>		99

Візуалізація рівнів

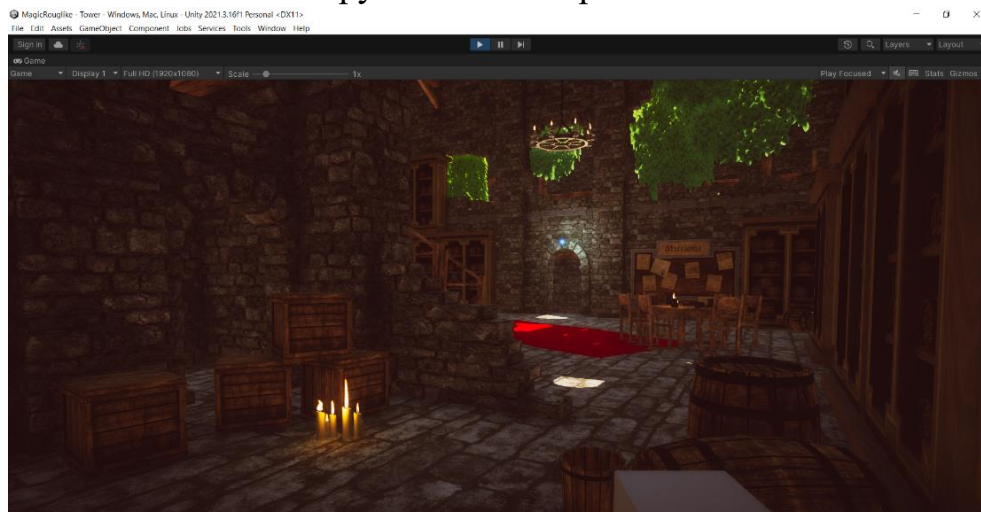
Перспективне зображення початку рівня



Перспективне зображення рівня з іншого ракурсу

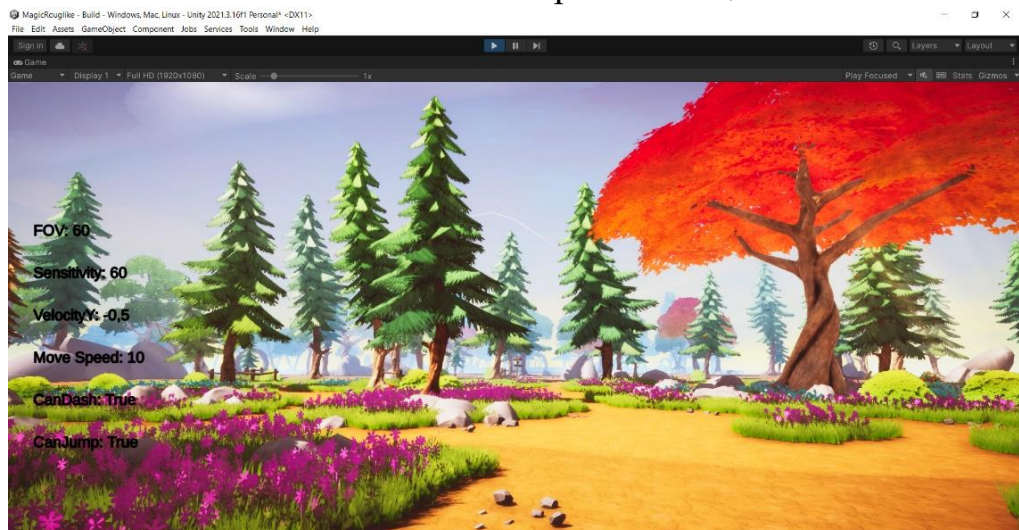


Рівень з зруйнованою вартовою вежею

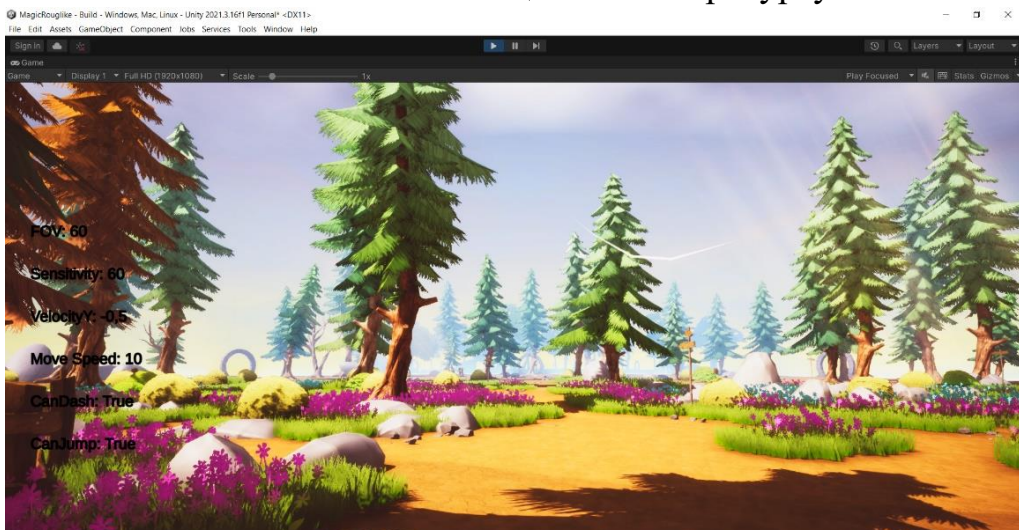


		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				100
Змн.	Арк.	№ докум.	Підпис	Дата		

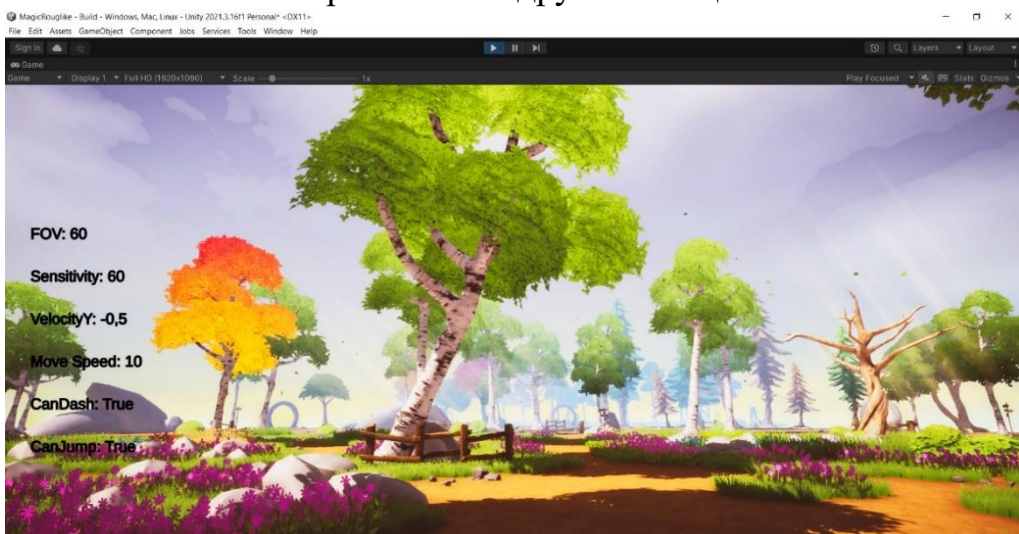
Ялинковий ліс першої локації



Ялинковий ліс локації з іншого ракурсу

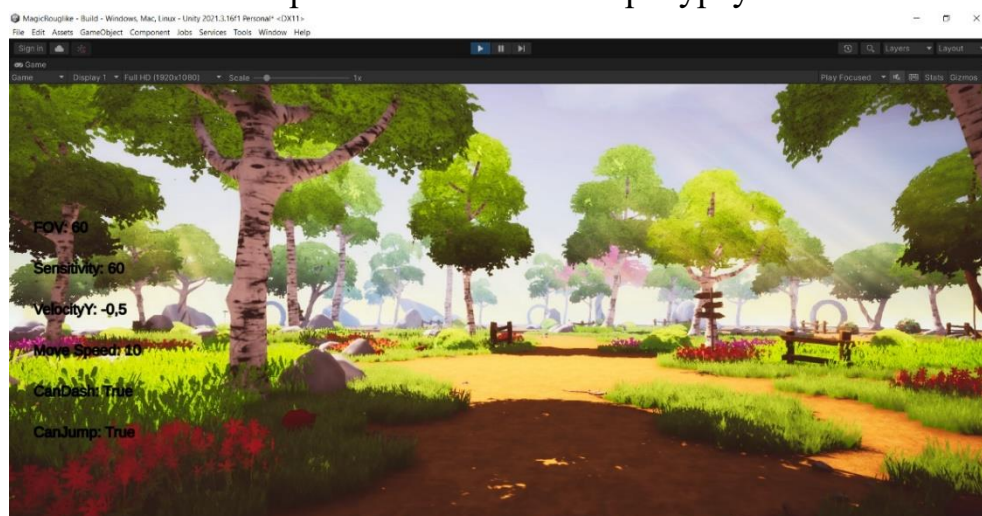


Березовий ліс другої локації

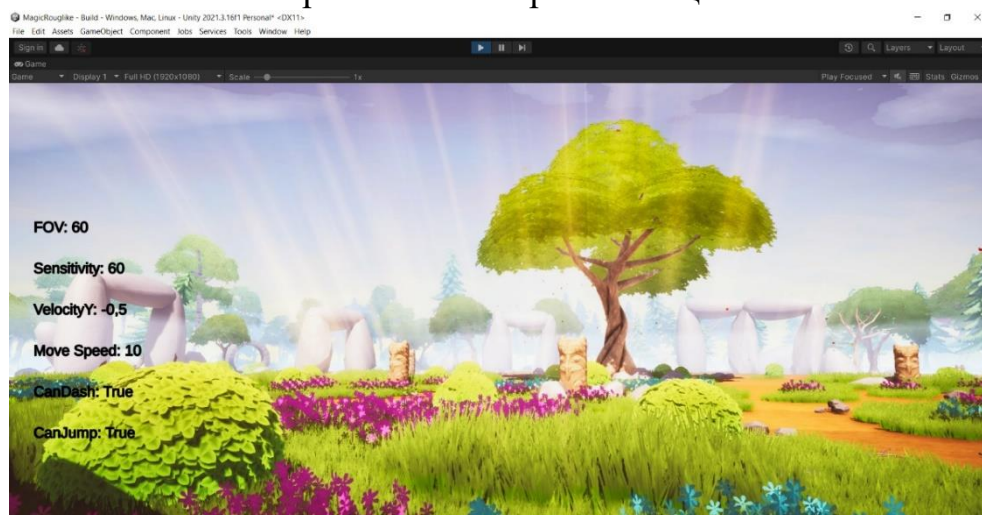


		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				101
Змн.	Арк.	№ докум.	Підпис	Дата		

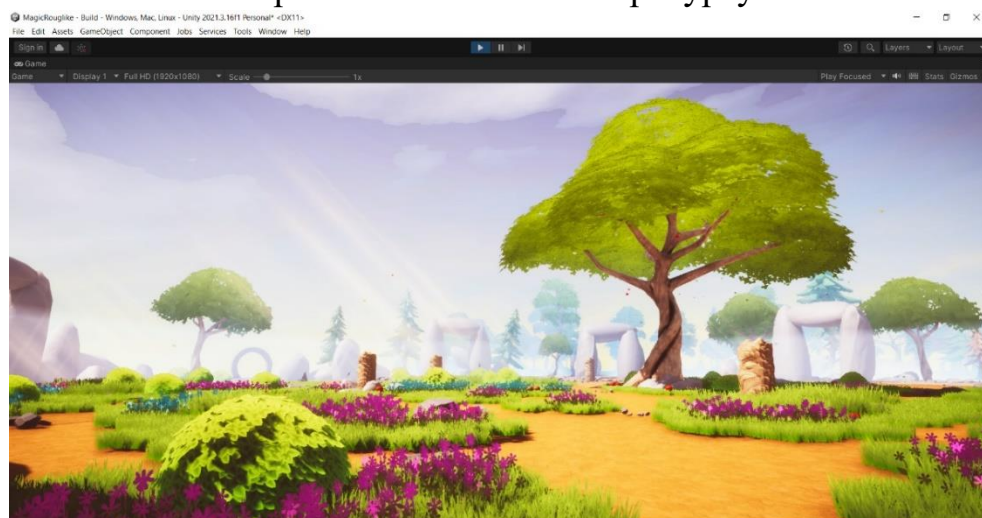
Березовий ліс з іншого ракурсу



Арена з босом третя локація



Арена з босом з іншого ракурсу



		А. О. ТУРЛІЙ			КН.КР.Б – 122 – 24 – ПЗ	Арк.
		О. В. КУЗЬМЕНКО				102
Змн.	Арк.	№ докум.	Підпис	Дата		