Національний університет «Полтавська політехніка імені Юрія Кондратюка»
(повне найменування вищого навчального закладу)
Навчально-науковий інститут інформаційних технологій та робототехніки
(повна назва факультету)
Кафедра комп'ютерних та інформаційних технологій і систем
(повна назва кафедри)

Пояснювальна записка
до дипломного проекту (роботи)

магістра
(рівеньвищої освіти)
на тему
Розробка комп'ютерної гри в жанрі action-adventure

Виконав: студент 6 курсу, групи 601-ТН
спеціальності
122 Комп'ютерні науки
(шифр і назва напряму)
Хавер Е.Ю.
(прізвище та ініціали)
Керівник Руденко О.А.
(прізвище та ініціали)
Рецензент
(прізвище та ініціали)

Полтава – 2021 року

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
«ПОЛТАВСЬКА ПОЛІТЕХНІКА ІМЕНІ ЮРІЯ КОНДРАТЮКА»

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА РОБОТОТЕХНІКИ

КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

спеціальність 122 «Комп'ютерні науки»

на тему

«Розробка комп'ютерної гри в жанрі action-adventure»

Студента групи 601-ТН Хавера Едуарда Юрійовича

Керівник роботи
кандидат технічних наук
Руденко О.А.

Завідувач кафедри
кандидат технічних наук,
доцент Головко Г.В.

Полтава – 2021

РЕФЕРАТ

Кваліфікаційна робота бакалавра: 89 с., 28рисунків, 2 додатки, 31 джерело.

Об’єкт дослідження: процес дослідження, вивчення та розробки гри з використанням ігрового рушія Unity3D.

Мета роботи: розробка та створення концепту та основних механік гри, з метою подальшої реалізації як повноцінного ігрового продукту.

Методи: збір інформації в галузі створенні комп’ютерних ігор, аналіз існуючих жанрів комп’ютерних ігор, програмних продуктів, створення макетів та попереднє планування процесу розробки, послідовна реалізація гри.

Ключові слова: комп’ютерна гра,action-adventure,Unity3D, ігровий рушій.

ABSTRACT

Bachelor's qualification work: 89 pages, 28 drawings,2 appendices, 31 sources.

Object of research: the process of research, study and development of the game using the game engine Unity3D.

Purpose: development and creation of the concept and basic mechanics of the game, in order to further implement as a full-fledged game product.

Methods: collection of information in the field of computer games, analysis of existing genres of computer games, software products, creation of layouts and preliminary planning of the development process, consistent implementation of the game.

Keywords: computer game, action-adventure, Unity3D, game engine.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ПЗ – програмне забезпечення.

АП – апаратне забезпечення.

ПП – програмний продукт.

ЕОМ – електронно-обчислювальна машина.

ШІ– штучний інтелект.

Asset – цифровий ресурс з якого складається гра. До асетів належать звуки, моделі, текстури, сцени та інші елементи, які використовуються у створенні гри.

Shader– комп’ютерна програма, яка використовується графічним процесором відеокарти для створення додаткових візуальних ефектів.

ВСТУП

Історія створення відеоігор почалася в п’ятдесятих роках минулого століття. Звісно, перші ігри досить сильно відрізнялись від тих, що ми бачимо зараз. Через низьку обчислювальну здатність тодішніх ЕОМ ігри являли собою симуляцію певного процесу або явища. Перша комп’ютерна гра була створена Олександром Дугласом у 1952 році та мала назву «OXO». Програма представляла собою реалізацію гри хрестики-нулики. Хоча переважна більшість людей вважають, що першою інтерактивною комп’ютерною грою стала «Теніс для двох». Створена в 1958 році фізиком-ядерником Вільямом Хіггінботамом, який мав на меті розповісти гравцям про дію гравітації, а також розважити їх.Сімдесяті роки стали новим етапом розвитку ігрової індустрії. З’явилися аркадні машини, що додало іграм елемент суперництва серед гравців, адже кожен намагався досягти найкращого результату. Трохи пізніше, на початку вісімдесятих років, у будинках людей почали з’являтися ігрові консолі, а подекуди навіть персональні комп’ютери. Технічний прогрес та вдосконалення апаратних складових дозволили створювати ігри з кращою графікою та більшою кількістю ігрових механік. В середині дев’яностих років почали з’являтися тривимірні ігри, а як результат цього, виникли нові жанри.

Ігрова індустрія продовжує розвиватися і зараз. З кожним роком з’являються нові ігри: від проектів, за створенням якого стоїть лише одна людина, до великомасштабних, де задіяні цілі команди розробників та окремі ігрові видавництва. Різноманіття ігрових продуктів дозволяє кожному знайти гру, яка буде йому до вподоби, а технічний прогрес дозволяє створювати ігрові додатки для різних платформ: від комп’ютера, ігрової консолі до портативних засобів, таких як смартфони. Розвиток апаратного та програмного забезпечення дозволив створювати графіку, яка мало чим відрізняється від кінематографу, активно розвивається напрям доповненої та віртуальної реальності. Ігрові рушії використовують не лише для реалізації

ігор, але і для створення анімаційних фільмів та проектування архітектурних споруд.

За час існування комп’ютерних ігор виникло багато різних категорій та поділів на них, але незалежно від того, в якому стилі виконана гра, для скількох користувачів вона призначена або який жанр було обрано, будь-яка гра має певну мету. Можливо гра виступає як навчальний матеріал, що дозволяє в інтерактивній формі вивчати та досліджувати теоретичну інформацію. Або ж за допомогою гри можна перевірити можливості логічного мислення гравця, його здібності до прийняття рішень. А можливо гра виступає як розвага та дозволяє користувачам даного продукту цікаво та весело провести свій час. Якщо говорити про гру як про один із видів розваг, то сучасні ігри виступають швидше в ролі мосту між іншими розважальними напрямами. Ігри поєднують в собі музику, кінематограф, інколи навіть спортивний елемент, дозволяють фантазувати: подорожувати «новими світами» та досліджувати їх. В іграх присутній інтерактив, що дозволяє гравцям стати «частиною» гри, адже саме від рішень гравця залежить подальший розвиток подій.

Тема дипломної роботи присвячена створенню комп’ютерної гри в жанрі action-adventure. Необхідно провести аналіз існуючих жанрів для визначення характерних особливостей та розглянути доступні програмні продукти, з метою визначення оптимального варіанту та його подальшої реалізації шляхом створення нового ігрового продукту.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ

Комп’ютерна гра являє собою ПЗ. Воно відрізняється від загального розуміння про створення програм, але також являється додатком, що націлений на певні потреби та групи людей. Зацікавленими категоріями скоріше виступають не певні робочі спеціальності, а люди різних вікових груп та класів. Сам процес створення гри не відрізняється від створення інших видів ПЗ. Проводяться усі етапи: аналіз ідеї створення майбутнього проекту, визначаються цілі та завдання, створюються концепти та макети дизайну, додається функціонал, проводиться перевірка створеного ПП. Але окрім цього існують додаткові характеристики, з якими також необхідно визначитись. Сюди відносяться жанр комп’ютерної гри, навколо якої і буде зосереджена робота, ігрові механіки, які є основною складовою функціональних можливостей гри.

Переважна більшість комп’ютерних ігор має наступні компоненти.

- Сюжет – складова гри, що описує ігровий світ, з яким знайомиться гравець. Являє собою «фундамент» проекту з розробки гри. Тематика історії задає напрямок для інших компонентів. Сюжет може мати різну масштабність: в деяких проектах надається мінімум інформації про навколишнє середовище, гравця знайомлять із персонажем або об’єктом, яким він керує та завданням, яке необхідно виконати; в інших, зазвичай великобюджетних іграх, сюжет описує окремий всесвіт із великою кількістю персонажів, територій та подій.
- Звуковий супровід – невід’ємний елемент, який присутній в будь-якій грі в певній мірі. Звукові ефекти формують враження від гри не гірше ніж інші компоненти, тому важливо створити відповідну атмосферу за рахунок правильно підібраних звуків взаємодії гравця з ігровим світом, а також музики, яка створює настрій для ігрових рівнів або локацій.

- Ігровий процес – набір можливих варіантів дій для гравця в середині гри. Набір ігрових можливостей намагаються зробити якомога більшим, але на початку гри користувачу надають лише частину, а протягом проходження сюжету додають нові функції, щоб підтримувати зацікавленість гравця та надати варіативність проходження.
- Графіка та візуальний стиль – даний елемент має найбільшу значимість у формуванні першого враження від гри у користувача. Сучасні комп'ютери дозволяють створювати ігри, які складно відрізнити від реальності в графічному плані.
- Суперники, елементи протидії – сукупність факторів, які чинять гравцю супротив та ускладнюють шлях до фінальної мети.

На завершальному етапі, визначившись із вище вказаними складовими, необхідно проаналізувати доступне ПЗ, в нашому випадку – ігрові рушії, для створення гри.

1.1 Жанри комп'ютерних ігор

Жанр гри визначає які ігрові механіки будуть використані, в якому просторі (дво- чи тривимірному) буде виконана гра, що в свою чергу редагує список графічних стилів, які можна використати. Також, залежно від жанру, звертається різна увага на звуковий супровід: в одних іграх звуки використовуються як додатковий атрибут і є не обов'язковими, в інших – виступають як спосіб орієнтування в просторі та визначення положення ворогів.

1. Action – ігри, головною особливістю яких є сконцентрованість гравця на ігровому процесі.Необхідно вчасно реагувати на постійні зміни. Зазвичай такі ігри мають декілька рівнів складності, або один рівень, що поступово підвищує складність під час проходження гри. Гравець взаємодіє із ігровим світом за допомогою персонажа: це може бути як жива істота: людина, тварина тощо; так і певний предмет, наприклад космічний корабель

або повітряна кулька, все залежить від уяви та вподобань розробника. Гра може складається із певної кількості рівнів, перехід відбувається поступово із одного рівня до іншого. Рівні можуть бути поділені на категорії або групи, так звані «розділи» або «місії». Проходячи рівні гравець повинен вивчати локації, знаходити корисні предмети, що допоможуть вилікувати або вдосконалити персонажа, вирішувати різного типу головоломки та змагатися із ворогами. Персонаж гравця має запас здоров'я – позитивне числове значення, яке може бути відображено різними графічними способами (шкала здоров'я, число або позначки). Також персонаж може мати додаткові навички або пасивні властивості, що дозволяють йому долати різні перешкоди або полегшувати ігровий процес. Дійшовши до завершення рівня або розділу гравець може зіткнутися із більш складним ворогом – босом. Боси мають більший запас здоров'я, ніж звичайні вороги і додаткові навички, що змушує гравця шукати нові варіанти дій для здобуття перемоги. Деякі actionігри можуть складатися лише із одного рівня. Такі ігри не мають завершення, гравець просто намагається встановити якомога кращий результат та порівняти його із досягненнями інших реальних людей. Жанр Actionігор має декілька розгалужень, які відрізняються ігровим процесом [1].

1.1 Shooter – різновид actionігор, основний акцент в яких звертають на можливість здійснювати постріли. Даний напрям з'явився з переходом до тривимірної графіки. Гравець може спостерігати за ігровим процесом з видом «від першої особи», тобто так, як би сам персонаж бачив ігровий світ. Інший розповсюджений варіант – вид «від третьої особи». Камера знаходиться за спиною героя гри, що дає змогу розгледіти модель персонажа. Для зручності існує ігрова механіка, у разі необхідності прицілитися перед пострілом, гравець натискає відповідну клавішу і камера зміщається вперед та фокусується на зброї, персонаж виконує анімацію прицілювання. Особливістю shooterігор є великий арсенал зброї, який відкривається поступово, під час проходження гри. Кожна нова зброя має кращі характеристики або призначена для використання в конкретній ситуації. Для

внесення реалізму та складності в ігровий процес, зброя має обмежену кількість використань – патрони. У разі використання всього запасу, персонаж має зброю ближнього бою, яка є менш ефективною, але також є корисною.

Багато ігор жанру shooterмають кооперативний або багатокористувацький режим. Гравці діляться на дві або декілька команд та намагають виконати поставлені завдання швидше за інших. Яскравими прикладами ігор цього жанру є CounterStrike: GlobalOffensive, PUBG, Overwatch [1].

1.2 Platformer – одна із категорій actionігор, в яких гравець взаємодіє із ігровими об’єктами, що називаються платформами. Це може бути підлога чи земля або об’єкти, які знаходять на певній висоті. Персонаж може рухатись по ним, перестрибувати з однієї на іншу. Існують різні типи платформ: одні дозволяють спокійно знаходитися на них, інші намагаються скинути персонажа, треті мають властивість «пружини» і змушують гравця постійно підстрибувати, що вимагає від гравця правильного розрахунку траєкторії польоту та подальшого пересування по рівню. Platformerігри створюються переважно в 2Dабо 2.5Dстилі. Зазвичай використовується піксельна або мультиплікаційна графіка, яка виділяє даний жанр з-поміж інших. Платформери мають сюжет, традиційно на початку гри розповідається коротка історія, у персонажа з’являється мета і впродовж гри ми намагаємося виконати поставлені завдання. Досліджуючи рівні, головний герой знаходить предмети, що відновлюють його запас здоров’я, дають бонусні ефекти (збільшена швидкість, вищі стрибки, невразливість до атак ворогів на певний час) або дозволяють змагатися із ворогами. Культовими іграми в жанрі platformer є Super Mario Bros, Sonic the Hedgehog, Chip’n Dale: Rescue Rangers, Meat Boy, Cuphead [1].

1.3 Fighting – іще один представник жанру action. Основна ігрова механіка – поєдинок між двома бійцями. Ігри даного жанру дозволяють змагатися як і із штучним інтелектом, яким керує комп’ютер, так і з іншим

гравцем по кооперативу знаходячись поряд або граючи через мережу інтернет. Перед початком двобою гравці повинні обрати персонажа, список яких досить обширний. Кожен персонаж відрізняється по зовнішньому вигляду та бойовому стилю. Обравши персонажів, гравці переходять до поєдинку. Основне завдання – нанести пошкодження персонажу супротивника і намагатися блокувати його атаки. Бій відбувається на ближній дистанції, вогнепальна або дистанційна зброя застосовується рідко, переважно під час виконання спеціальних атак. Жанр fightingдосить динамічний і вимагає від гравців високої концентрації та швидкої реакції. Гравці повинні вчасно реагувати на атаки та протидіяти їм. Ігри цього жанру мають детальну бойову систему, натискаючи відповідні клавіші, гравці дають команди персонажам атакувати певною частиною тіла. Існують комбінації клавіш, правильне натискання яких, дозволяє персонажу здійснити потужну атаку або серію ударів. Вид камери в fightingіграх розміщений перед бійцями, що створює ефект 2D, хоча всі об'єкти в кадрі та персонажі є тривимірними. Сюжет в таких іграх присутній, але не є обов'язковою складовою [1].

1.4 Beat'emup ігри – різновид actionігор, схожий на fighting і має певні властивості platformerігор. Гравцю необхідно перемогти не одного опонента, а хвилі ворогів, які з'являються періодично. Ігри даного жанру складаються з декількох рівнів, які гравець проходить послідовно. Переміщаючись по кожному із них, потрібно вивчати можливі маршрути, знаходити об'єкти із якими можна взаємодіяти. Вороги поділяють на категорії, кожна з яких поступово з'являється під час проходження гри. Відомими іграми цього жанру є Double Dragon, Final Fight, God of War [1].

1.5 Stealth ігри – ігри в яких ігровий процес зосереджений на тихому, непомітному проходженні рівнів та усуненню ворогів. Гравцеві можуть бути поставлені різні завдання, але головна вимога – зробити це не привертаючи уваги. Щоб зробити ігровий процес цікавим, на кожному рівні існує багато варіантів виконання завдання. Гравцю надається можливість досягти мети

різними шляхами або використовувати різні інструменти. Це привносить в гру елемент стратегії та планування дій, адже гравець повинен продумати план дій і альтернативні варіанти на всяк випадок. Початкові рівні в таких іграх є досить простими, щоб розповісти гравцям, як можна взаємодіяти із ігровими об'єктами. Із середини гри починаються випробування, додаються нові механіки. Оскільки існує не один варіант виконання завдання, ігри жанру stealthзмушують гравців проходити кожен рівень або гру в цілому декілька разів, щоб перевірити кожен можливий варіант розвитку подій. Представника даного жанру є Dishonored, Mark of the Ninja, Thief [1].

1.6 Survival ігри– різновид actionігор, головною метою в яких є виживання на невідомій території. В таких іграх існує один або декілька персонажів. Гравець обирає одного із них і розпочинає виживання. У кожного персонажа є інвентар – місце, де він може зберігати ресурси. Початкових ресурсів досить мало, тому гравець повинен вивчати ігровий світ та збирати нові ресурси. Craft–одна із основним механік survivalігор – можливість створювати нові предмети або ресурси. Знайшовши необхідні ресурси, гравець може створити інструмент (сокиру, меч, лопату тощо), який відкриває можливості для отримання нових видів ресурсів, які в свою чергу дозволять створити кращі предмети. Щоб ускладнити процес виживання, в гру додають різні несприятливі ситуації та ворожих істот. Переважна більшість survivalігор не мають фіналу, гравці намагаються вижити якомога довше. Гра завершається якщо помирає персонаж або виконується інша додаткова умова. Ігри даного жанру мають багатокористувацький режим, гравці розподіляють ролі, кожен виконує свою роботу і працює на команду. Подібна співпраця робить ігровий процес цікавим та веселим. Представниками даного жанру є Don'tStarveTogether, Rust, Minecraft, Subnautica [1].

1.7 Rhythm ігри – ігри, в яких гравці повинні відчувати ритм і вчасно виконувати дії. Даний жанр пов'язаний із музикою або танцями. Гравець слідкує за вказівками на екрані та натискає відповідні клавіші або виконує

певні рухи. За правильне виконання гравець отримує бали, а в кінці композиції бачить загальний результат. Даний жанр не потребує від гравця тактичних навичок або вивчення складних комбінацій, але можуть бути потрібні додаткові контролери, які зчитують рухи. Іграми rhythm жанру є Dance Dance Revolution та Guitar Hero [1].

2. Adventureігри – один із основних жанрів комп'ютерних ігор, являють собою інтерактивну історію в центрі якої знаходиться головний герой. Інша назва жанру – квест. Найважливішими елементами гри є розповідь історії та вивчення ігрового світу, а ігровий процес зосереджено на розгадуванні головоломок і задач. Квести мають спокійний темп, такі елементи як динамічні битви, планування маршруту проходження локації та різні задачі, що вимагають від гравця постійної концентрації та вчасної реакції на зміни, не характерні для даного жанру. У персонажа відсутній запас здоров'я і немає прямої загрози життю. Гравець слідкує за розповіддю сюжету, детально вивчає вміст кожного рівня та послідовно вирішує задачі. В подібних іграх гравець виступає скоріше в ролі спостерігача та помічника головного героя гри. Такий підхід нічим не гірший, ніж action, оскільки гравець так само фокусується на подіях, що відбуваються у грі. Жанр adventure має декілька різновидів [1].

2.1 Textadventure – гра, дії в якій виконуються за допомогою команд. Команди являють собою набір дієслів, гравець за допомогою клавіатури вводить інформацію і персонаж виконує певну дію. Гра складається лише з тексту і нагадує книгу, в якій ви обираєте як буде розвиватися історія. Жанр виник ще на початку ери комп'ютерних ігор, зараз він не існує на пряму, а є складовою більш складних ігрових проектів [1].

2.2 Graphicaladventure – гра, в якій необхідно вирішувати головоломки та допомогти герою досягти його мети. Гра складається із багатьох рівнів. Кожен рівень розділено на декілька локацій, на кожній знаходиться безліч предметів, які можна комбінувати. Даний жанр не є динамічним, тому гравцю не потрібно використовувати різні комбінації

клавіш, щоб здійснювати стрибки або спеціальні атаки. Управління виконується за допомогою миші [1].

2.3 Visualnovels– ігровий жанр, що виник в Японії і користується там популярністю. Гра являє собою набір статичний зображень, що передають атмосферу ситуації та інформують про події. Гравець виступає у ролі читача, має змогу обрати одну із відповідей під час діалогу персонажів, що вплине на подальший розвиток сюжету [1].

2.4 Walking simulator – різновид жанру adventure. Гравець може вільно пересуватися по локації, вивчати інформацію, що розкриває сюжетну складову (записи інших персонажів, книги, документи тощо). В такій гри не можливо програти, адже не існує факторів, які б на це впливали. Існує декілька кінцівок гри: хороша, нейтральна, погана (може існувати декілька варіацій кожної категорії), яку гравець отримає, залежить від того, які варіанти дій він обирав в продовж гри [2].

2.5 Interactivemovie– жанр пригодницьких ігор, що поєднує в собі елементи комп'ютерної гри та фільму. В грі присутні декілька головних персонажів, які не знають один одного, кожен розповідає свою історію. Гравця поступово знайомлять із кожним персонажем, іноді головні герої можуть зустрітися за певних обставин. Оскільки гра має складову кінематографу, на екрані відсутні елементи користувацького інтерфейсу. Як і в walkingsimulator, більша частина ігрового процесу це пересування локаціями, діалоги з іншими персонажами, пошук корисних речей. Але відмінна особливість в тому, що персонажі можуть загинути. В грі наявні моменти, коли необхідно швидко реагувати на події інакше існує ризик втратити ігрового персонажа. Гра має декілька кінцівок, в залежності від того, скільки головних героїв змогли вціліти та якій дії виконував гравець протягом проходження гри [2].

3. Action-adventure– жанр комп'ютерних ігор, що поєднує в собі елементи квесту та екшену. Ігри пропонують гравцям долати перешкоди як інтелектуального (різні головоломки, задачі), так і фізичного напряму (бої із

ворогами, пересування по рухомим платформам тощо). До елементів квесту відносять сюжет, персонажів та взаємодію з ними, інвентар для зберігання знайдених предметів. Actionскладовою є битви із противниками, долання різних перешкод, пошук предметів. Кожен жанр доповнює інший, це допомагає створити повноцінний продукт, який зможе як розважити гравця так і перевірити його здібності [2].

4. Role-playing ігри – жанр комп'ютерних ігор, в якому гравець керує одним або декількома персонажами. Кожен персонаж є унікальним, має власне ім'я, зовнішній вигляд та належить до певного класу, який має свої властивості на навички. До ідентифікаційних ознак жанру можна віднести: рольову систему, дослідження ігрового світу, сюжет та бойову систему. Усі персонажі у грі мають характеристики: здоров'я, показник атаки, захисту і класові навички. Також героїв поділяють за дальністю атак, існують стрілки та воїни ближнього бою. Перемагаючи ворогів, гравець отримує досвід, який можна витрачати, щоб вдосконали персонажів та зробити їх сильнішими. Існують такі різновиди Role-playing ігор [2].

4.1. Actionrole-playingігри–піджанр role-playingігор, особливістю якого є бій у режимі реального часу, коли гравець має безпосередній контроль над персонажами і повинен вчасно реагувати на дії ворогів. Бойова система являється комбінацією різних жанрів [2].

4.2. MMORPGігри – багатокористувацька гра в режимі реального часу. Гравець може створювати персонажів, але одночасно керувати лише одним. В грі присутні групи, які називаються гільдіями або фракціями, гравець повинен обрати сторону, яку буде представляти в подальшому. MMORPG ігри постійно оновлюються та проводять тематичні заходи обмежені в часі. Гравці об'єднуються в команди та намагаються виконати поставлені завдання. Гра не має чіткого завершення, гравець не може програти. Якщо персонаж помирає, то через певний проміжок часу він відроджується. Тематика таких ігор стосується переважно фентезійного світу [2].

5. Simulationігри – ігри, що відображають певний процес або дію. Симуляції можуть стосуватися певної роботи або життя в цілому. Сюжет відсутній, гравець сам створює історію. Досить гнучкий жанр і способи реалізації залежать від розробників. Ігри даного жанру мають економічний аспект, гравець повинен правильно використовувати наявні ресурси. Simulationігри дозволяють проводити різноманітні експерименти. Гравець виступає у ролі творця, він визначає майбутнє персонажів, що перебувають у симуляції [2].

5.1 Constructionandmanagementsimulation – ігри, в яких користувач приміряє на собі роль керівника. Ви можете стати мером міста або директором підприємства. Головна мета – досягти успіху, збудувати та вдосконалити свою організацію. Дані ігри не потребують від гравця приймати швидкі рішення, ви завжди можете оцінити усі варіанти та обрати кращий, не поспішаючи [2].

5.2 Lifesimulation – ігри, що імітують життя. Гравець керує одним або кількома віртуальними персонажами (людиною чи іншою істотою). Така гра може обертатися навколо певних соціальних процесів, або симулювати цілу екосистему. Даний жанр є більш динамічним, потрібно постійно слідкувати за своїми персонажами чи популяцією[2].

5.3 Vehiclesimulation – ігри, в яких гравець керує певним транспортним засобом. Це може бути літак, корабель, вантажівка тощо. Гравець має одне завдання – виконувати перевезення. Дані ігри дозволяють спробувати робочі професії, які існують в реальному житті. Від гравця не потребують розумових зусиль та постійної концентрації на ігровому процесі [2].

6. Strategyігри – один із найстаріших жанрів. Стратегії переплітаються із іншими жанрами, але властивість по яким можна чітко визначити, що це саме стратегія – детально продуманий план дій. Гравець повинен реагувати на дії опонента, аналізувати можливі варіанти розвитку ситуації, намагатися зменшити ризик зазнати поразки [2].

6.1. Towerdefense – гравець повинен захищатися від хвиль ворогів за допомогою веж. На початку гри доступна певна кількість ресурсів, яка дозволить побудувати перші вежі, після чого починають наступати вороги. Вежі атакують ворогів та отримують ресурси за знищених противників. Гравець повинен будувати додаткові вежі, вдосконалювати уже існуючі. Якщо вдається пережити усі хвилі, гравець перемагає [2].

6.2. Multiplayer online battle arena (MOBA) – ігровий жанр, що поєднує в собі action, role-playing і strategy. Не потрібно будувати базу чи збирати ресурси. Ігровий процес відбувається на арені. Гравці діляться на дві команди, обирають персонажів та починають змагання. Перемагає команда, яка здолає суперників [2].

Проаналізувавши список доступних жанрів, оптимальним варіантом буде action-adventureгра. Жанр має всі необхідні елементи, які потрібні для створення цікавого ігрового продукту. В грі можна поєднати бійки, проходження небезпечних частин рівнів гри та додати сюжет, що буде відповідати за цілі в грі. Наступним кроком буде аналіз існуючих ігрових рішень, що дозволить виокремити цікаві механіки та інтерпретувати їх у свій проект.

1.2 Огляд існуючих ігрових продуктів жанру action-adventure

1. GodofWar – однокористувацька action-adventure гра, розроблена студією Santa Monica і опублікована компанією Sony Interactive Entertainment (SIE).

Положення камери. Вид камери у грі «від третьої особи», гравці можуть бачити головного героя. Положення камери зафіксовано позаду головного героя, але гравці можуть обертати камеру та оглянути героя з усіх сторін.Гра виконана в кінематографічному стилі, без перемикань чи переходів на вид із інших камер, завантаження екранів також відсутні.

Ігровий процес. Ігровий світ досить обширний, гравець може вільно пересуватися по ньому. Існують закриті ділянки, які можна відкрити спеціальними предметами, що з'являться під час проходження гри.В грі присутня можливість швидкої подорожі між ключовими локаціями гри. Протягом усієї гри гравці борються зі скандинавськими міфологічними ворогами:темними ельфами, вульверами, драугам та іншими істотами. Кожний вид ворогів має свої здібності і тактику бою. Валькірії являються босами, але з ними не обов'язково битися. В грі присутні сторонні завдання. Виконуючи їх, гравці отримують ресурси для вдосконалення персонажа.

Бойова система. Бойова система складається із різноманітних атак, правильна послідовність котрих створює комбінації, що наносять більше пошкоджень ворогам. Головний герой (Кратос) має зброю (сокиру), яку використовує як в ближньому так і дальньому бою. Сокиру можна кинути в ворогів і повернути назад неначе бумеранг. Зброю також можна кидати в ігрові об'єкти, щоб відкрити шлях або створити вибух. З часом зброю можна модернізувати. Це дозволяє гравцям обирати власний стиль гри. Окрім звичайної зброї персонаж має особливі навички. Одна із них SpartanRage. Навик має шкалу, яка заповнюється під час бою із ворогами.Активувавши вміння, головний герой використовує потужні комбо-атаки, щозавдає значніпошкодження ворогам.

Вдосконалення під час гри. У грі також присутні елементи рольової відеоігри.Подорожуючи світом, гравець знаходить ресурси, які дозволяють йому створювати нові види броні або модернізувати уже створені. Гравці також накопичують валюту під назвою hacksilver – ресурс, що дозволяє придбати нові предмети. Бали досвіду (XP) використовуються для вивчення нових бойових навичок. Окрім Кратоса в грі присутній іще один персонаж – Атрей. Гравець не може на пряму керувати ним, але він слідкує за Кратосом та допомагає йому під час битв.

2. Batman: ArkhamCity– цепригодницька гра, розроблена компанією Rocksteady Studios і видана Warner Bros. Interactive Entertainment.

Положення камери. Вид камери «від третьої особи». Камера знаходиться за спиною головного героя. Гравці можуть рухати положення камери навколо персонажа. Існують локації та скриптові сцени в яких відбувається перехід до виду «від першої особи». В грі присутні відеоролики, виконані в кінематографічному стилі. Перехід між локація відбувається шляхом затемнення екрану під час завантаження.

Ігровий процес. Гра з відкритим світом, що включає в себе тактику із стелс-ігор. Дія гри відбувається в Аркхем Сіті – імпровізаційному невеликому містечку всередині іншого міста. Із самого початку гравець може вільно пересуватись локацією.Гравець має багато варіантів проходження певної ситуації, використовуючи комбінацію гаджетів та бойову систему. Окрім відкритого бою, існує спосіб проходити рівні не піднімаючи тривоги. Головний геройможе пересуватися не лише на ногах, але і використовувати свій плащ для польоту. Перед цим необхідно піднятися якомога вище, щоб політ тривав довше. Особливою ігровою механікою є «детективне бачення», гравець може використовувати його коли необхідно оцінити ситуацію: визначити положення ворогів, знайти необхідний шлях, або відтворити місце злочину за траєкторією руку предметів. Гравець має доступ до внутрішньо-кримінальної бази даних, яка включає в себе криміналістичні головоломки, а також мережу для злому частот спілкування.

Бойова система.Використовуючи поліпшену версію бойової системи Freeflowяка була і в попередній грі серії Arkham, гравець тепер може блокуватиудари кількох ворогів одночасно, ловити метальні снаряди, атакувати з повітря і виконувати безперервні комбінації атак. Більшість гаджетів Бетмена тепер можна використовувати в бою. Вороги озброєні різними рівнями броні і зброї.Атаки від основної зброї, такої як бейсбольні біти і металеві труби, наносять незначні пошкодження і можуть бути заблоковані.Вогнепальна зброя завдає значні пошкодження. Деякі вороги повинні бути роззброєні, перш ніж їх можна буде нейтралізувати в бою: ворогів із електрошокерами можна атакувати тільки ззаду, вороги зі щитами

вимагають повітряних атак для роззброєння, а вороги, що носять бронежилети, повинні бути оглушені швидкими послідовними ударами. Що здолати більших за розмірами ворогів, гравець повинен комбінувати різні атаки. Такі вороги мають декілька етапів, після проходження кожного з них, гравець може на короткий час отримати контроль над ворогом і за його допомогою нанести пошкодження іншим опонентам.За успішно проведений бій, гравець отримує бали досвіду, які дозволяють вдосконали персонажа, відкрити нові атаки або збільшити запас броні.Деякі гаджети, отримані в Batman: Arkham Asylum (попередня гра серії), присутні на початку Arkham City, в той час як інші стають доступними під час гри. Більшість з них мають поліпшені або нові можливості; наприклад, криптографічний секвенсор, який використовується для злому консолей безпеки, може також контролювати короткохвильові радіоканали.Тепер пускова установка лінії може бути розгорнута як канат або змінити напрямок під час польоту. Інші предмети, що повертаються з першої гри, включають в себе: дистанційно керований бетаранг, вибухонебезпечний гель, який тепер можна використовувати під час бою, щоб збити ворогів з ніг. Нові предмети в арсеналі Бетмена включають: димові гранати, які дезорієнтують супротивників і допомагають сховатися в безпечному місці; пістолет з дистанційним електричним зарядом (REC), який може оглушувати ворогів і тимчасово приводити в рух мотори; гранати FreezeBlast, які дозволяють тимчасово зупинити ворогів або можуть бути кинуті в воду для створення саморобних платформ;Disruptor, який може дистанційно відключати зброю і вибухові міни.

Гра має список головних завдань, а також додаткові, виконання яких дає гравцеві додаткові бали досвіду. Побічні місії можна виконувати в будь-який час, вони відрізняють за тематикою та потребують від гравця різних навичок.

3. Dishonored–гра жанру action-adventure, розроблена ArkaneStudios і видана BethesdaSoftworks.

Положення камери. Гра«від першої особи». Гравець не може побачити все тіло головного персонажа, а лише руки. Проте сам персонаж згадується в ігрових предметах (картинах та малюнках) та відеороликах.

Ігровий процес. Гра сконцентрована на непомітні дії, використання гаджетів і навколишнього середовища для усунення поставлених цілей. Ігровий світ являє собою серію окремих, логічно поєднаних місць, які мають безліч шляхів дослідження з точки зору ігрового руху і здібностей. Між місіями гравець потрапляє на нейтральну територію пабHoundPits, де персонаж гравця на ім’я КорвоАттано може зустрічатися зі своїми союзниками, отримувати брифінги та альтернативні цілі, а також обмінювати добуту здобич в нове спорядження або модернізувати уже існуюче. В ігрову зону входять вантажні доки, королівські маєтки, бідні вулиці та неприступна фортеця. Гра включає в себе систему збереження контрольних точок, проте гравець може зберегти свій прогрес в будь-якому місці, для безпечного вивчення невідомих територій. Збереження відключено під час бою. У грі є чотири рівні складності, які змінюють ефективність зілля здоров’я і мани (магії), а також силу пошкоджень від ворогів і їх здатність побачити головного героя в потаємних місцях.Dishonored містить елементи рольової гри, такі як здатність покращувати свої здібності і робити моральний вибір з акцентом на нелінійні наслідки. Гра побудована так, щоб дозволити гравцеві завершити її, не вбиваючи неігрових персонажів (NPC), включаючи босів і цілі місії. Кожна місія містить кілька способів досягнення цілей. Переміщення і дослідження рівнів призначене для підтримки здібностей персонажа гравця, а не конкретних шляхів, націлених на певний стиль гри.В деяких місіях ігрові об’єкти генеруються випадковим чином, що змушує гравця вивчати територію кожного разу.Дії гравця не оцінюються як добрі чи злі, а замість цього відслідковуються системою "хаосу", яка записує кількістьнасильства і смертей, спричинених гравцем. Це змінює ігровий світ, впливаючи на історію, не караючи гравця і не змушуючи його вибирати один стиль гри замість іншого. Наприклад, NPC, який не схвалює насильство,

може відмовитися підтримати гравця або навіть може зрадити його. Гра реагує на хаос, викликаний діями гравця: відбуваютьсязміни діалогів, збільшується кількість присутніх щурів і переслідуваних громадян, додаються нові сцени. Це може вплинути на активну місію і майбутні місії. Система також впливає на те, яка із двох кінцівок гри буде обрана, спираючись на те, які персонажі залишились в живих. Використання насильства дозволяє виконувати місії за менший час, ніж використання прихованого підходу, але насильство потребує більше внутрішньоігрових ресурсів, таких як зілля здоров'я і мани, які потрібні частіше у відкритому бою.

Бойова система. В грі бойова система реалізована як додаткова складова, яку можна оминути. Якщо гравець проходить гру, не вбиваючи жодну ціль, то він використовує переважно вміння та бойовий прийом, що нейтралізує ворогів, але не вбиває їх. У разі вибору «агресивного» проходження бойова система стає головною складовою, оскільки гравець повинен протидіяти великій кількісті ворогів. Надприродні здібності та далекобійна зброя знаходять у лівій руці персонажа, а меч у правій, що дозволяє поєднувати більшість вмінь та робити ігровий процес динамічним, не витрачаючи багато часу на постійне перемикання одного виду арсеналу на інший. Зі збільшенням хаосу, збільшується і кількість ворогів, що ускладнює ігровий процес і іноді все ж краще обрати безшумний варіант, ніж вступати в поєдинок із великою кількістю ворогів. Окрім битв із ворогами, гравець може використовувати зброю щоб потрапити у потаємний сховок, або відкрити певний прохід.

Вдосконалення під час гри. В Dishonored гравецьволодіє шістьма активними та чотирма пасивними здібностями. Окрім цього в грі можна знайти сорок кістянихамулетів, які дають гравцеві надприродні можливості.Гравець може використовувати лише від трьох до шести кістяних амулетів одночасно, тому в залежності від ситуації, він може змінювати їх, одягаючи потрібні. Це привносить в гру елемент тактики та

планування. Щоб використовувати надприродні здібності гравцеві потрібна мана.Вона частково відновлюється після використання навиків, що дозволяє використовувати сили через певний період часу. Більш складні вміння вимагають від гравця використовувати зілля мани, щоб поповнити запас магічної енергії. Основні надприродні здібності відкриваються і купуються за допомогою рун. Після відкриття кожне вміння можна поліпишити.Одне із основних вмінь «Темне бачення», яке дозволяє гравцеві бачити ворогів крізь стіни, їх поле зору і підсвічувати інтерактивні об’єкти. «Перенесення» – здатність до телепортації на короткі відстані; «Володіння», яке дозволяє гравцю тимчасово знаходитисяв тілі іншої живої істоти. Гравець може використовувати зброю, включаючи меч, гранати, арбалет і пістолети. Монети виступають в ролі валюти, за яку можна придбати нову зброю або вдосконали існуючу.

Головні складові гри жанру action-adventure.

- Сюжет – історія, яка розповідає про вигаданий світ та знайомить гравця із персонажами.
- Завдання – гравець постійно має певну мету. Зазвичай одна масштабна ціль поділена на етапи, щоб полегшити досягнення основної.
- Багаторівневість – гра складається із багатьох локацій, що відрізняються візуально та за наповненістю предметами. Одні рівні виступають в ролі контрольних, де гравець може поповнити запас ресурсів, інші виступають в ролі місця, де відбуваються найцікавіші події.
- Наявність персонажів – гра повинна містити персонажа, який поставлений в центр подій – головного героя, яким керує гравець. Окрім нього повинні бути додаткові персонажі, вони можуть бути як позитивними та і негативними (вороги). Вороги поділяються на категорії. Звичайні вороги є не складними і намагаються перемоги гравця кількісно. Боси мають спеціальні властивості та складніші в цілому, тому поєдинок відбувається один на один.

- Бойова система – гравець взаємодіє із ворожими персонажами, зазвичай це реалізовано саме за допомогою поєдинків. Персонаж має зброю, переважно ближнього бою.

1.3 Порівняння ігрових рушіїв

Ігровий рушій– це програмне забезпечення, яке надає набір необхідних функцій для швидкого і ефективно створення комп'ютерних ігор.У багатьох випадках ігрові рушії надають набір інструментів візуальної розробки на додаток до часто використовуваних програмних компонентів. Ці інструменти надаються в інтегрованому середовищі розробки, що дозволяє спростити і прискорити розробку ігор на основі уже готови пакетів даних. Розробники ігровихрушіїв намагаються постійно вдосконалювати свій продукт, розробляючи надійні програмні пакети, які включають безліч елементів, які можуть знадобитися розробникам гри. Більшість ігрових рушіїв надають засоби, які полегшують розробку, такі як графіка, звук, фізика та функції штучного інтелекту. Переважна більшість рушіїв не прив'язана до конкретної платформи і дозволяє запускати гру на ігрових приставках і персональних комп'ютерах. Найчастіше ігрові рушії розробляються на основі компонентної архітектури, яка дозволяє замінювати або розширювати певні елементи в ПЗ, що робить його універсальним засобом розробки. Незважаючи на специфіку назви, ігрові рушії часто використовуються для інших видів інтерактивних додатків з графічними потребами в реальному часі, такими як маркетингові демонстрації, архітектурна візуалізація, моделювання та навчання [3].

Найвідомішими ігровими рушіями є CryEngine, UnrealEngine, Unity3D. Проведемо аналіз ПП за такими критеріями: основна інформація, модель розповсюдження, мови програмування, доступний функціонал [3].

1. CryEngine – ігровий рушій, створений німецькою приватною компанією Crytek в 2002 році. За час існування отримав велику кількість оновлень та модифікацій. Остання на даний момент версія CryEngineV.

Модель розповсюдження. Сам рушій та додатковий набір інструментів розповсюджуються безкоштовно. Із запуском CryEngine 5.5 розробники заявили про зміну моделі розповсюдження. Відтепер користувачі даного рушія повинні виплачувати п’ять відсотків від прибутку, якщо їх дохід перевищує суму в п’ять тисяч доларів. У разі, якщо розробники ігор використовують старі версії CryEngineта не планують переходити до версії 5.5 чи вище, вони мають змогу відмовитися від виплат, надіславши відповідну заяву [4].

Мови програмування. CryEngineдозволяє писати скрипти на C++. В CryEngineVз’явилась можливість використовувати C# [4].

Функціонал. CryEngine являє собою програмне забезпечення для повноцінної розробки гри. Він підтримує ряд графічних можливостей.

1. Робота зі світлом.

- AreaLights – тип джерела світла, що дозволяє створювати світло у вигляді певної фігури та займати необхідну площу. В порівнянні із звичайними точковими джерелами, дозволяє створювати світлове шоу та робити акцент на потрібних місцях.
- Voxel-BasedGlobalIllumination (SVOGI) – дозволяє створювати карти світла.
- Per-ObjectShadowMaps – можливість будувати карту тіней, які прив’язують до певного об’єкту.
- ScreenSpaceDirectionalOcclusion – створення додаткового об’єму в місцях перетину світла та тіні.
- ImageBasedLighting – створення освітлення на основі зображення, що дозволяє швидко розмістити джерела світла на рівні.
- PhysicallyBasedRendering – функція дозволяє створювати реалістичні поверхні за допомогою системи освітлення PBR. Модель

рендеринга імітує взаємодію світла і поверхні матеріалів з використанням реальної фізики. Симуляція поведінки світла в реальному часістворює реалістичні ефекти.

- Real-TimeDynamicWaterCaustics – модуль для створення реалістичних водних ефектів. Спеціально створені шейдери дозволять отримати неперевершений результат.
- Tessellation – функція, що дозволяє створити додаткову об’ємність предметів. CryEngine підтримує апаратне прискорення тесселяції для всіх сіток, включаючи анімованих персонажів. Підтримуються три різних типи тесселяції: Фонг, трикутники PN і відображення зсуву.
- 3D HDRLensFlares – модуль для роботи із ефектом світла. Дозволяє додавати блиск сонячного світла і різні кольорові градієнти.
- PostProcessing – додає реалістичні ефекти, робить гру красивішою. До основних атрибутів належать:MotionBlurandDepthofField, Real-TimeLocalReflections, HDR Filmic Tone Mapping [4].

2. Додаткові інструменти.

- Sandbox – універсальний інструмент для створення контенту. Надає розробникам повний контроль над їх мультиплатформенними розробками в режимі реального часу, пропонуючи безліч ефективних інструментів, які забезпечують швидку ітерацію при розробці ігрових світів. Набір інструментів виконаний за принципом WYSIWYG – «Що бачиш, те й отримуєш» дозволяє в реальному часі створювати, редагувати і переглядати в грі всі аспекти і функції гри.
- MaterialEditor – інструмент для роботи із моделями та матеріалами. Набір шейдерів дозволяє створювати реалістичні матеріали. Окрім цього існує можливість створити власні шейдери.
- FBX Support – підтримка цифрового контенту. Створюйте моделі у форматі fbx та з легкістю імпортуйте до свого проекту.
- Flowgraph – система візуального сценарію. Дозволяє створювати ігрову логіку та управляти подіями в ігровому рівні [4].

3. Робота із штучним інтелектом та анімаціями.

- CharacterTechnology – технологія створення анімації персонажів. Дозволяє створювати реалістичну міміку персонажів, додавати ефект руху волосся та багато іншого.
- ParametricSkeletalAnimation – набір параметрів, що дозволяють створювати покрокову анімацію.
- AdvancedAISystem – система дозволяє створювати сценарії дій для неігрових персонажів.
- PhysicalizedCharacterCustomization – можливість додавати персонажу додаткові візуальні атрибути. Це може бути перстень, пояс тощо. Ці предмети являють собою моделі, тому дана функція дозволяє комбінувати ігрові об’єкти та прораховувати фізику їх взаємодії.
- ProceduralMotion-Warping&High-EndIKSolutions дозволяє полегшите життя аніматорам за допомогою процедурних рішень. Процедурні алгоритми, такі як CC D-IK, аналітичний IK, IK на основі прикладів або фізичне моделювання, використовуються для розширення попередньо створених анімацій, щоб уникнути типового згенерованого комп’ютером виду. Масштабна технологія деформації CryEngine зберігає стиль і зміст базового руху, повністю беручи до уваги обмеження, що накладаються інтерактивним середовищем [4].

4. Робота із аудіо.

- AudioControlsEditor (ACE). Цей потужний інструмент забезпечує швидке, гнучке створення і підключення подій, перемикачів, станів і RTPC, а також детальне управління файлами, що попередньо були завантажені до Soundbanks. Зміни в ACE оновлюються в режимі реального часу з миттєвим відтворенням звуку, перезапуск програми не потрібен. Редактор управління звуком (ACE) має повнофункціональний компонент пошуку та фільтрації, що дозволяє швидко і просто знайти, переглянути або редагувати навіть велику кількість звукових подій.

- AudioAbstraction. Безпрецедентна свобода вибору проміжного програмного забезпечення. CryEngine агностично зв’язується з будь-яким проміжним аудіо ПЗ, яке використовують для роботи зі звуком. Наприклад CRI ADX2, FMOD Studio, Miles Sound System або Wwise.
- HRTF AudioSpatialization – можливість використовувати тривимірну обробку звуку в реальному часі для забезпечення точних реальних акустичних властивостей поширення звуку.
- AudioComponents – зручний доступ до всіх аудіофункцій через аудіокомпоненти
- DynamicResponseSystem (DRS) – система динамічного реагування (DRS) представляє новий спосіб реалізації діалогів в грі. Він поєднує відносини діалог-зворотний зв’язок і ігрову логіку. DRS досягає цього, додаючи абстрактний шар між тим, що відбувається, і діє зворотним зв’язком діалогу. DRS також можна використовувати для використання будь-якої іншої системи, такої як, наприклад, анімація, система частинок або фізика[4].

5. Робота з фізикою.

- Physics – модуль для відображення фізичних властивостей істот, предметів та явищ під час гри, починаючи від елементів природи і закінчуючи вибухами та руйнуваннями. Модуль забезпечує підтримку широкого спектру функцій, включаючи довільні сітки на динамічних об’єктах і спеціальний режим моделювання для поєднаних конструкцій.
- VegetationTouchBending – реалізація в ігровому середовищі об’єктів рослинного світу. Можливість створювати рослини та сценарії взаємодії з іншими ігровими сутностями.
- Built-inBuoyancyandWaterSimulation – реалізація фізичних властивостей води, повітря та довільних об’єктів. Створення реалістичних симуляцій для різних обсягів водної площі: від річок до океанів.
- VariousDestructionModels – функція, що дозволяє створювати логіку поведінки для частин зруйнованого об’єкту.

- AdvancedRopes – інструмент для роботи із мотузками та ланцюгами. Дозволяє створювати реалістичну симуляцію в поєднанні із явищами природи[4].

6. Перевірка продуктивності проекту.

- In-GameProfiling – дозволяє знайти проблеми з продуктивністю прямо під час роботи додатку. CryEngine озброєний мітками профілю всюди. Крім того, інформація про виникненні проблеми та їх тривалість допомагає оцінити їх вплив на проект та дозволяє уникнути їх в майбутніх проектах.
- Data-Driven thread management – система управління потоками. Незалежна від платформи, система потоків CryEngine створює умови для ефективного розподілу роботи на програмні і апаратні потоки. Менеджер, який зберігає в собі всю інформацію дозволяє планувати і контролювати паралельні обчислення під час виконання.
- Statoscope – внутрішній діагностичний інструмент, що використовується для доступу до інформації, де витрачаються характеристики рушія. Модуль визначає проблеми із продуктивністю, детально розглянувши, як працює гра. Користувач може відслідковувати за інформацією в режимі реального часу або зберегти інформацію, записану в модуль, в окремий файл та переглянути пізніше. Окрім цього, користувач може провести одразу два тести, що перевірити, з якими налаштуваннями гра працює краще[4].

2. UnrealEngine – ігровий движок, що розробляється і підтримується компанією EpicGames.

Модель розповсюдження. До 2015 року розповсюджувався на основі щомісячної підписки. Починаючи із 2015 року став безкоштовним, однак, якщо дохід від гри становить більше трьох тисяч доларів, розробники повинні передавати п’ять відсотків компанії EpicGames [5].

Мови програмування. Рушій підтримує мову програмування C++ а також візуальний скриптинг BluePrint[5].

Функціонал.

1. UnrealEnginemає багато інструментів для створення основних елементів ігрового рівня та їх редагування.

- TheUnrealEditor. UnrealEngine включає в себе UnrealEditor, інтегроване середовище розробки, доступне в Linux, MacOS і Windows для створення контенту. Завдяки підтримці багатокористувацького редагування художники, дизайнери і розробники можуть одночасно вносити зміни в один і той же проект UnrealEngine безпечним і надійним способом.
- Scalablefoliage – інструмент для автоматичного покриття величезних територій та об'єктів різними типами трави, квітів, невеликих каменів за допомогою інструменту «Трава», і створюйте великі ліси, заповнені безліччю різних видів дерев.
- Assetoptimization – підготовка та оптимізація складної моделі для підвищення продуктивності. UnrealEngine пропонує такі інструменти, як автоматична генерація LOD (рівень деталізації); покриття і руйнування, які усувають приховані поверхні і непотрібні деталі; і інструмент Proxy Geometry, який об'єднує кілька сіток і їх матеріали в одне ціле.
- Mesheditingtools. UnrealEngine включає базові інструменти редагування сітки для виправлення невеликих проблем в геометрії без необхідності виправлення їх в пакеті з вихідним кодом і повторного імпорту. У редакторі StaticMeshEditor можна обирати сітку різними способами – шляхом вибору, за матеріалом, елементу або зі збільшенням чи зменшенням і створювати, видаляти або перевертати обрані грані або відокремлювати їх.
- Landscapeandterraintools – набір інструментів для створення масштабних просторів і ландшафтів відкритого світу з горами, долинами і навіть печерами за допомогою системи Landscape [5].

2. UnrealEnginedозволяє працювати із анімацією і не обов'язково для комп'ютерних ігор.

- Characteranimationtools – набір інструментів для роботи із персонажами та їх анімаціями.

- AnimationBlueprints – створення та управління складною поведінкою анімації за допомогою анімаційних креслень. AnimationBlueprint – це спеціалізований Blueprint, який керує анімацією скелетної сітки. В редакторі креслень анімації, можна виконувати змішування анімації, безпосередньо управляти кістками скелета або задавати логіку, яка в кінцевому підсумку буде визначати остаточну позу анімації для скелетної сітки, яку слід використовувати для кожного кадру.
- TakeRecorder дозволяє записувати анімації з захоплення руху, пов'язані з персонажами в сцені. Записуючи акторів в підпослідовності і впорядковуючи їх по метаданим, можна легко управляти складними постановками.
- Sequencer: state-of-the-artcinematics – нелінійний інструмент для кінематографічного редагування і анімації в реальному часі. Зміна освітлення, блокування камери, персонажів і налаштування декорації для кожного кадру[5].

3. Графічний модуль дозволить працювати із світлом, матеріалом та знайти потрібні налаштування для рендеру.

- Flexiblematerialeditor – редактор для створення матеріалів та поверхонь об'єктів.
- Photorealrasterizingandraytracinginrealtime – високоякісні візуальні ефекти голлівудського якості за допомогою растеризатора і трасувальника променів UnrealEngine: трасування відображень, тіней, прозорості, навколишнього оклюзії, освітлення на основі зображень і загального освітлення, Ефекти включають в себе динамічні м'які тіні від локальних джерел світла з трасуванням променів.
- Sophisticatedlighting – реалістичні ефекти внутрішнього і зовнішнього освітлення: атмосферний ефект сонця і неба, об'ємний туман, об'ємні світлові карти, попередньо розраховані сценарії освітлення.

- VirtualTexturing – UnrealEngine пропонує два методи для підтримки дуже великих текстур, розділяючи їх на маленькі плитки і завантажуючи тільки видимі.
- Post-processandscreen-spaceeffects - ряд ефектів постобробки дозволяють покращити загальний вигляд сцени.
- Advancedshadingmodels – вдосконалені моделі затінення UnrealEnginedозволяю отримувати реалістичні результати об’ємності на широкому спектрі об’єктів і поверхонь [5].

4. UnrealEnginedозволяє створювати реалістичні ефекти та симуляції явищ.

- Niagaraparticlesandvisualeffects дозволяють створити якісні ефекти вогню, диму, пилу і води за допомогою системи частинок у вбудованому редакторі візуальних ефектів Niagara.
- Clothingtools – інструмент для моделювання одягу та тканевого матеріалу. Параметри одягу можна задавати безпосередньо в UnrealEditor.
- Chaos physics and destruction system. Chaos – це високопродуктивна фізична система від UnrealEngine. Використовуючи функцію «Руйнування хаосу», користувачі можуть створювати великомасштабні сцени кінематографічної якості, направлені на процес руйнації. Chaos підтримує роботу із одягом, волоссям і вторинними ефектами як пил і дим [5].

5. Створювати ігрові механіки і налаштовувати штучний інтелект дозволяє окремий модуль.

- Advancedartificialintelligence (AI) – модуль дозволяє навчити неігрових персонажів рухатися локаціями та здійснювати продумані рухи за допомогою креслень або дерев поведінки. Динамічна навігаційна сітка оновлюється в режимі реального часу при переміщенні об’єктів для оптимального проходження маршруту в будь-який час.

- UnrealMotionGraphicsUIDesigner (UMG) – модуль для створення елементів користувацького інтерфейсу. За допомогою віджетів Blueprint можна створювати меню, індикатори та інші візуальні показники.
- VariantManager дозволяє створювати і редагувати варіанти свого активу, які включають параметри видимості, перетворення і призначення матеріалів, а також активувати або деактивувати їх в Unreal. Це ідеально підходить для оглядів дизайну і макетів локацій.
- Blueprintvisualscriptingsystem – за допомогою візуальних сценаріїв Blueprintможна швидко створювати прототипи і керувати інтерактивним контентом, не використовуючи для цього програмний код. Blueprintдозволяє будувати поведінку і взаємодію об’єктів, зміни в користувацькому інтерфейсі, налаштувати управління персонажем [5].

6. UnrealEngine має базу моделей та сервісів, які допоможуть в створенні гри.

- QuixelMegascans – онлайн сервіс, що містить бібліотеку якісних текстур та моделей.
- Marketplaceecosystem. UnrealEngine має тисячі високоякісних ресурсів і плагінів для прискорення розробки гри. Користувач може отримати доступ до локацій, персонажів, анімації, текстур, звукових і візуальних ефектів, музичних треків, макетів, плагінів, додаткових інструментів і створених проектів початкового рівня [5].

3. Unity3D-міжплатформове середовище розробки комп’ютерних ігор, розроблена американською компанією Unity Technologies.

Модель розповсюдження. Unityмає декілька варіацій, в залежності від обраної змінюється вартість. Якщо дохід компанії не перевищує ста тисяч доларів, користувачі не зобов’язані здійснювати виплати або внески, там можуть користуватися безкоштовною ліцензованою версією, яка майже не відрізняється від інших ліцензій [6].

Мови програмування. Unityдозволяє створювати скрипти на C# та JavaScript[6].

Функціонал. Unity3Dповноцінне ПЗ для створення комп’ютерних ігор.

1. Робота із анімаціями та відео.

- Пакет анімації – набір інструментів, що включає повторювані анімації, повний контроль ваг анімації під час виконання, виклик подій у межах відтворення анімації, витончена державна машинаієрархії та переходи, поєднання форм для анімації обличчя та багато іншого.
- 2D InverseKinematics (IK) – пакет дозволяє налаштувати анімаційний скелет персонажів. 2D IK автоматично обчислює положення та обертання ланцюга кісток, що рухаються навколо указаної точки. Це полегшує анімування кінцівок персонажів для анімації або маніпулювання скелетом у режимі реального часу, оскільки не потрібно вручну встановлювати положення кісток.
- Timeline – інструмент для створення кінематографічного контенту, ігрових сценаріїв, аудіо послідовностей та складних ефектів.
- Cinemachine – пакет для легкого налаштування камери в грі. Інструмент уже має набір логіки для камер, тому можна з легкістю створити камеру з необхідним видом, поєднувати із запланованими сценами та різними ефектами.
- VideoPlayer – компонент, який дозволяє прикріпляти відеофайли до ігрових об’єктів, таким чином створюючи реалістичну анімовану поверхню. Такий підхід дозволяє створити правдоподібний вигляд в сцені і при цьому система витрачає менше ресурсів на рендер локації[6].

2. Робота із аудіо.

- Audiocomponents / mixer – набір інструментів для створення об’ємного просторового звучання, поєднання різних звукових доріжок в один файл, додання звукових ефектів та багато іншого. Рушій підтримує основі аудіоформати, що дозволяє без проблем імпортувати готові звукові файли.
- Spatializer – додатковий плагін, що дозволяє змінювати спосіб передачі аудіоджерела в навколишній простір. Вбудоване панорамування джерел звуку дозволяє регулюватигучність звукудля лівого та правого вуха

на основі відстані та кута між об’єктом, до якого іде звук (персонаж гравця) та джерелом звуку. Це дозволяє гравцеві орієнтуватися в просторі та розуміти, де саме відбувається певна дія [6].

3. Робота із графічними ефектами.

- ParticleSystem– інструмент для створення додаткових об’єктів в сцені, таких як рідини, хмари, полум’я, пил або туман.
- VisualEffectGraph – інструмент для управління візуальними ефектами. Дозволяє створювати декілька ParticleSystem, які взаємодію між собою, додавати статичні сітки та контролювати властивості шейдерів, створювати сценарії для активації та деактивації ефектів, перегляд змін під час редагування ефектів для швидкої оптимізації та створення необхідного візуального відображення [6].

4. Робота із інтерфейсом

- UIElements– набір інструментів для створення користувацького інтерфейсу. Створений на принципах роботи веб-технологій, підтримує таблиці стилів, динамічну і контекстну обробку подій та збереження ведених даних.
- UnityUI– пакет, що являє собою інструментарій для створення користувацьких інтерфейсів для ігор та додатків. Усі елементи представлені у вигляді ігрових об’єктів, що спрощує роботу.
- TextMeshPro– плагін, що дозволяє працювати із текстом. Великий вибір шрифтів дозволяє створити необхідний стиль для гри. Окрім цього існує можливість детальної роботи із кольорами та ефектами[6].

5. Робота зі світлом.

- GlobalIllumination – група методів, що моделюють як пряме, так і непряме освітлення для забезпечення реалістичних результатів освітлення. Unity має дві системи освітлення, що поєднують в собі пряме та непряме освітлення.Система «BakedGlobalIllumination» включає світлові карти, світлові сфери тасфери віддзеркалення. Усі шляхи візуалізації підтримують систему глобального освітлення Baked. Система «RealtimeGlobalIllumination»

включає глобальне освітлення в реальному часі за допомогою програми Enlighten та додає додаткові функціональні можливості для світлових сфер.

- ProgressiveLightmapper – це швидка система відстеження на основі трасування, яка забезпечує запечені світлові карти і світлові ефекти з прогресивними оновленнями.
- RayTracing – це функція, яка дозволяє отримувати доступ до даних, які відсутні на екрані. Даний метод використовується для зчитування взаємодії між декількома об’єктами. Окрім цього існує можливість створювати реалістичний ефект додаткового світла.
- ReflectionProbes – даний інструмент схожий на камеру, що фіксує сферичний вигляд його оточення у всіх напрямках. Захоплене зображення зберігається і може використовуватися предметами зі світловідбиваючими матеріалами. Створюється ефект дзеркально чистих поверхонь об’єктів, у яких можна побачити відображення[6].

6. Робота в самому редакторі

- CustomExtensions – метод, який дозволяє збільшити зручність роботи в редакторі. За допомогою нього можна взаємодіяти із змінними, заданими в скриптах, що дозволяє не редагувати сам скрипт, а змінювати назначені об’єкти та числові значення властивостей прямо в редакторі.
- PackageManager– менеджер, який містить найкорисніші плагіни для Unity. Створюючи новий проект, редактор не додає весь доступний функціонал, щоб зменшити навантаження на систему та не засмічувати гру модулями, які не будуть використовуватись. Якщо розробнику потрібний певний плагін, він може легко його встановити та користуватись. PackageManagerмістить список попередніх версій кожного плагіну, на випадок, якщо нова версія виявилась не стабільною або має помилки у роботі.
- UnityHub – сторонній додаток, що дозволяє управляти усіма існуючими проектами та версіями ігрового рушія. Програма має окремий

розділ, де зібрані навчальні матеріали та готові проекти початкового рівня, що користувачі могли вивчити основи та почати розробку власного проекту.

- Prefab – редактор дозволяє поєднувати ігрові об’єкти (камеру, звук, модель, скрипт тощо) в шаблон, який можна використовувати для створення нових об’єктів, що будуть мати усі необхідні складові різних напрямів роботи. Окрім цього префаби можна переносити в інший проект, що дозволяє підвищити рівень продуктивності роботи, не витрачаючи час на повторне створення ігрових об’єктів.
- Physics – Unity дозволяє працювати із фізикою прямо в редакторі. Фізичний рушій здатен прорахувати взаємодію між об’єктами: перевірити ймовірність зіткнень тіл, наявність гравітації, створити додатковий рух об’єктів та багато інший фізичних явищ [6].

7. Інструменти для створення ігрових світів.

- Terrain – інструмент для створення ландшафтів та великих поверхонь. Можна легко створювати гори, острова чи поля. Окрім цього можна наносити текстури і додавати рослинність (траву, квіти та дерева).Орієнтовано переважно на роботу в 3D.
- Tilemap – метод створення рівнів гри. Простір поділений на рівні плитки, які заповнюються відповідним зображенням, що відображає землю, виступ на певній висоті або небезпечну територію. Застосовується в розробці 2D ігор[6].

8. Рендер та робота із візуальною складовою.

- ScriptableRenderPipeline (SRP) – функція, що дозволяє управляти процесом рендеру через скрипти, написані на C#.
- HighDefinitionRenderPipeline (HDRP) – це високоточний сценарій рендерингу, створений компанією. HDRP використовує методи освітлення, засновані на фізичній основі, лінійному освітленні, HDR освітленні. Дозволяє створювати графічні сцени високої якості.
- UniversalRenderPipeline – гнучкий сценарій рендерингу, який дозволяє покращити графічне відображення сцени. Дозволяє обрати значення

впливу на сцену, що дає можливість використовувати його навіть в невеликих проектах із невеликими системними вимогами. За необхідності в параметри рендеру можна внести зміни використовуючи мову програмування C#.

- Post-processing – набір графічних методів, що застосовуються як повноекранні фільтри та ефекти камери. Дозволяє покращити візуальну складову гри.
- ShaderGraph – інструмент для роботи із шейдерами. Дозволяє створювати додаткові візуальні ефекти для персонажів та декорацій, спрямованих на посилення реалізму гри[6].

9. Інструменти для оптимізації гри.

- Profiler– інструмент, за допомогою якого можна отримати інформацію про продуктивність гри. Дозволяє переглядати розподіл ресурсів під час роботи додатку. Profiler збирає та відображає дані про продуктивність гри в таких областях, як процесор, оперативна пам’ять, графічне навантаження та аудіо. Результати відображаються у вигляді діаграм, що дозволяє швидко виявити проблемні місця.
- FrameDebugger – дозволяє зупинити гру на певному кадрі та переглядати функції, які використовуються для візуалізації цього кадру. Окрім переліку недоліків, debugger також дозволяє переходити від одної функції до іншої, щоб детально переглянути, як сцена будується з графічних елементів [6].

Проаналізувавши ігрові рушії можна зробити наступний висновок: усі рушії представляються собою повноцінне ПЗ для створення комп’ютерних ігор. Кожен ПП дозволяє працювати із аудіо, створювати ландшафти, додавати візуальні ефекти. За моделлю розповсюдження усі рушії мають безкоштовну ліцензію. Але існує ряд критеріїв, за якими був обраний саме Unity3D:

- Мова програмування – рушій дозволяє створювати скрипти на мові програмування C#, що є найкращим варіантом.

- Зрозумілий інтерфейс – програма побудована так, що користувач може з легкістю знайти всі необхідні компоненти та функції1. Окрім цього, можна переміщувати вікна, що дозволяє створити необхідний робочий простір.
- Велика кількість навчального матеріалу – Unityдобре підходить для початкової роботи та знайомства із основами розробки ігор. Користувач поступово знайомиться із компонентами та можливостями рушія, не витрачаючи час на вивчення складних проектів.
- Плагіни – додаткові інструменти, що пришвидшуються та покращують результат виконаної роботи. Розширення збільшують початкові можливості редактору, що дозволяє створювати ігрові рівні або додавати графічні ефекти без особливих труднощів.
- AssetStore – власний електронний ресурс розробників Unity, де можна знайти необхідні ігрові об’єкти. Чудово підходить для початківців, які знайомляться із створення комп’ютерних ігор.

РОЗДІЛ 2
ПРОЕКТНІ РІШЕННЯ

2.1 Короткий опис сюжету

Сюжет – один із ключових елементів пригодницької гри. Гравець долучається до історії, подорожує світом, знайомиться із дійовими особами та впливає на подальший розвиток подій.

Необхідно сформувати зав’язку сюжету, ознайомити гравця з місцем початку сюжету. Розподілимо опис на декілька складових частин.

Час. Події гри відбувають в майбутньому. Люди проживають за межами планети Земля. Окрім людей існує безліч інших рас, які мирно співіснують між собою, в тому числи і з людством. Рівень технологій дозволяє легко здійснювати космічні перельоти із одної точки космічного простору в інший. Людство активно прогресує: досліджує космічні тіла, відкриває нові планети, вдосконалює існуючі технології.

Персонажі. В грі будуть присутні декілька дійових осіб. Головна героїня із позивним «Ангел 6», космічна база «Альфа 3», дружні персонажі: дослідники Марсі та Флік. В ролі ворогів будуть виступати істоти дослідницької планети.

Попередні події. Успішно відремонтувавши космічний корабель «Ангел 6» продовжує свою подорож до космічного бази, але під час польоту отримує повідомлення з проханням допомогти на одній із рядом розташованих планет. «Ангел 6» вирішує прийняти сигнал та відправляється в розвідку.

Початок гри. Космічний корабель вдало приземляється на планеті, «Ангел 6» вивчає місцевість та знаходить Марсі – саме вона відправила запит про допомогу.

Основа частина гри. Марсі знайомить головну героїню із ситуацією – Марсі разом зі ще одним дослідником прибули на цю планету в пошуках

дорогоцінного ресурсу. Під час розвантаження корабля, дослідники почули дивний шум і Флік вирішив дізнатися що це, але так і не повернувся. Марсі вирішила не ризикувати і відправила запит про допомогу. «Ангел 6» вирішує допомогти, знаходить місцеположення дорогоцінного ресурсу, окрім цього її доводиться вступити в двобій із істотами, яких вона уже зустрічала раніше. Героїня знаходить ворота, які ведуть в глиб острова і можливо відкриють нову інформацію про те, що тут відбувається.

Завершення. Отримавши ключ від воріт, «Ангел 6» проникає в стародавнє місто в пошуках Фліка.

2.2 Проектування рівнів гри

Гра складається із одної локації та трьох рівнів. Кожен рівень знаходиться на різній висоті, що створює ефект багатоповерхового будинку.Сама локація обмежена ігровими об’єктами, тому гравець не може вийти за межі створеного простору. Ігровими об’єктами являються стіни, скелі, морська глибина та інші тверді тіла. Коженрівень відрізняється вмістом наявних предметів та візуальним оточенням. Гравець повинен дослідити кожну із них. На локаціях будуть місця, в які не можливо потрапити одразу, необхідно знайти певний предмет або виконати дії. Також на локаціях будуть присутні вороги.

Перший рівень. Початок гри, події відбуваються на відкритій місцевості. Гравець вивчає локацію та знайомиться з грою. Необхідно створити місцевість, яка дозволить гравцю вивчити управління персонажем, познайомить з механіками гри та дозволить насолодитися візуальною складовою гри. Під час пересування по рівню «Ангел 6» буде розповідати свої думки стосовно ситуації, це дозволить гравцю отримати додаткову інформацію про персонажа.

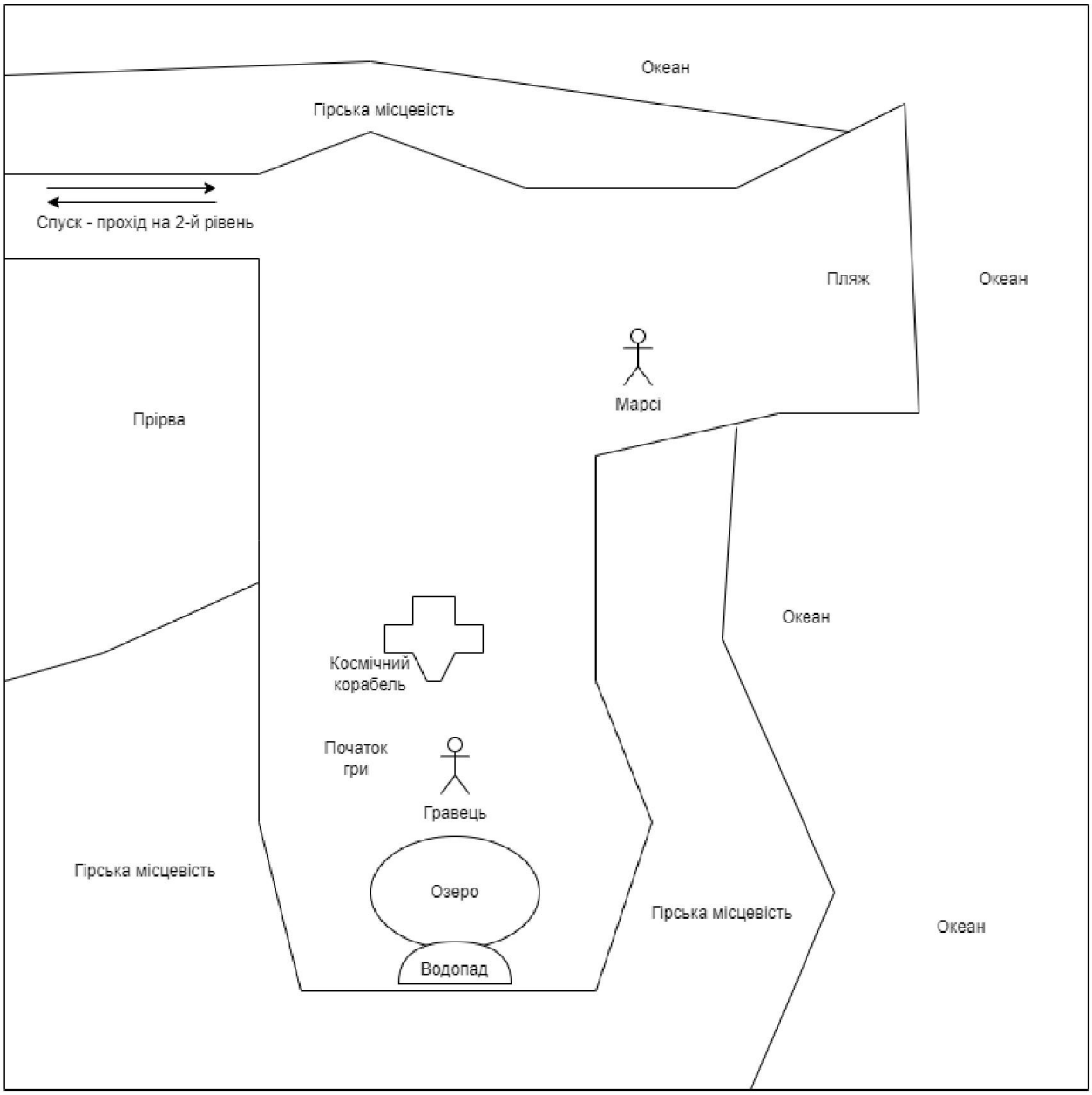

Рисунок 2.1 – Схема першого рівня гри

Другий рівень. Локація, до якої гравець потрапляє пройшовши по спуску старої шахти. Тут відбувається перша зустріч із ворожими персонажами, проходить ознайомлення гравця з бойовою системою.Паралельно проходить подальша розповідь сюжету. В результаті досліджень гравець знаходить вхід до печери, що і являє собою третій рівень локації.

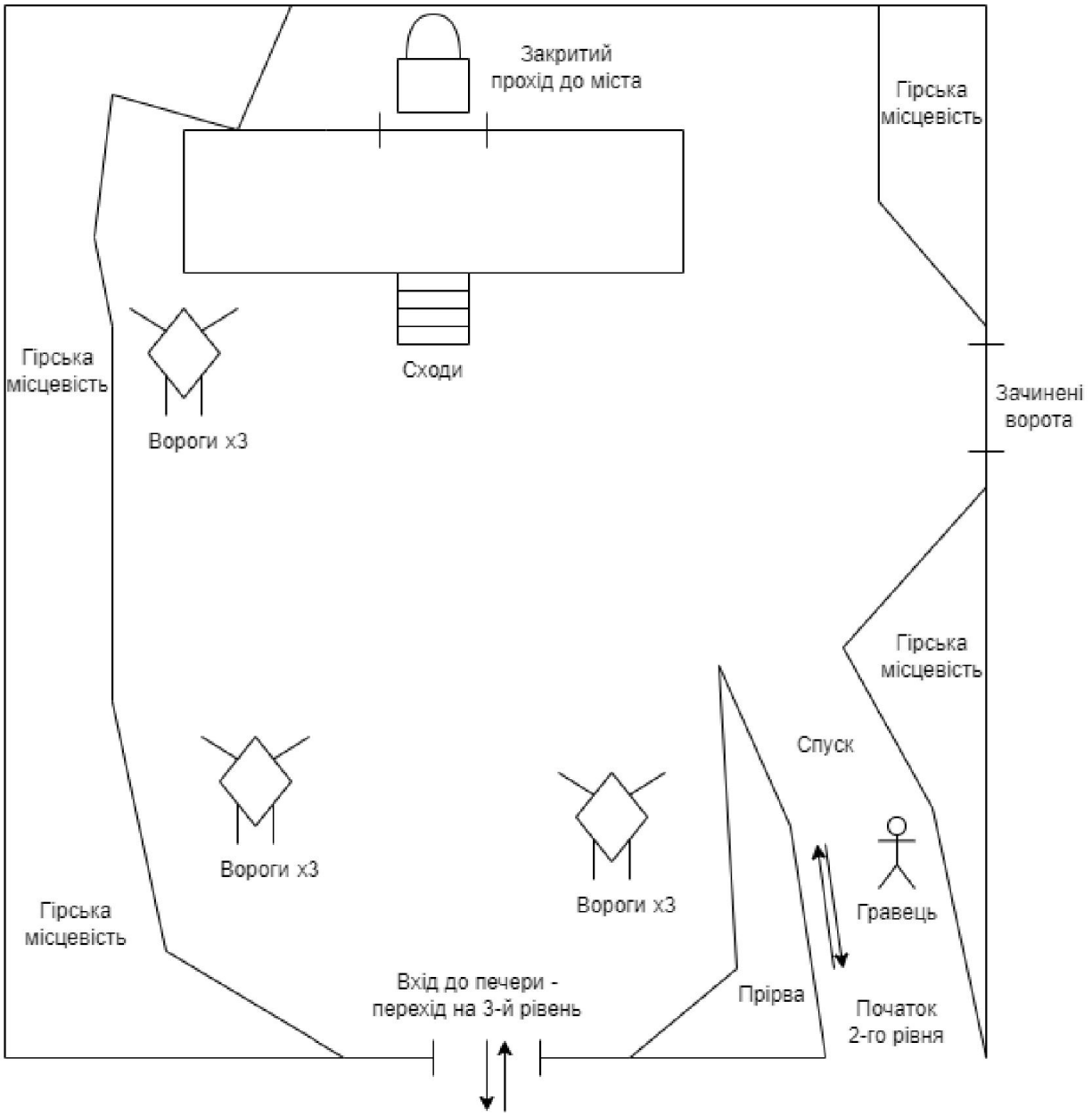

Рисунок 2.2 – Схема другого рівня гри

Третій рівень.Спочатку рівень являє собою звичайну печеру, але по мірі просування гравця стає зрозуміло що це скарбниця, де можливо вдасться знайти відповіді на запитання та необхідні матеріали. Спочатку гравцю необхідно пройти ряд перешкод – охоронну систему від небажаних гостей. Після успішного проходження гравець потрапляє до головної зали, в якій знаходяться ключ від воріт міста та охоронці ключа. Доки гравець не забрав предмет, він може спокійно пересуватися по кімнаті та вивчати її. Якщо забрати ключ, це активує охоронців і доведеться вступати в бій. Окрім ключа

в самій печері знаходяться залежні ресурсів, які шукали дослідники – це дозволяє гравцю повернутися до Марсі та повідомити новину.

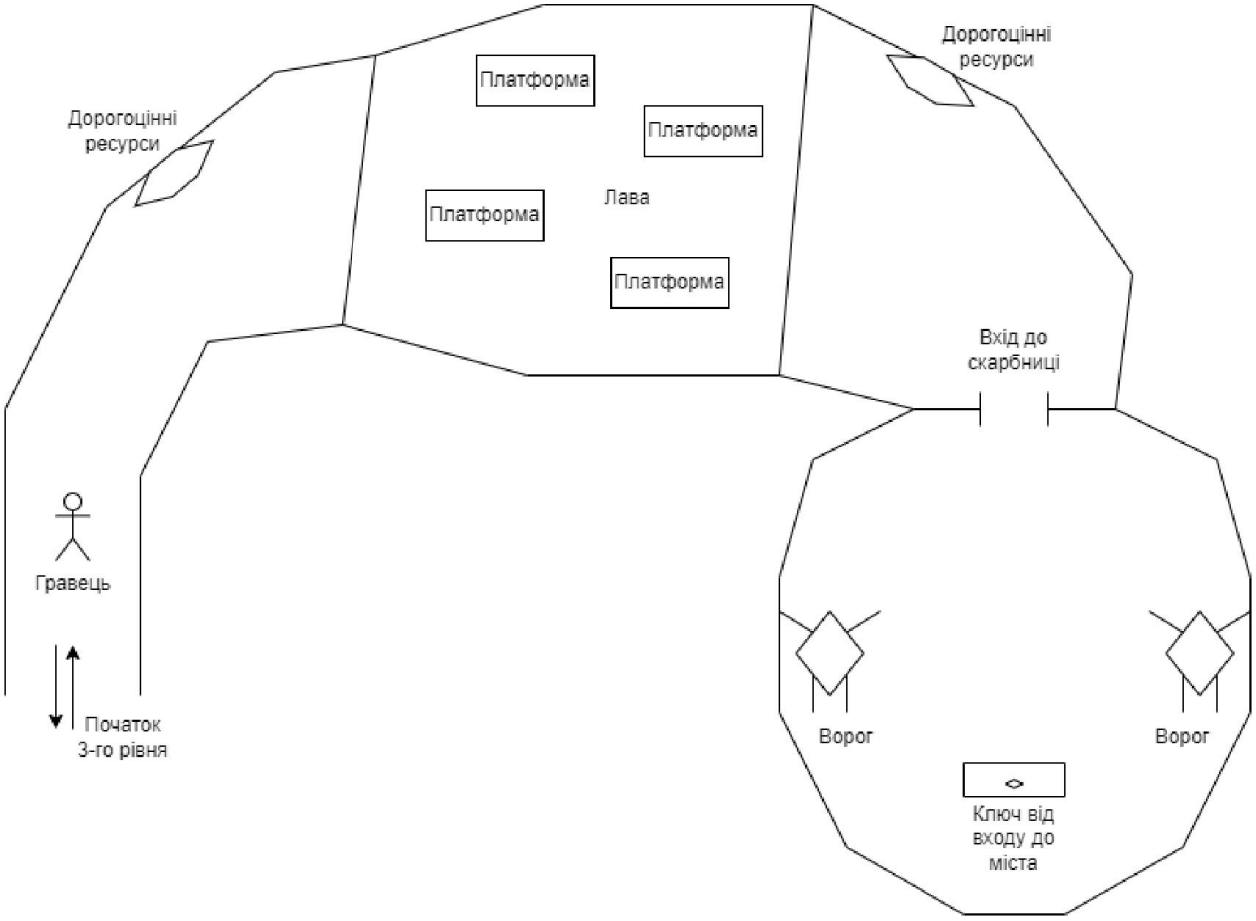

Рисунок 2.3 – Схема третього рівня

В подальшому гравець може вільно пересуватися між рівнями, виконуючи ряд завдань. В кінцевому результаті гравець отримує ключ і завершає гру біля проходу до міста на другому рівні локації.

2.3 Проектування основних ігрових механік

Гра повинна містити наступні ігрові механіки:

Положення камери:

- вид камери в режимі «від третьої особи»;
- камера повинна бути розміщена позаду персонажа на середній дистанції, це дозволить бачити самого персонажа та мати достатню видимість по боках;

- додати можливість рухати камеру за допомогою миші для зручного вивчення оточуючих предметів та локації;

Управління персонажем:

- додати можливість рухатися в усіх напрямках за допомогою відповідних кнопок на клавіатурі;
- додати можливість здійснювати стрибки за допомогою відповідної кнопки на клавіатурі;
- встановити фіксовану швидкість пересування персонажа;

Неігрові персонажі:

- додати персонажів, управління якими виконує комп’ютер;
- додати персонажів, мирно налаштованих по відношенню до гравця;
- додати можливість ведення діалогів із дружелюбними персонажами;
- додати можливість отримання завдань від дружелюбних персонажів;
- додати персонажів, нейтрально налаштованих по відношенню до гравця;
- створити правила, які змушують змінити відношення до гравця із нейтрального на агресивне;
- додати персонажів, агресивно налаштованих по відношенню до гравця;
- створити стани та можливість переходу між ними для зміни поведінки персонажів в залежності від навколишнього середовища;

Бойова система:

- створити запас здоров’я для персонажа гравця та ворожих персонажів;
- налаштувати систему зарахування пошкоджень і зменшення запасу здоров’я

- додати тимчасовий захист для гравця та ворожих персонажів після отримання пошкоджень.

Звуки:

- додати звуки для персонажа гравця;
- додати звуки для ворожих персонажів;
- додати звуки бою;
- додати звуки взаємодії із активними об’єктами;
- додати музику на задньому фоні.

Інтерфейс:

- створити ігрове меню для початку гри;
- створити меню паузки для завершення гри;
- додати шкалу здоров’я для персонажа гравця;
- створити діалоговевікно;
- створитивікно підказок;
- додати відображення завдання після оновлення поточного.

Система подій:

- створити тригер для оновлення точки відновлення;
- створити тригер для відображення діалогового вікна;
- створити тригер для відображення вікна підказок;
- створити тригер для генерації нових ворогів;
- створити тригер для оновлення поточного завдання;
- створити тригер для взаємодії з активними об’єктами;
- систему подій (поява діалогів, оновлення поточного завдання).

2.4 Вибір додаткових інструментів для Unity3D

ProBuilder –плагін дозволяє створювати ігрові об’єкти та цілі рівні. ProBuilder є гібридним інструментом 3D-моделювання та дизайну рівнів, оптимізованим для створення простих геометричних об’єктів, але здатним і на детальне редагування і UV-розгортання. ProBuilder використовують для

швидкого створення прототипів структур, елементів складних ландшафтів, транспортних засобів і зброї, а також для додавання довільної геометрії зіткнень, тригерних зон і навігаційних сіток.До переваг плагіну відносять можливість легкого обміну файлами Unity з іншими інструментами створення цифрового контенту, що дозволяє у разі необхідності вдосконалити створені моделі. До основних можливостей належать: текстурування, додання кольорів, створення складних фігур, створення параметричних об’єктів.

ProGrids – плагін, який часто використовують у поєднанні із попереднім. Візуальна і функціональна сітка з прив’язкою до всіх трьох осей. ProGrids прискорює роботу і підвищує її якість, дозволяючи розробляти рівні з легкістю, швидкістю і точністю. Інструмент особливо корисний в розробці модульних або «плиткових» оточень. Фіксація сітки забезпечує рівномірне розташування об’єктів.Світова сітка ніколи не змінює положення чи орієнтацію. Завдяки цьому можна розуміти, де знаходяться предмети, наскільки вони віддалені один від одного та як далеко вони переміщуються.ProGrids допомагає створювати рівні із ProBuilder, створюючи ідеальну геометрію. Можна легко створювати фігури та об’єкти.ProGrids має візуальні підказки, які допомагають чітко розуміти, із яким саме об’єктом працює користувач.

Cinemachine – набір інструментів для динамічних та інтелектуальних камер без написання коду, які дозволяються створювати кращі кадри на основі композиції сцени та налаштовувати поведінку камери в режимі реального часу для 2D та 3D проектів. Плагін являється вдосконаленою версією стандартної камери редактора Unity, що спрощує налаштування логіки камери та прискорює процес розробки через відсутність написання скриптів для необхідної поведінки камери, оскільки більша частина уже реалізована в самому плагіні. Набір моделей поведінки дозволяє використовувати Cinemachine в іграх різних жанрів, що робить плагін універсальним рішенням. Додавши в сцену ігровий об’єкт Cinemachine,

розробник отримує доступ до великої кількості параметрів з допомогою яких можна задати обмеження руху камери, налаштувати логіку взаємодії з іншими об’єктами в сцені, задати пріоритет роботи камер, якщо їх декілька, додати ефекти розмитості, мерехтіння.

UnityAssetStore–це велика бібліотека асетів. Unity Technologies і члени спільноти створюють асети і публікують в цьому магазині. Типи асетів варіюються від текстур, анімацій і моделей до готових проектів, навчальних матеріалів і плагінів редактора. Багато асетів надаються безкоштовно, інші доступні за помірними цінами. Всі асети можна завантажувати безпосередньо в свій проект Unity. Магазин Unity Asset Store можна відвідувати двома способами: на веб-сайті або за допомогою движка Unity.

Blender – програма для роботи з комп’ютерною графікою яку застосовують для створення анімаційних фільмів, візуальних ефектів, 3D-печатних моделей, анімованої графіки, інтерактивних 3D-додатків, віртуальної реальності та комп’ютерних ігор. Додаток має обширні можливості, до основних можна віднести: 3D моделювання, накладання текстур на моделі, створення текстур, редагування растрової графіки, моделювання рідин і диму, моделювання частинок, рендеринг, графіка руху, редагування відео і створення анімацій. Процес створення ігрових об’єктів не викликає труднощів, оскільки графічний редактор має великий набір інструментів. Додаток користується попитом, а як наслідок існує велика кількість навчального матеріалу та документації, що дозволить за короткий проміжок часу вивчити базові складові та приступити до створення. Після створення моделі, Blender дозволяє експортувати її в один із форматів, які підтримує Unity, тому процес перенесення моделі із одного редактора до іншого проходить без проблем.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ ПРОЕКТУ

3.1 Створення ігрового світу

Поверхня землі була створена за допомогою 3Dоб’єкту «Поверхня» та матеріалів трави, піску, каменю.

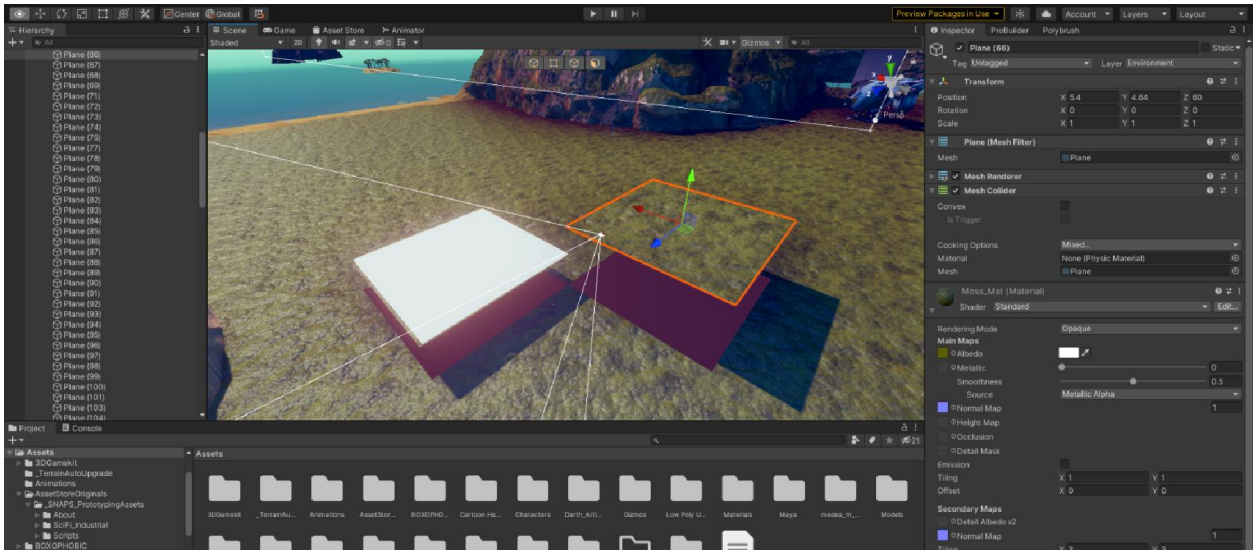

Рисунок 3.1 – Ігровий об’єкт «Поверхня» із стандартним матеріалом та матеріалом трави

Для зручного редагування площі в подальшому фігури були об’єднані в один об’єкт за допомогою плагіна ProBuilder.

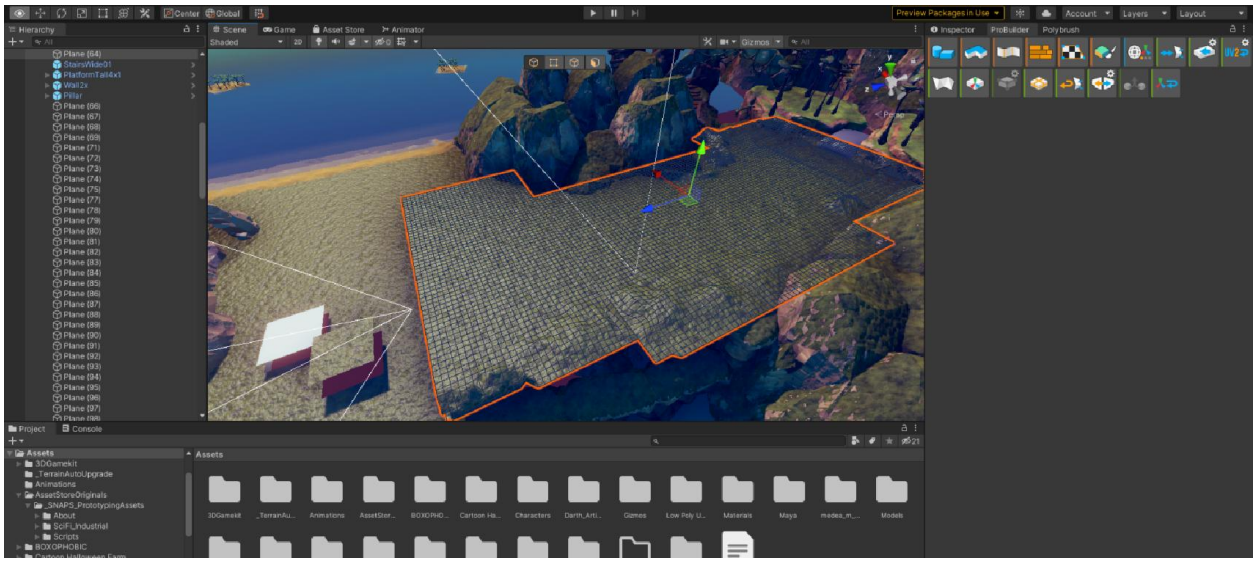

Рисунок 3.2 – Об’єкти «Поверхня» об’єднані в один елемент

В якості «стін» ігрового світу були використані моделі скель.

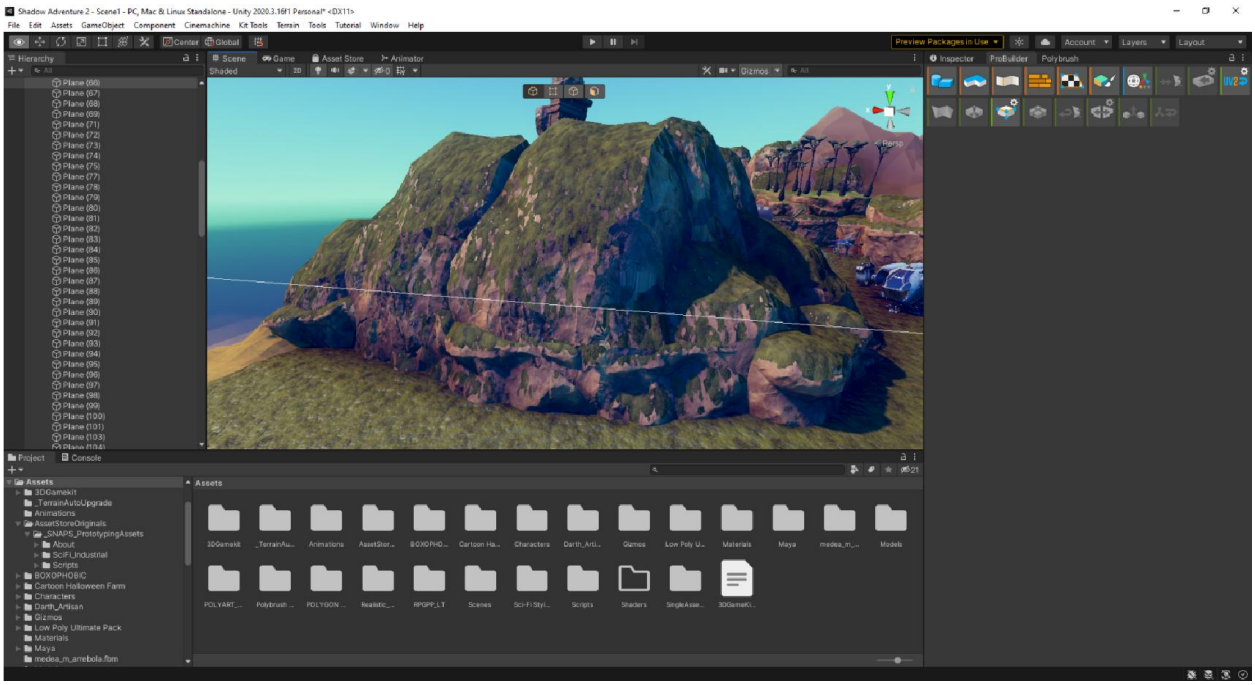

Рисунок 3.3 – Моделі скель обмежують вихід гравця за межі ігрового світу

Для обмеження руху персонажа у воді були додані об’єкти «Куб». В параметрах об’єктів були відключені їх графічні відображення в сцені, але фізична взаємодія все ще працює, тому зіткнувшись з ними, гравець не зможе пройти далі.

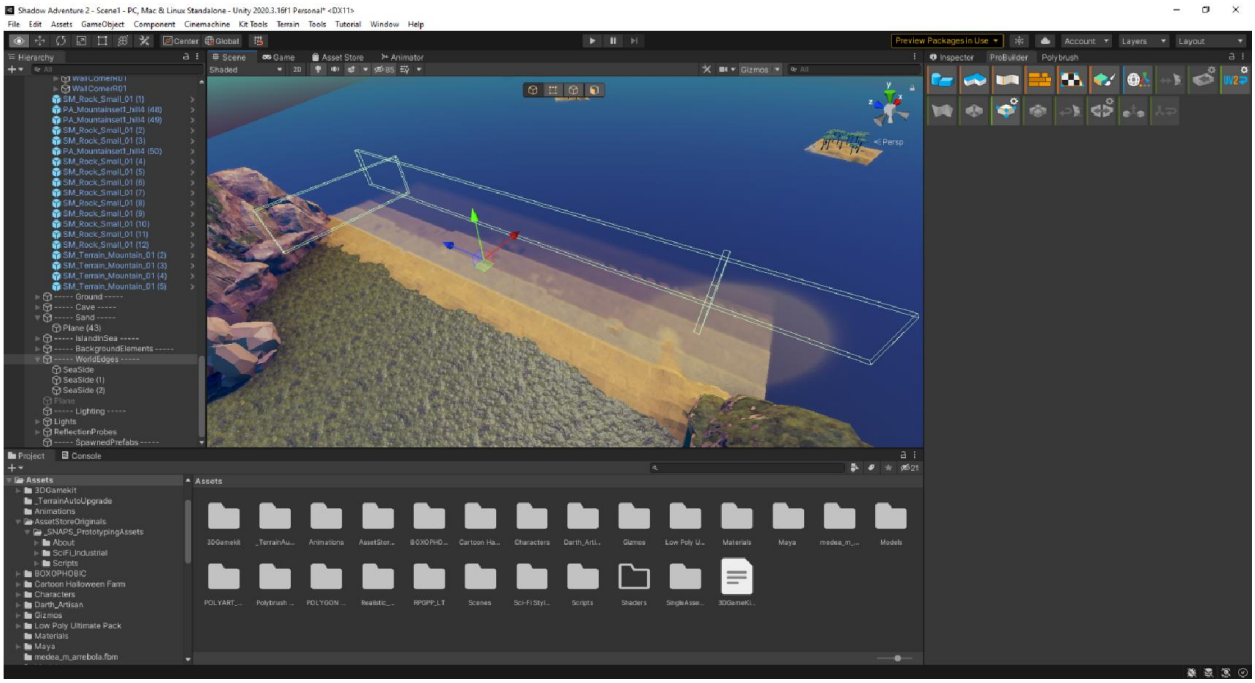

Рисунок 3.4 – Обмеження території рівня за допомогою фізичного колайдеру

Для наповнення ігрового світу були додані елементи декору: трава, каміння, дерева та інші об’єкти.

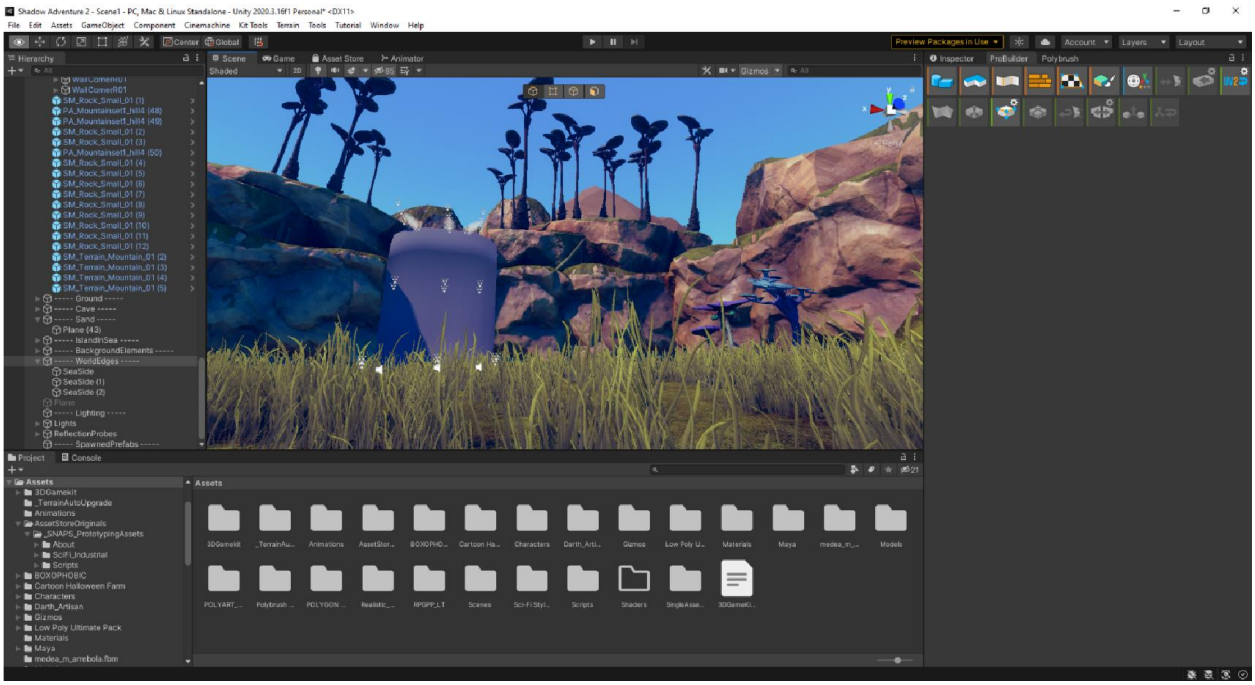

Рисунок 3.5 – Моделі трави та дерев дозволяють усунути порожнечу локації

Деякі прості моделі дерев та елементи ландшафту були створені власноруч в програмі Blender.

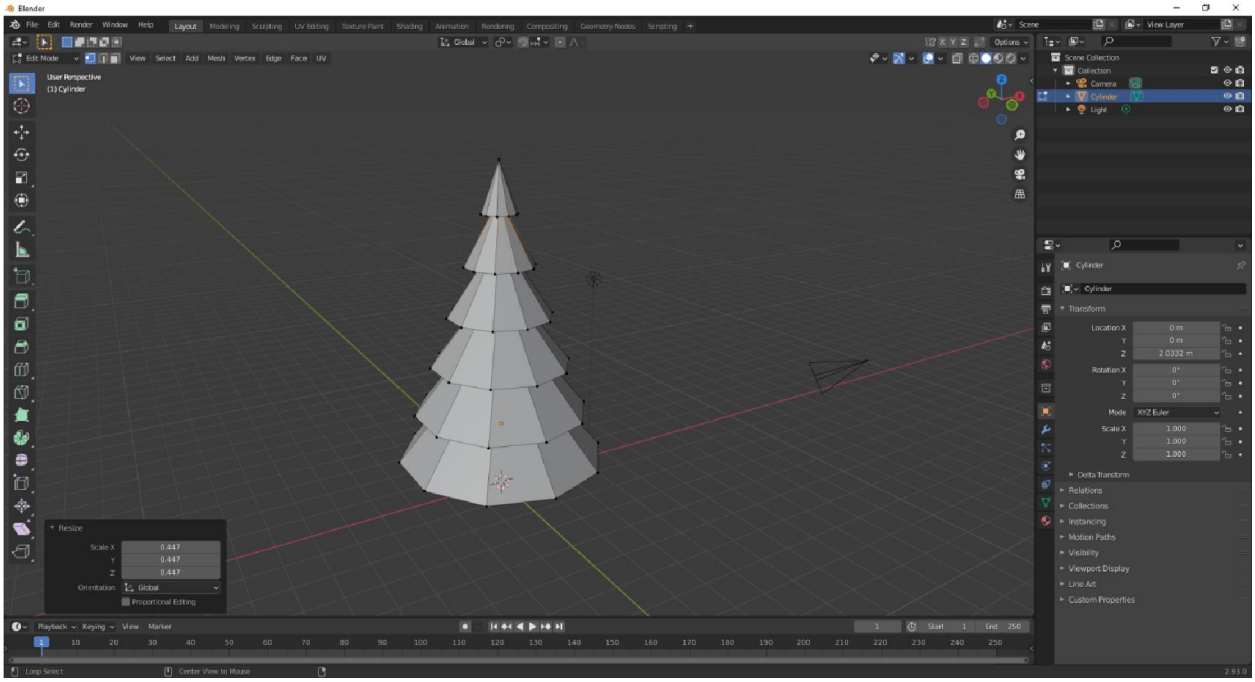

Рисунок 3.6 – Процес створення простої моделі дерева в Blender

Створено об’єкти дальнього плану для віддалення горизонту та створення ефекту масштабності.

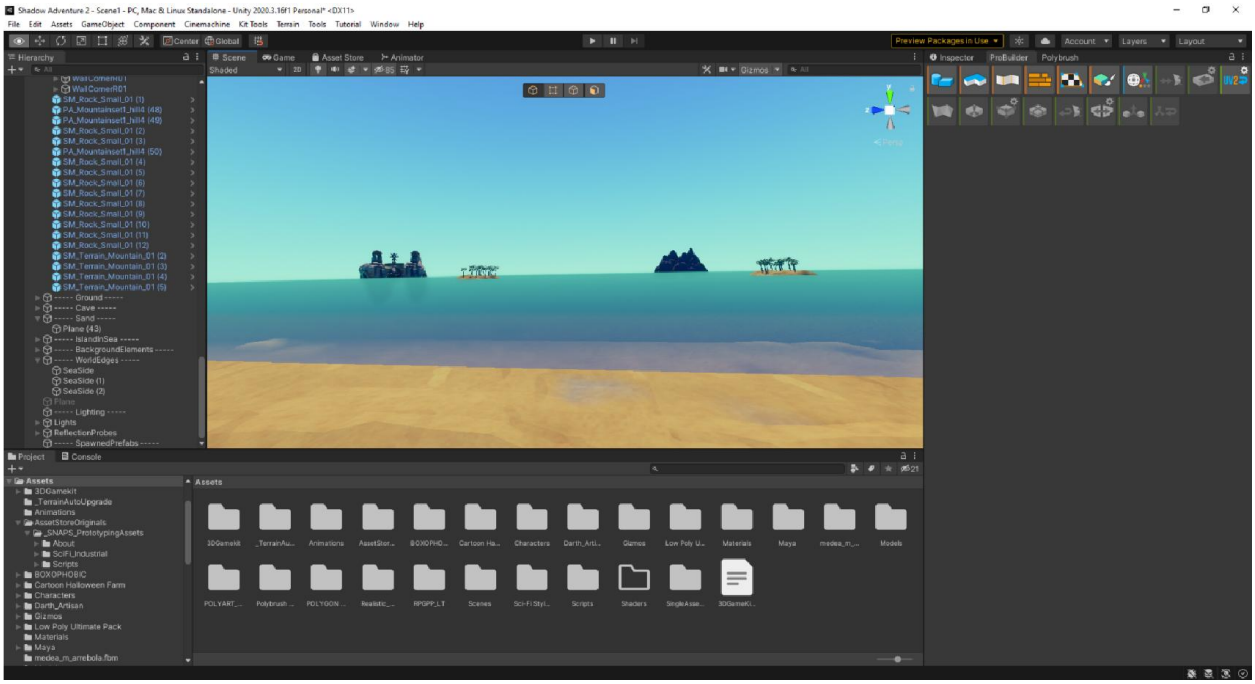

Рисунок 3.7 – Об’єкти дальнього плану в океані

В сцену додано джерело світла, яке симулює справжнє сонце. В залежності від положення об’єкта в просторі можна відтворювати різний час доби. Якщо сонце знаходиться над поверхнею, інші об’єкти в локації матимуть тінь а на воді буде відображення моделі сонця. Якщо джерело світла перемістити за межі локації, з’явиться характерна темнота.

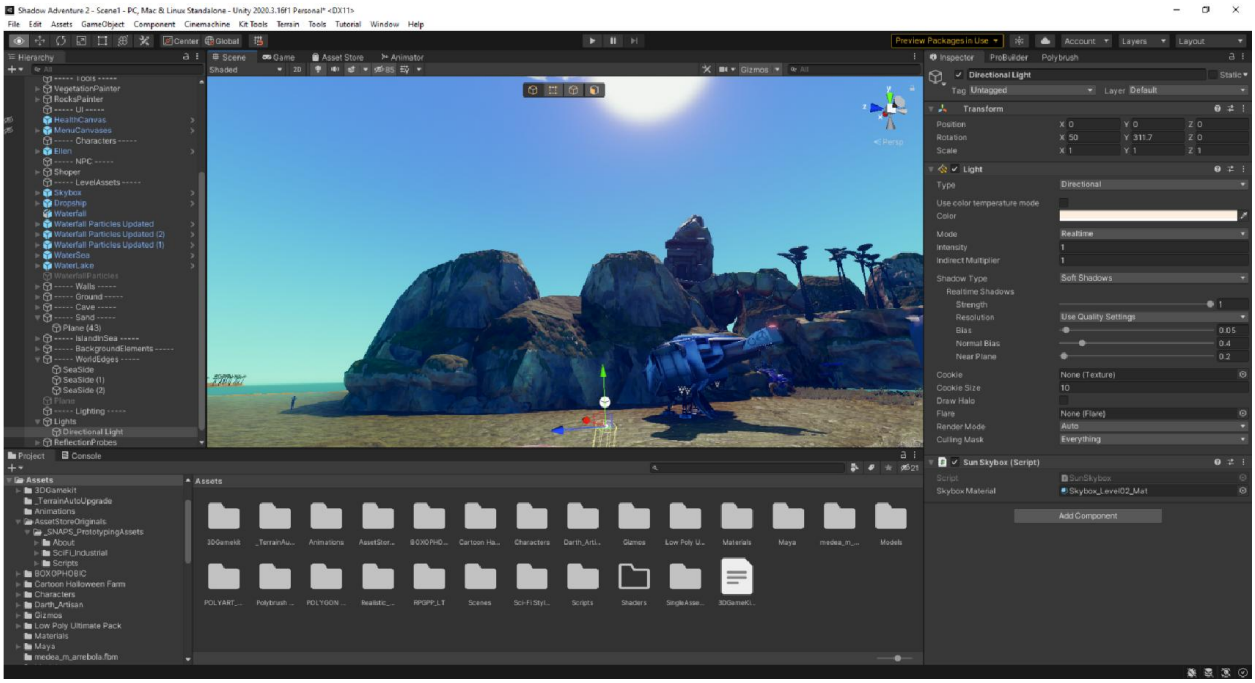

Рисунок 3.8 – Робота із джерелом світла

За допомогою модуля «Ефекти» та «Системи частинок» був створений водоспад.

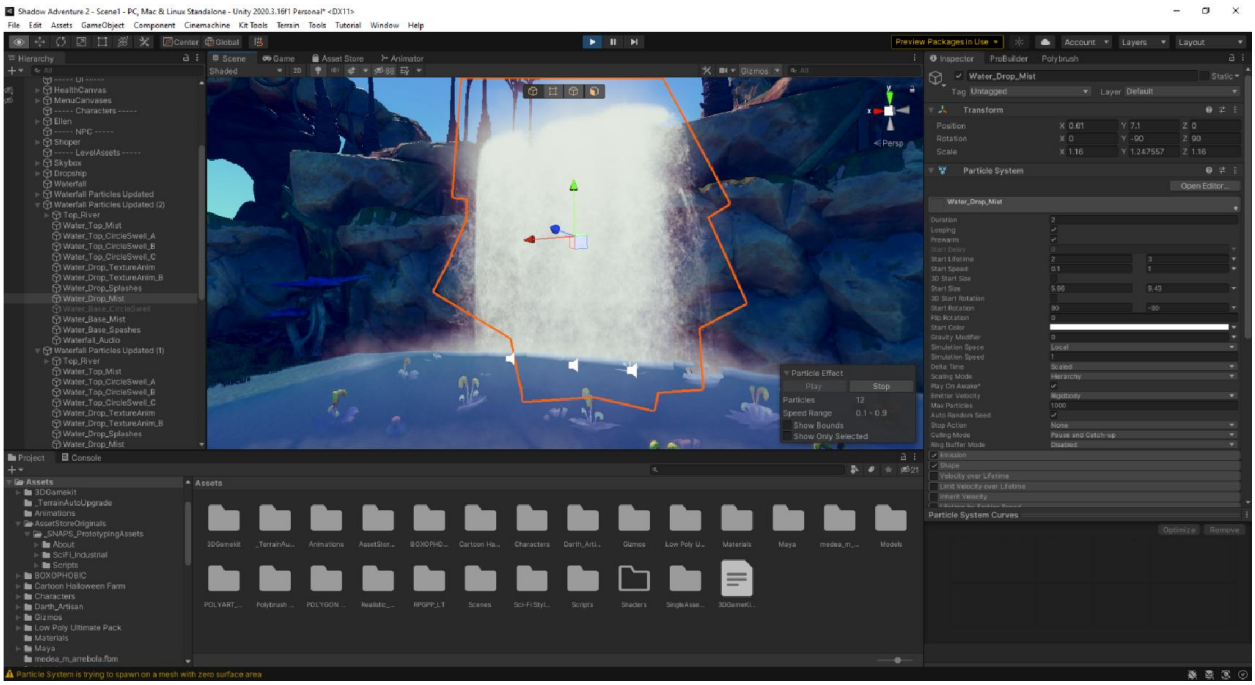

Рисунок 3.9 – Модель водоспаду реалізована за допомогою «системи частинок»

3.2 Додання персонажів до гри

Моделі персонажів були взяті із офіційного магазину Unity3D – AssetStore. Персонажі мають основні анімації: пересування по локації, стрибки, нанесення ударів, перебування на місці/очікування, отримання пошкоджень та програшу. Кожна модель має різні розміри та набір матеріалів і кольорів, що дозволяє легко ідентифікувати та розрізняти персонажів.

Рисунок 3.10 – Модель головного персонажа гри «Ангела 6»

Рисунок 3.11 – Модель дружнього персонажа Марсі

Рисунок 3.12 – Модель ворожого персонажа Воркотуна

Рисунок 3.13 – Модель нейтрального персонажа Голема

Рисунок 3.14 – Модель космічного корабля «Ангела 6»

3.3 Створення інтерфейсу

З допомогою модуля для роботи з інтерфейсом було створено головне меню**.**Натиснувши кнопку «Розпочати гру» відбувається завантаження локації і гравець може приступати до проходження. Кнопка «Вийти» закриває гру та повертає користувача до робочого столу.

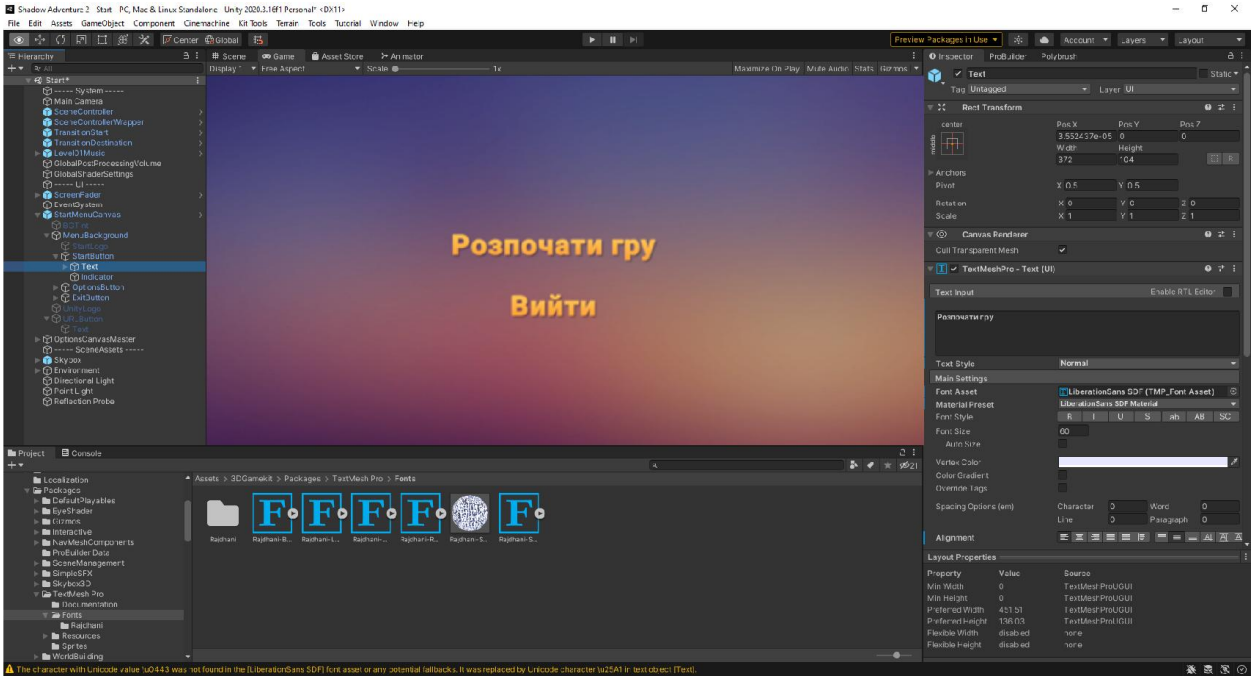

Рисунок 3.15 – Головне меню гри

В самій грі реалізовано меню паузи для зупинки ігрового процесу на певний час і подальшого продовження гри або закриття додатку і повернення на робочий стіл.

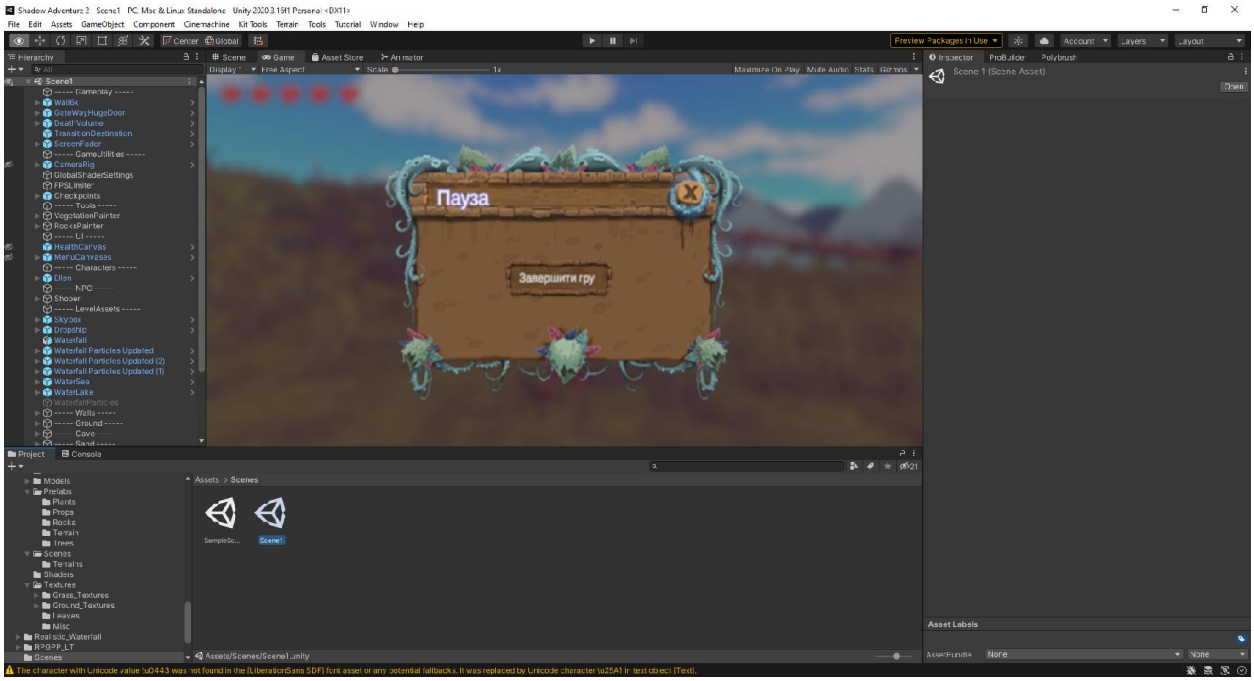

Рисунок 3.16 – Меню паузи в грі

Створено меню для відображення діалогів. Під час гри діалоги знаходяться в сцені, але являються неактивними. Коли персонаж гравця проходить тригер, діалогове вікно відображається на екрані.

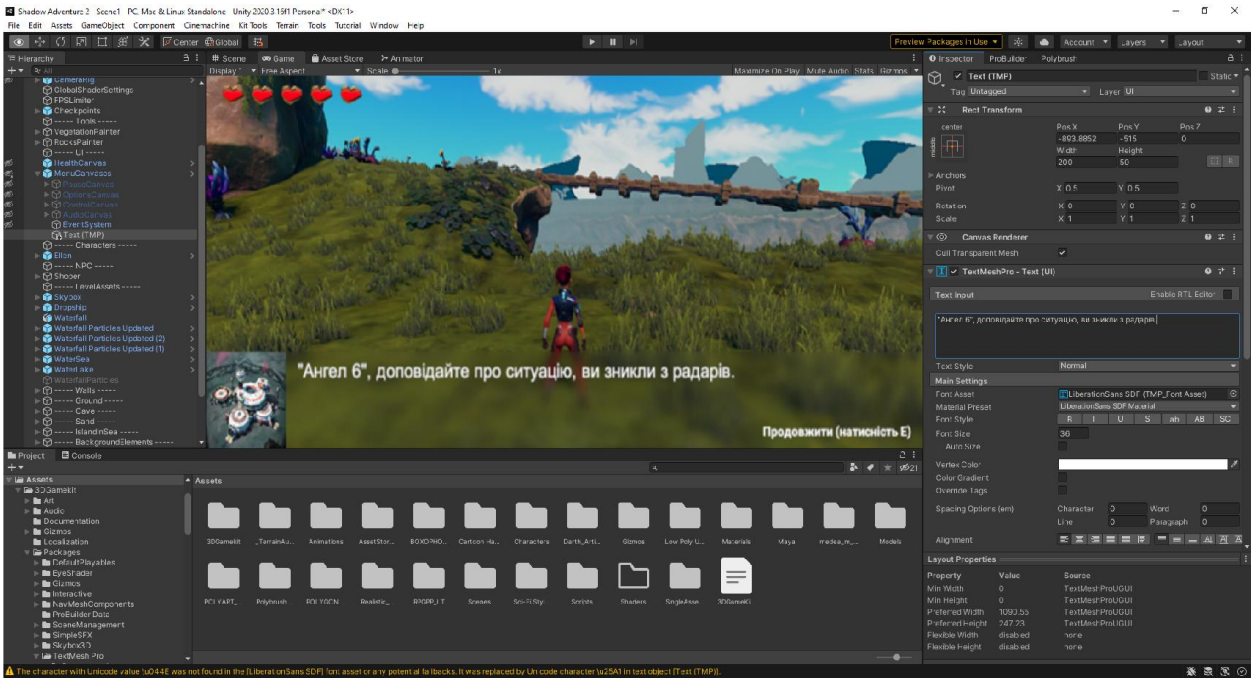

Рисунок 3.17 – Діалогове вікно після активації тригеру

Під час гри на екрані постійно знаходиться шкала здоров’я. Гравець може орієнтуватися по ній і планувати тактику битви: маючи не повний запас здоров’я можливо краще зіграти акуратно, спробувати підманити і битися з одним ворогом замість нападу на групу ворогів.

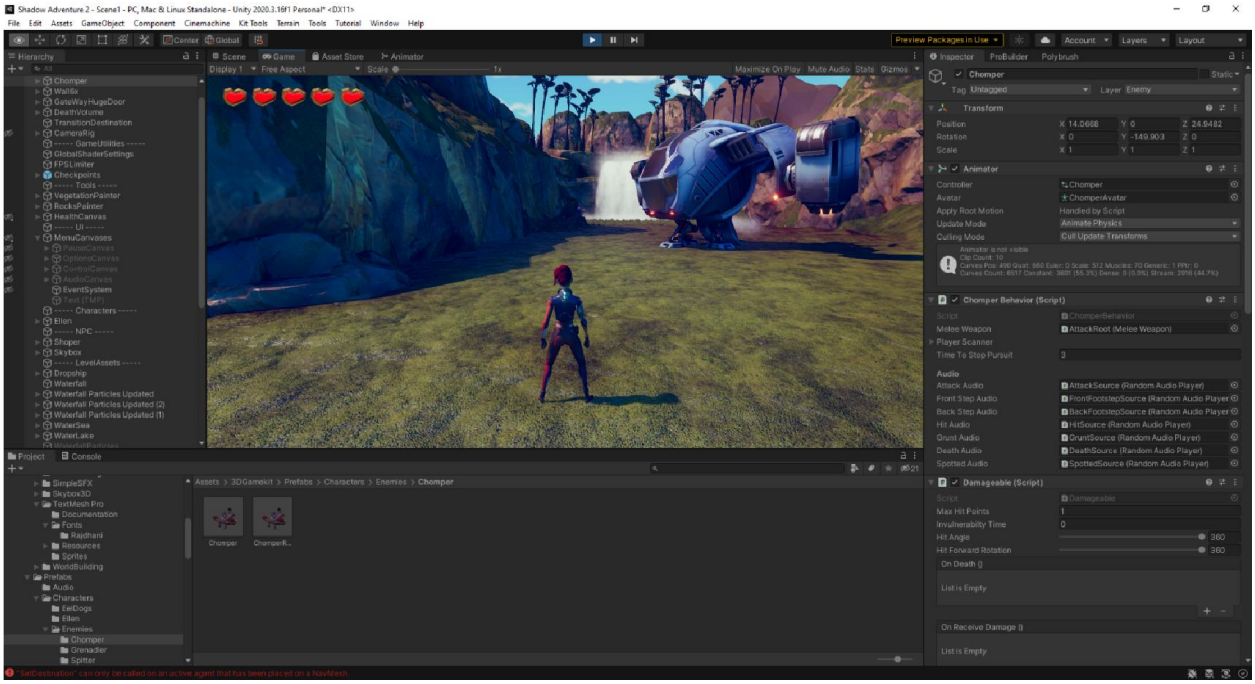

Рисунок 3.18 – Повний запас індикатора здоров’я

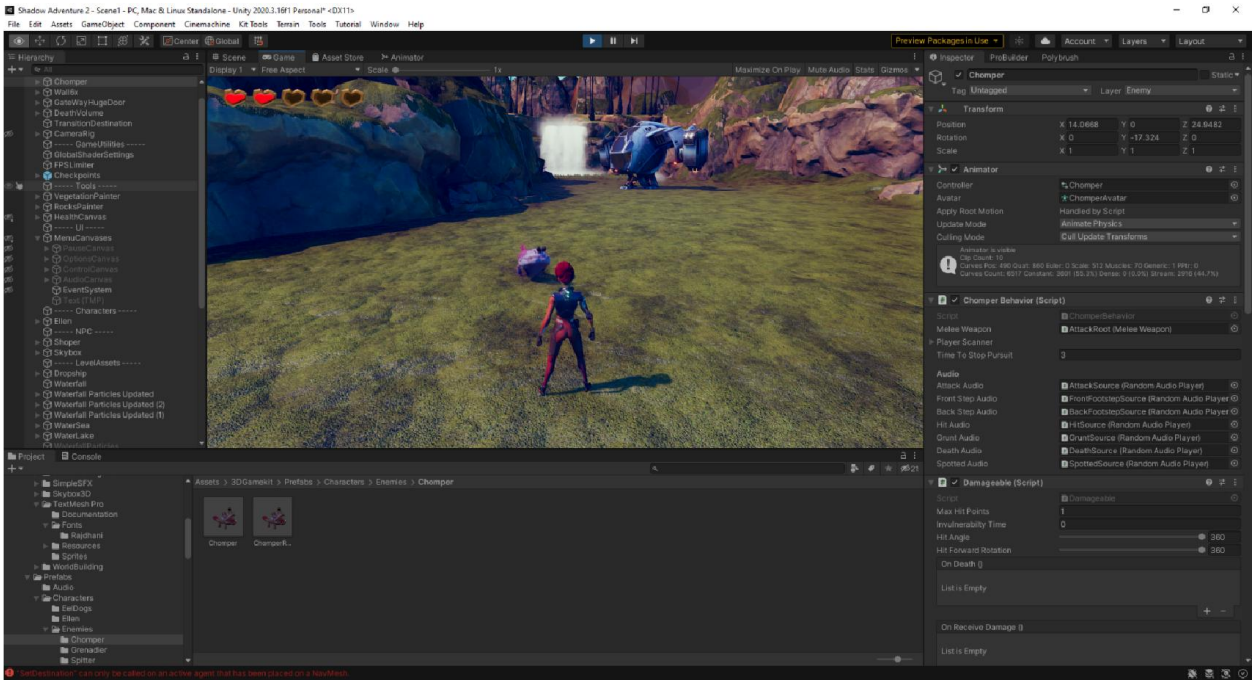

Рисунок 3.19 – Індикатор здоров’я після отримання пошкоджень

3.4 Створення системи подій

Для подій (поява діалогового меню, підказки, оновлення завдання, генерація нової групи ворогів, використання предмету) створено тригери за допомогою простих графічних фігур. Для тригерів відключені їх графічні відображення в сцені, але подібно стінам, що обмежують пересування гравця по локації їх фізичні моделі продовжуюсь працювати. Хоча є одна відмінність від фігур, що виступають в якості «стін» – персонаж може проходити крізь модель тригеру, тим самим активуючи його.

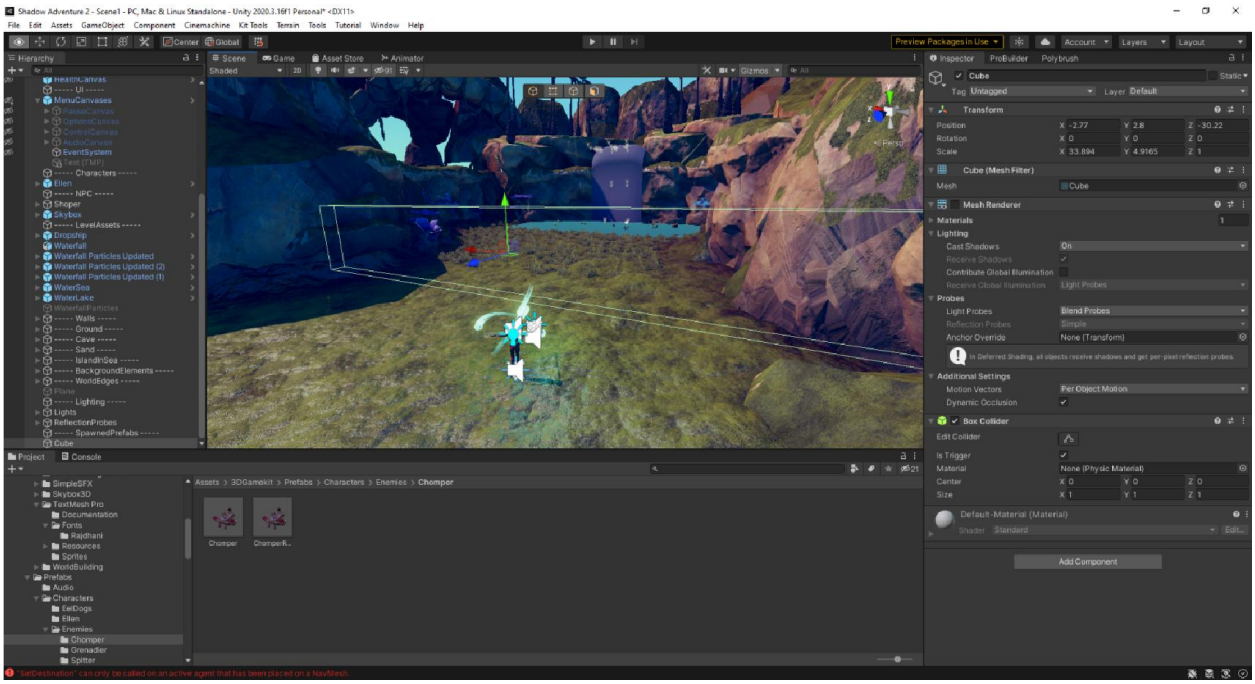

Рисунок 3.20 – Тригер для активації діалогового меню

3.5 Додання звуків до гри

Персонажі мають базовий набір звуків, що відтворюються відповідно до дій які вони виконують. Окрім цього звуки мають деякі ключові об’єкти та елементи навколишнього середовища. Додано музику на задній фон.

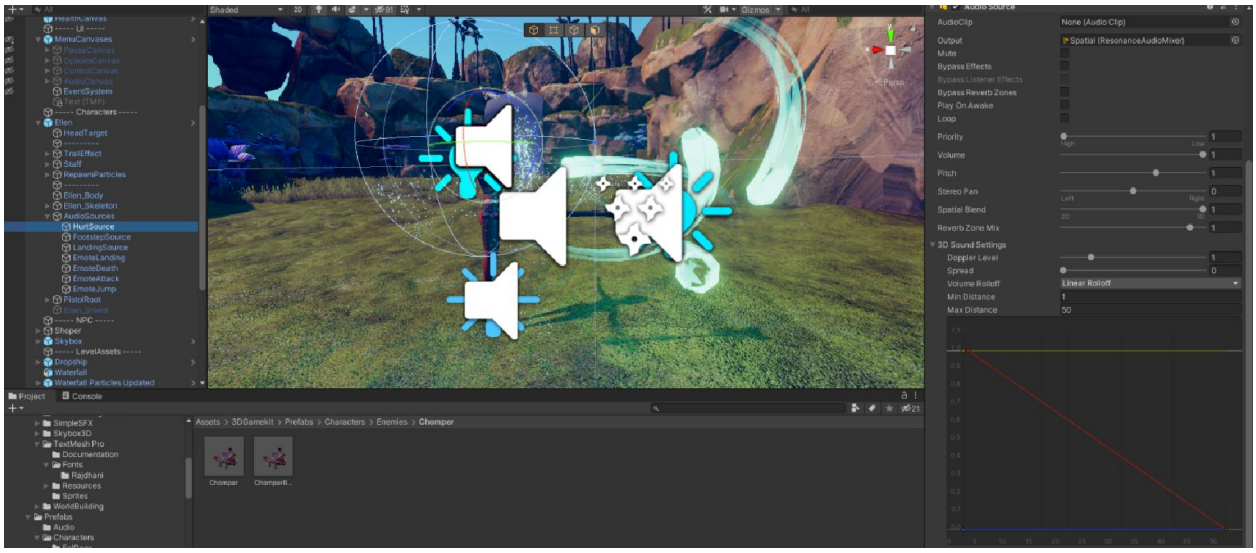

Рисунок 3.21 – Звукові властивості персонажа

3.6 Створення бойової системи та налаштування ворогів

Кожен персонаж має визначені значення сили атаки та запас здоров’я. Кожен ворожий удар забирає 1/5 від загальної кількість здоров’я головного персонажа. Воркотуни мають незначний запас здоров’я і помирають після одного влучного удару. Големи мають значно більший запас здоров’я – можуть витримати 15 ударів. Для головного персонажа та големів створено захисний щит, який активується після отримання пошкоджень. Він триває декілька секунд, проте дозволяє відійти на безпечну відстань або встигнути переломити результат битви.

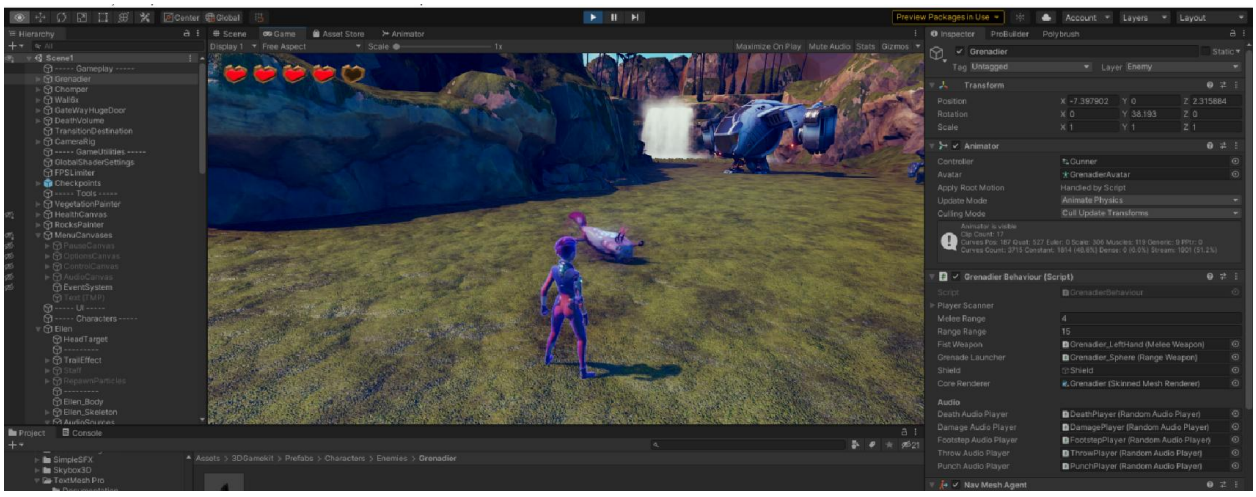

Рисунок 3.22 – Активація захисного щита після отримання пошкоджень

Ворожі персонажі мають поле зору. Якщо гравець потрапляє в межі зору, відбувається перехід до одного зі станів в залежності від поточним умов. За замовчуванням ворожі персонажі знаходяться в стані спокою. Якщо воркотун побачить гравця, він почне його переслідувати і буде намагатися вступити в битву. У разі якщо гравець встиг відійти на далеку відстань, воркотун втрачає гравця із поля зору і повертається до місця свого попереднього перебування. Големи нейтральні по відношенню до гравця, доки він не підібрав визначений предмет. Якщо після цього попасти в поле зору голема він спробує зупинити гравця, вступивши з ним в битву.

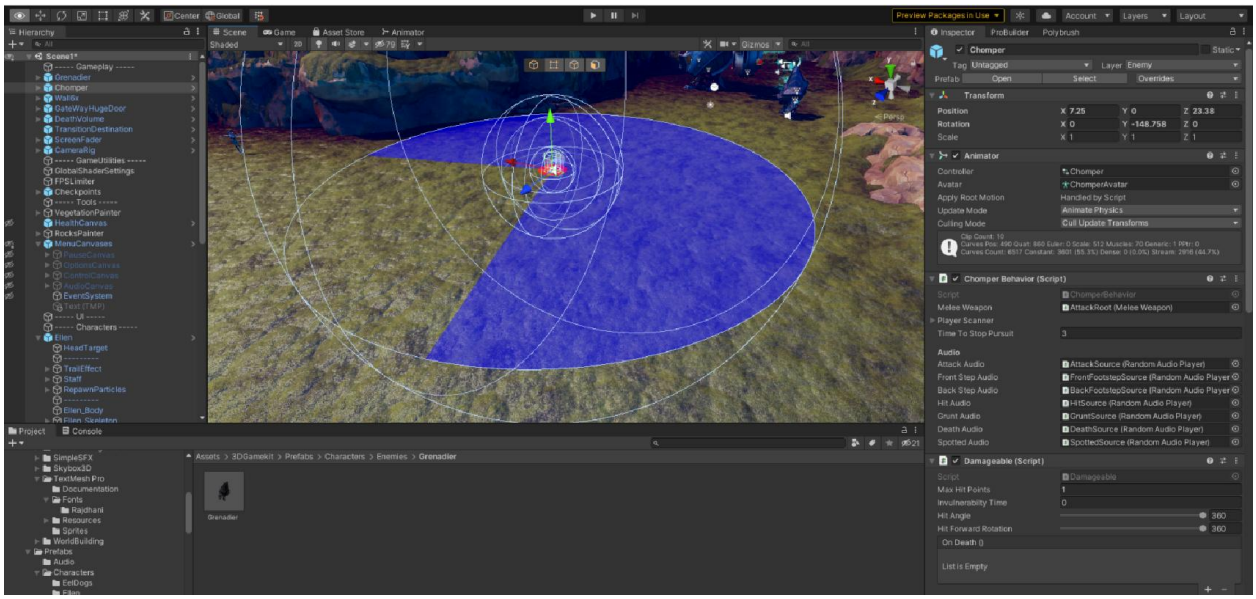

Рисунок 3.23 – Поле зору воркотуна

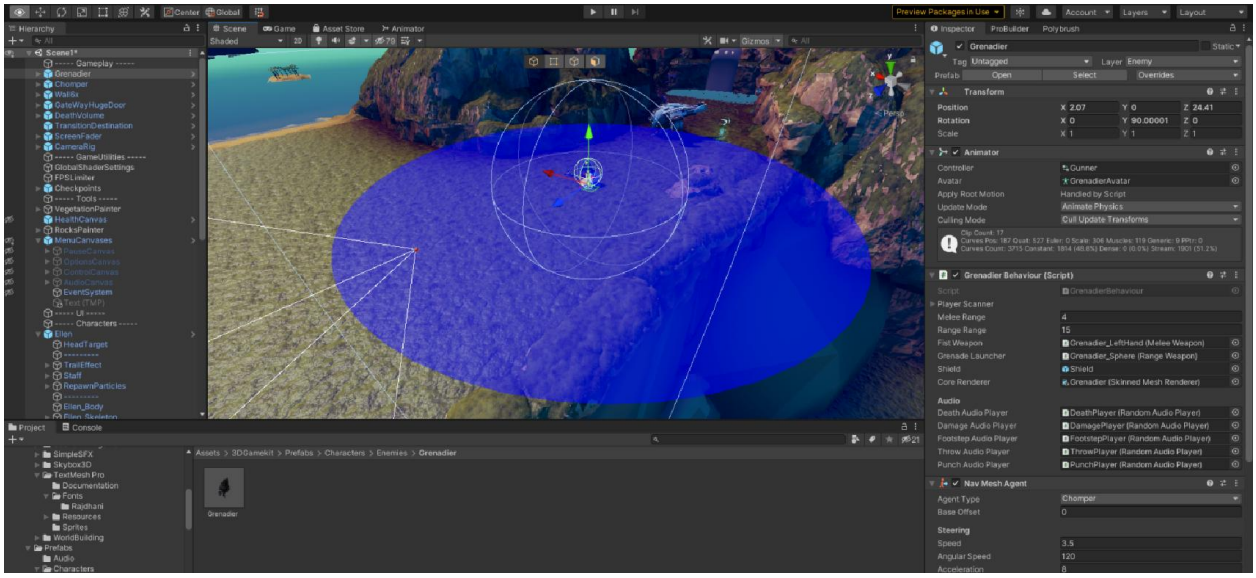

Рисунок 3.24 – Поле зору голема

За замовчуванням неігрові персонажі не можуть пересуватися по створеним локаціям. Щоб виправити це, необхідно створити навігаційну сітку за допомогою відповідного модуля «Інтелектуальний агент» і обрати пункт «Поверхня навігаційної сітки».

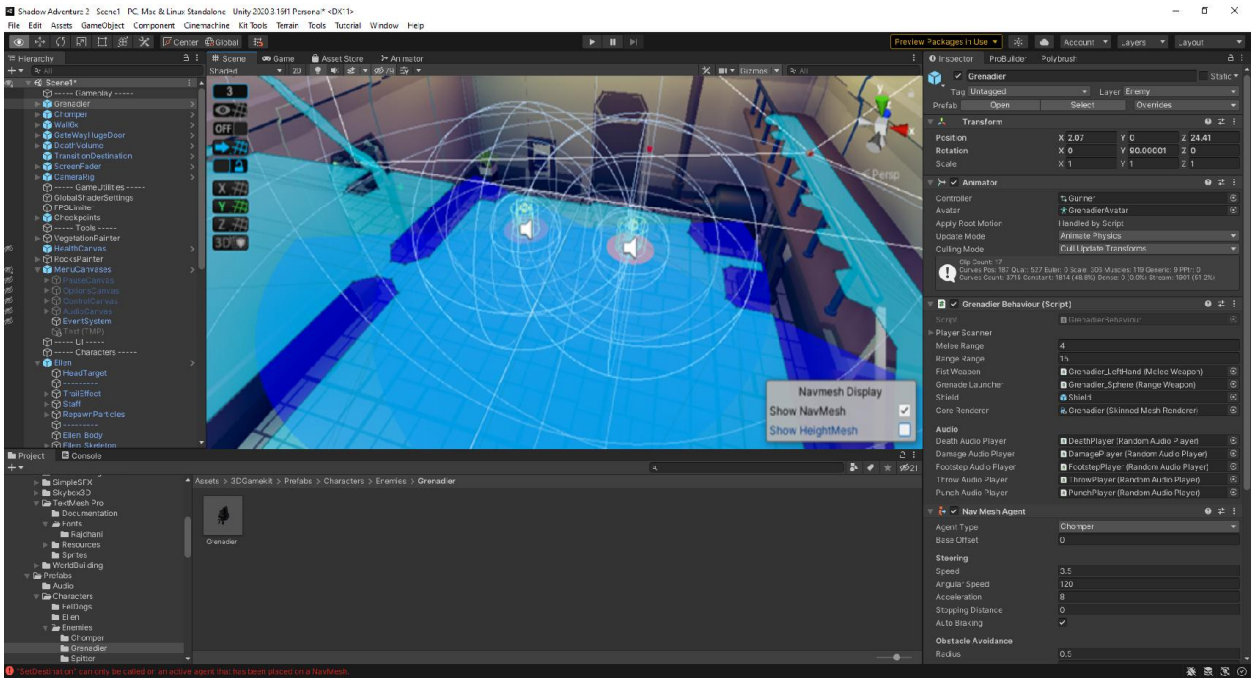

Рисунок 3.25 – Навігаційна сітка для неігрових персонажів

РОЗДІЛ 4
ТЕСТУВАННЯ

Тестування гри являє собою процес тестування програмного забезпечення для контролю якості. Основна функція ігрового тестування – виявлення і документування дефектів і помилок в роботі ПЗ.

Не існує одного чітко визначеного стандарту тестування ігор, більшість методологій створюються окремим розробниками і видавництвами відеоігор. Методології постійно вдосконалюють і вони можуть відрізнятися для різних типів ігор (наприклад, методологія тестування MMORPGбуде відрізнятися від тестування симулятору). Більшість методів, такі як модульне тестування перейняті із загальних методів тестування ПЗ. До найважливіших методологій тестування відеоігор можна віднести:

- функціональне тестування. Найчастіше саме воно асоціюється із ігровим тестування. Даний вид не потребує великих технічних знань. Тестувальники шукають загальні проблеми в самій грі або її користувацькому інтерфейсі: проблеми із стабільністю роботи, ігровими механіками і цілісністю виконання сценаріїв гри.
- тестування на відповідність. Метод розповсюджений у великих ігрових компаніях. Наприклад ліцензіати консольних платформ мають чітко визначені технічні вимоги: Sonyпублікує контрольний список технічних вимог (TRC), Microsoftпублікує вимоги до Xbox (XR), а Nintendoпублікує список «керуючих принципів» (Lotcheck). Певні вимоги носять технічний характер і виходять за межі тестування гри.
- тестування на сумісність. Проводиться для ПК додатків. Зазвичай проводить на останніх етапах розробки, оскільки сумісність залежить від фінальної версії гри. Проводять два етапи тестування: на початку бета-тестування, що мати час для вирішення проблем і трохи згодом, в бета-версії після усунення зауважень або під час релізу продукту. Тестування відбувається на різних конфігураціях АП. Зазвичай список комерційно

важливого забезпечення надає видавець. Тестування на сумісність гарантує, що гра буде працювати на різних конфігураціях АЗ та ПЗ. АЗ охоплює продукцію різних виробників і різні периферійні пристрої введення, такі як геймпади та джойстики.

- тестування продуктивності – комплекс варіантів тестування, метою якого є визначення дієздатності, стабільності, використання ресурсів і других атрибутів якості ПЗ в умовах різних сценаріїв використання і навантаження. Даний вид тестування дозволяє знаходити можливі слабкості і вади в системі з метою запобігти їхньому негативному впливу на роботу програми в умовах використання.
- тестування навантаження перевіряє ліміт системи, як наприклад кількість гравців на серверу, кількість активних об’єктів на екрані або кількість потоків, запущених для конкретної програми. Для даного виду тестування потребується велика кількість тестувальників або спеціальне додаткове ПЗ, яке імітує велику активність.

Із усіх доступних методів найбільш вдалим буде функціональне тестування. За його допомогою можна перевірити роботу основних ігрових механік та процесів гри.

Після проведення тестування не було виявлено жодних критичних помилок, весь функціонал працює як і планувалося. Набір тестових випадків, які були виконані, представлені у додатку В.

ВИСНОВОК

В результаті виконання дипломної роботи було здійснене проектування, реалізація та тестування однокористувацької тривимірної комп’ютерної гри в жанрі action-adventure. Був проведений збір та аналіз теоретичної інформації із галузі створення комп’ютерних ігор для визначення основних елементів, притаманних саме цьому жанру ігор. В якості ПЗ для створення гри було обрано ігровий рушій Unity3D. Програма складається із модулів, що дозволяють працювати із усіма необхідними компонентами (світло, звук, анімації, камера, графічні ефекти) та дозволяє створювати скрипти на мові програмування C#.

Перед реалізацією гри було проведено підготовчий етап: було створено сюжет гри, створено макети рівнів гри, був проведений пошук необхідних компонентів (моделі персонажів, елементи оточення, матеріали та текстури, звуки). Для полегшення роботи в редактор Unityбули встановлені додаткові плагіни: Cinemachine, ProBuilderта ProGrids.

Результатами реалізації стало наступне: створено головне меню гри, з якого гравець може розпочати проходження гри; побудовано локацію, що розподіляється на три рівні. На початку першого рівня наявні підказки, які виступають в ролі посібника із використання – гравцеві розповідають як керувати рухами персонажа та здійснювати атаки. Підказки та розповідь історії реалізовані за допомогою діалогового вікна. Присутні елементи користувацького інтерфейсу у вигляді шкали здоров’я персонажа гри та меню паузи для продовження або завершення гри. У гру додано дружелюбних та ворожих персонажів, створено сценарій взаємодії із персонажем гравця. Гра має логічне сюжетне завершення.

В процесі тестування не було виявлено критичних помилок. Персонаж виконує команди управління, діалогова система, взаємодія із іншими ігровими об’єктами та завантаження сцен працюють як і було сплановано.

Головною метою даного ПЗ було створення розважального продукту, що дозволить користувачам цікаво провести час та відпочити. Реалізований ПП можна вважати завершеним та готовим для безкоштовного розповсюдження, оскільки дана робота в першу чергу була орієнтована на отримання практичних навичок в області створення комп’ютерних ігор. Фінал гри побудований таким чином, що дозволяє продовжити підтримку гри шляхом створення сюжетного продовження, доданням нових механік та функцій.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. MatthewsV. TheManyDifferentTypesofVideoGames&TheirSubgenres [Електронний ресурс]/ VinceMatthews // IdTechBlog. – 2018. – Режим доступу до ресурсу: https://www.idtech.com/blog/different-types-of-video-game-genres.

2. TheEvolutionofVideoGameGenres [Електронний ресурс]/ GameDesigning // GameDesigningBlog. – 2020. – Режим доступу до ресурсу:https://www.gamedesigning.org/gaming/video-game-genres/.

3. Thinkwik. CryEnginevsUnrealvsUnity: SelecttheBestGameEngine [Електронний ресурс]/ ThinkwikGroup // MediumBlog. – 2018. – Режим доступу до ресурсу: https://medium.com/@thinkwik/cryengine-vs-unreal-vs-unity-select-the-best-game-engine-eaca64c60e3e.

4. CryEngineVManual[Електронний ресурс]// CRYTEKGmbH. – 2019. – Режим доступу до ресурсу:https://docs.cryengine.com/.

5. UnrealEditorManual[Електронний ресурс]// EpicGames. – 2020. – Режим доступу до ресурсу:https://docs.unrealengine.com/en-US/Engine/Editor/index.html.

6. WorkinginUnity[Електронний ресурс]// UnityTechnologies. – 2020. – Режим доступу до ресурсу:https://docs.unity3d.com/Manual/UnityOverview.html.

7. SoloGameStudio. ComparisonofGameEngines 2020 [Електронний ресурс]/ SoloGameStudioGroup // IndieGameDevBlog. – 2020. – Режим доступу до ресурсу:https://indiegamedev.net/2020/02/11/comparison-of-game-engines-2020/.

8. ScriptingOverview[Електронний ресурс]// UnityTechnologies. – 2020. – Режим доступу до ресурсу:https://docs.unity3d.com/Manual/ScriptingSection.html.

9. RoachJ. Howtomakeavideogame [Електронний ресурс]/ JacobRoach // DigitalTrendsBlog. – 2020. – Режим доступу до ресурсу:https://www.digitaltrends.com/gaming/how-to-make-a-video-game/.

10. QuinnZ. ABeginner'sGuideToMakingYourFirstVideoGame [Електронний ресурс]/ ZoeQuinn // KotakuBlog. – 2013. – Режим доступу до ресурсу:https://kotaku.com/a-beginners-guide-to-making-your-first-video-game-5979539.

11. WhatIsGameDevelopment? [Електронний ресурс]// freeCodeCamp. – 2019. – Режим доступу до ресурсу:https://www.freecodecamp.org/news/what-is-game-development/

12. VideoGameHistory? [Електронний ресурс]// History. – 2017. – Режим доступу до ресурсу:https://www.history.com/topics/inventions/history-of-video-games

13. Computerandvideogames [Електронний ресурс] // ScienceDaily. – 2020. – Режим доступу до ресурсу: https://www.sciencedaily.com/terms/computer_and_video_games.htm

14. 5 Game Elements That Create Effective Learning Games[Електронний ресурс]/G-Cube Webwide Software PvtLtd // eLearning IndustryBlog. – 2020. – Режим доступу до ресурсу:https://elearningindustry.com/5-game-elements-create-effective-learning-games

15. TheBeginnersGuidetoVideoGameDevelopment[Електронний ресурс]// Gamedesigning – 2020. – Режим доступу до ресурсу:https://www.gamedesigning.org/video-game-development/

16. VideoGameHistoryTimeline[Електронний ресурс]// MuseumofPlayBlog. – 2020. – Режим доступу до ресурсу:https://www.museumofplay.org/about/icheg/video-game-history/timeline

17. The best action-adventure games on PC in 2020[Електронний ресурс]// PC GamersN. – 2020. – Режим доступу до ресурсу:https://www.pcgamesn.com/best-action-adventure-games

18. BestActionAdventureGamestoPlay[Електронний ресурс]// G2ABlog. – 2020. – Режим доступу до ресурсу:https://www.g2a.com/news/features/best-action-adventure-games-to-play/

19. Gibson J. Introduction to Game Design Prototyping and Development / Jeremy Gibson., 2017. – 1024 с.

20. Hocking J. Unity in Action: Multiplatform Game Development in C# with Unity 5 1st Edition / Joe Hocking., 2015. – 352 с.

21. What is an Action/Adventure Game? [Електронний ресурс]/ Gameranxgroup // GameranxBlog. – 2011. – Режим доступу до ресурсу: https://gameranx.com/features/id/3350/article/what-is-an-action-adventure-game/.

22. Top 10 MostPopularGameGenresintheWorld 2018[Електронний ресурс]// technavioBlog. – 2018. – Режим доступу до ресурсу:https://blog.technavio.com/blog/top-10-most-popular-game-genres.

23. Syed A. Which game engine you should choose? [Електронний ресурс]/ AsemSyed // TechnobyteBlog. – 2018. – Режим доступу до ресурсу:https://technobyte.org/which-game-engine-should-you-choose/.

24. GraphicsOverview[Електронний ресурс]// UnityTechnologies. – 2020. – Режим доступу до ресурсу:https://docs.unity3d.com/Manual/Graphics.html.

25. Artificialintelligenceinvideogames [Електронний ресурс] // Wikipedia. – 2021 – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Artificial_intelligence_in_video_games.

26. Creating a Simple AI with Unity and C#[Електронний ресурс]/ Lance Talbert // Redgate Hub. – 2018. – Режим доступу до ресурсу:https://www.red-gate.com/simple-talk/development/dotnet-development/creating-a-simple-ai-with-unity-and-c/

27. Bourg D. M.AI for Game Developers/ David M. Bourg, Glenn Seeman., 2004. – 400с.

28. Yannakakis G. N. Artificial Intelligence and Games/ Georgios N. Yannakakis, Julian Togelius., 2018. – 359с.

29. DaGraca M. Practical Game AI Programming / MicaelDaGraca., 2017. – 341 с.

30. Millington I.Artificial intelligence for games - Second Edition/ Ian Millington, John Funge., 2009. – 895 с.

31. Aversa D. Unity Artificial Intelligence Programming – Fourth Edition/ Dr. Davide Aversa, Aung SithuKyaw., 2018. – 238 с.

ДОДАТОК А
ВИХІДНІ КОДИ ПРОГРАМНОГО ЗАСОБУ

1. Скрипт управління ліфтом (MoveLift.cs)

```
usingSystem.Collections;
usingSystem.Collections.Generic;
using UnityEngine;

public class MoveLift :MonoBehaviour
{
    private float time = 7f;
    private float upTime = 15f;
    private bool checkTrigger = false;

    public GameObject player;
    public GameObject door;
    public GameObjectupDoor;
    public float doorSpeed = 0.535f;
    public float distance = 24.17f;


    void OnTriggerEnter(Collider myTrigger)
    {
        if (myTrigger.gameObject.name == "Ellen")
        {
checkTrigger = true;
        }
    }

    private void Update()
    {

        if (checkTrigger == true)
        {
            if (time >= 0)
            {
                if (time >= 3)
door.transform.Translate(Vector3.left * doorSpeed * Time.deltaTime);
                if (time <= 2)
                {
player.transform.Translate(0, distance, 0);
Debug.Log("Lift is moving");
                    time = 0;
                }
                time -= Time.deltaTime;
            }

            if (upTime>= 0)
            {
                if (upTime<= 6)
                {
upDoor.transform.Translate(Vector3.right * doorSpeed * Time.deltaTime);
                }
upTime -= Time.deltaTime;
            }
        }
    }
}
```

2. Скрипт управління подією (SingleEvent.cs)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class SingleEvent :MonoBehaviour
{
    public GameObject replic1;

    // Start is called before the first frame update
    void Start()
    {
        replic1.SetActive(false);
    }

    private void OnTriggerEnter(Collider myTrigger)
    {
        if (myTrigger.gameObject.name == "Ellen")
        {
            replic1.SetActive(true);
        }
    }

    // Update is called once per frame
    void Update()
    {
        if (Input.GetButtonDown("E"))
        {
            replic1.SetActive(false);
        }
    }
}
```

3. Скрипт управління діалогом на початку гри (ChangeReplics.cs)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ChangeReplics :MonoBehaviour
{
    private intcountReplics = 1;
    private bool checkFirstJump = false;

    public GameObject replic1;
    public GameObject replic2;
    public GameObject replic3;
    public GameObject replic4;
    public GameObject replic5;
    public GameObject replic6;
    public GameObject tip1;
    public GameObject tip2;

    void Start()
    {
        replic1.SetActive(true);
        replic2.SetActive(false);
        replic3.SetActive(false);
        replic4.SetActive(false);
        replic5.SetActive(false);
        replic6.SetActive(false);
```

```
        tip1.SetActive(false);
        tip2.SetActive(false);
    }

    void OnTriggerEnter(Collider myTrigger)
    {
        if (myTrigger.gameObject.name == "Ellen")
        {
            tip2.SetActive(true);

checkFirstJump = true;
        }
    }

    // Update is called once per frame
    void Update()
    {
        if (Input.GetButtonDown("E"))
        {
countReplics++;

            if (countReplics == 2)
            {
                replic1.SetActive(false);
                replic2.SetActive(true);
            }
            if (countReplics == 3)
            {
                replic2.SetActive(false);
                replic3.SetActive(true);
            }
            if (countReplics == 4)
            {
                replic3.SetActive(false);
                replic4.SetActive(true);
            }
            if (countReplics == 5)
            {
                replic4.SetActive(false);
                replic5.SetActive(true);
            }
            if (countReplics == 6)
            {
                replic5.SetActive(false);
                replic6.SetActive(true);
            }
            if (countReplics == 7)
            {
                replic6.SetActive(false);
                tip1.SetActive(true);
            }
        }

        if                  (Input.GetButtonDown("Horizontal")                  ||
(Input.GetButtonDown("Vertical")))
        {
            tip1.SetActive(false);
        }

        if (checkFirstJump == true &&Input.GetButtonDown("Jump"))
        {
            tip2.SetActive(false);
        }
    }
```

```
}
```

4. Скрипт відкриття дверей (OpenDoor.cs)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class OpenDoor :MonoBehaviour
{
    private float time = 5f;
    private bool checkTrigger = false;

    public float doorSpeed = 0.535f;
    public GameObject door;

    void OnTriggerEnter(Collider myTrigger)
    {
        if (myTrigger.gameObject.name == "Ellen")
        {
checkTrigger = true;
Debug.Log("Trigger on");
        }
    }

    // Update is called once per frame
    private void Update()
    {
        if (checkTrigger == true)
        {
            if (time >= 1)
            {
door.transform.Translate(Vector3.right * doorSpeed * Time.deltaTime);
                time -= Time.deltaTime;
            }
            else
            {
checkTrigger = false;
Debug.Log("Door is open");
}
        }
    }
}
```

5. Скрипт переходу до наступного рівня (LevelLoader.cs)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class LevelLoader :MonoBehaviour
{
    public Animator transition;

    public float transitionTime = 1f;

    void OnTriggerEnter(Collider myTrigger)
    {
        if (myTrigger.gameObject.name == "Ellen")
        {
```

```
LoadNextLevel();
        }
    }

    public void LoadNextLevel()
    {
StartCoroutine(LoadLevel(SceneManager.GetActiveScene().buildIndex + 1));
    }

IEnumeratorLoadLevel(intLevelIndex)
    {
transition.SetTrigger("Start");

        yield return new WaitForSeconds(transitionTime);

SceneManager.LoadScene(LevelIndex);
    }

    // Update is called once per frame
    //void Update()
    //{

    //}
}
```

6. Скрипт активації ліфту (ActivateLift.cs)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ActivateLift :MonoBehaviour
{
    private float distance = 2.145f;

    public GameObjectLiftButton;
    public GameObjectLiftDoor;

    void OnTriggerEnter(Collider myTrigger)
    {
        if (myTrigger.gameObject.name == "Ellen")
        {
LiftButton.GetComponent<Renderer>().material.color = Color.green;
LiftDoor.transform.Translate(distance, 0,0);
Debug.Log("Lift is working");
        }
    }
}
```

7. Скрипт керування додатковим світлом для персонажа (BodyLight.cs)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BodyLight :MonoBehaviour
{
    public GameObjectbodyLight;
```

```
    private bool switchLight = true;

    private void OnTriggerEnter(Collider myTrigger)
    {
        if (myTrigger.gameObject.name == "Ellen")
        {
            if (switchLight == true)
            {
bodyLight.SetActive(false);
switchLight = false;
            }
            else
            {
bodyLight.SetActive(true);
switchLight = true;
            }
        }

    }

}
```

8. Скрипт для управління репліками (DialogueManager.cs)

```
usingSystem.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class DialogueManager :MonoBehaviour
{
    public Text dialogueText;
    public GameObjectspaceBase;
    public GameObjectpersonEllen;

    private Queue<string> sentences;
    private bool switchSprites = false;

    void Start()
    {
        sentences = new Queue<string>();
    }

    public void Update()
    {
        bool pressKeyE = Input.GetButtonDown("E");

        if (pressKeyE)
        {
DisplayNextSentence();
        }
    }

public void StartDialogue(Dialogue dialogue)
{
Debug.Log("It’s Works");

sentences.Clear();

foreach (string sentence in dialogue.sentences)
        {
sentences.Enqueue(sentence);
        }
```

```
DisplayNextSentence();
    }

    public void DisplayNextSentence()
    {
        if (sentences.Count == 0)
        {
EndDialogue();
            return;
        }

        string sentence = sentences.Dequeue();
StartCoroutine(TypeSentence(sentence));
    }

IEnumeratorTypeSentence(string sentence)
    {
dialogueText.text = "";
foreach (char letter in sentence.ToCharArray())
        {
dialogueText.text += letter;
            yield return null;
        }
    }

    void EndDialogue()
    {
Debug.Log("End of work");
    }
}
```

9. Скрип взаємодії персонажа та смертельно небезпечних об’єктів (DeathVolume.cs)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

namespace Gamekit3D
{
    [RequireComponent(typeof(Collider))]
    public class DeathVolume :MonoBehaviour
    {
        public new AudioSource audio;


        void OnTriggerEnter(Collider other)
        {
var pc = other.GetComponent<PlayerController>();
            if (pc != null)
            {
pc.Die(new Damageable.DamageMessage());
            }
            if (audio != null)
            {
audio.transform.position = other.transform.position;
                if (!audio.isPlaying)
audio.Play();
            }
        }
```

```
        void Reset()
        {
            if (LayerMask.LayerToName(gameObject.layer) == "Default")
gameObject.layer = LayerMask.NameToLayer("Environment");
var c = GetComponent<Collider>();
            if (c != null)
c.isTrigger = true;
        }

    }
}
```

10. Скрипт для управління кнопками головного меню (MainMenu.cs)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class MainMenu :MonoBehaviour
{

    public void PlayGame ()
    {
SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);
    }


    public void QuitGame ()
    {
Debug.Log("QUIT!");
Application.Quit();
    }
}
```

11. Скрипт для управління рухами персонажа (PlayerInput.cs)

```
usingUnityEngine;
using System;
using System.Collections;
using Gamekit3D;


public class PlayerInput :MonoBehaviour
{
    public static PlayerInput Instance
    {
        get { returns_Instance; }
    }

    protected static PlayerInputs_Instance;

    [HideInInspector]
    public bool playerControllerInputBlocked;

    protected Vector2 m_Movement;
    protected Vector2 m_Camera;
    protected bool m_Jump;
    protected bool m_Attack;
    protected bool m_Pause;
    protected bool m_ExternalInputBlocked;
```

```
    public Vector2 MoveInput
    {
        get
        {
if(playerControllerInputBlocked || m_ExternalInputBlocked)
                return Vector2.zero;
            return m_Movement;
        }
    }

    public Vector2 CameraInput
    {
        get
        {
if(playerControllerInputBlocked || m_ExternalInputBlocked)
                return Vector2.zero;
            return m_Camera;
        }
    }

    public bool JumpInput
    {
        get         {         returnm_Jump&&         !playerControllerInputBlocked&&
!m_ExternalInputBlocked; }
    }

    public bool Attack
    {
        get        {        returnm_Attack&&        !playerControllerInputBlocked&&
!m_ExternalInputBlocked; }
    }

    public bool Pause
    {
        get { returnm_Pause; }
    }

WaitForSecondsm_AttackInputWait;
Coroutinem_AttackWaitCoroutine;

const float k_AttackInputDuration = 0.03f;

    void Awake()
    {
m_AttackInputWait = new WaitForSeconds(k_AttackInputDuration);

        if (s_Instance == null)
s_Instance = this;
        else if (s_Instance != this)
            throw new UnityException("There cannot be more than one PlayerInput
script.  The instances are " + s_Instance.name + " and " + name + ".");
    }


    void Update()
    {
m_Movement.Set(Input.GetAxis("Horizontal"), Input.GetAxis("Vertical"));
m_Camera.Set(Input.GetAxis("Mouse X"), Input.GetAxis("Mouse Y"));
m_Jump = Input.GetButton("Jump");

        if (Input.GetButtonDown("Fire1"))
        {
            if (m_AttackWaitCoroutine != null)
```

```
StopCoroutine(m_AttackWaitCoroutine);

m_AttackWaitCoroutine = StartCoroutine(AttackWait());
        }

m_Pause = Input.GetButtonDown ("Pause");
    }

IEnumeratorAttackWait()
    {
m_Attack = true;

        yield return m_AttackInputWait;

m_Attack = false;
    }

    public bool HaveControl()
    {
return !m_ExternalInputBlocked;
    }

    public void ReleaseControl()
    {
m_ExternalInputBlocked = true;
    }

    public void GainControl()
    {
m_ExternalInputBlocked = false;
    }
}
```

12. Скрипт, що відповідає за відображення шкали здоров’я (HealthUI.cs)

```
using System.Collections;
using UnityEngine;

namespace Gamekit3D
{
    public class HealthUI :MonoBehaviour
    {
        public Damageable representedDamageable;
        public GameObjecthealthIconPrefab;

        protected Animator[] m_HealthIconAnimators;

        protected readonlyintm_HashActivePara = Animator.StringToHash("Active");
        protected                  readonlyintm_HashInactiveState                  =
Animator.StringToHash("Inactive");
        protected const float k_HeartIconAnchorWidth = 0.041f;

IEnumeratorStart()
        {
            if (representedDamageable == null)
                yield break;

            yield return null;

m_HealthIconAnimators = new Animator[representedDamageable.maxHitPoints];
```

```
            for (inti = 0; i<representedDamageable.maxHitPoints; i++)
            {
GameObjecthealthIcon = Instantiate(healthIconPrefab);
healthIcon.transform.SetParent(transform);
RectTransformhealthIconRect = healthIcon.transform as RectTransform;
healthIconRect.anchoredPosition = Vector2.zero;
healthIconRect.sizeDelta = Vector2.zero;
healthIconRect.anchorMin += new Vector2(k_HeartIconAnchorWidth, 0f) * i;
healthIconRect.anchorMax += new Vector2(k_HeartIconAnchorWidth, 0f) * i;
m_HealthIconAnimators[i] = healthIcon.GetComponent<Animator>();

                if (representedDamageable.currentHitPoints<i + 1)
                {
m_HealthIconAnimators[i].Play(m_HashInactiveState);
m_HealthIconAnimators[i].SetBool(m_HashActivePara, false);
                }
            }
        }

        public void ChangeHitPointUI(Damageable damageable)
        {
            if (m_HealthIconAnimators == null)
                return;

            for (inti = 0; i<m_HealthIconAnimators.Length; i++)
            {
m_HealthIconAnimators[i].SetBool(m_HashActivePara, damageable.currentHitPoints>= i
+ 1);
            }
        }
    }
}
```

13. Скрипт управління візуальними ефектами під час атак (TimeEffect.cs)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

namespace Gamekit3D
{
    public class TimeEffect :MonoBehaviour
    {
        public Light staffLight;

        Animation m_Animation;

        void Awake()
        {
m_Animation = GetComponent<Animation>();

gameObject.SetActive(false);
        }

        public void Activate()
        {
gameObject.SetActive(true);
staffLight.enabled = true;
```

```
            if (m_Animation)
m_Animation.Play();

StartCoroutine(DisableAtEndOfAnimation());
        }

IEnumeratorDisableAtEndOfAnimation()
        {
            yield return new WaitForSeconds(m_Animation.clip.length);

gameObject.SetActive(false);
staffLight.enabled = false;
}
    }
}
```

ДОДАТОК В
ТЕСТОВІ СЦЕНАРІЇ

Таблиця В.1– Тестування головного меню гри

Опис		Перевірка роботи кнопок меню	
Передумови		Гра запущена	
№	Дія	Очікуваний результат	Фактичний результат
1.	Натиснути кнопку «Розпочати гру»	Відбудеться завантаження ігрової локації	Ігрова локація завантажена, можна розпочинати проходження гри
2.	Натиснути кнопку «Вихід»	Відбудеться вихід із гри без відображення помилок	Гра закрита, всі пов’язані з нею процеси зупинено.

Таблиця В.2– Тестування управління ігровим персонажем

Опис		Перевірка можливості керувати персонажем	
Передумови		Працює один із рівнів гри	
№	Дія	Очікуваний результат	Фактичний результат
1	2	3	4
1.	Затиснути клавішу «W»	Персонаж почне рух вперед	Персонаж рухається вперед відносно камери
2.	Відпустити клавішу «W»	Персонаж припинить рух	Персонаж зупинився
3.	Затиснути клавішу «S»	Персонаж почне рух назад	Персонаж рухається назад відносно камери

Продовження таблиці В.2

1	2	3	4
4.	Відпустити клавішу «S»	Персонаж припинить рух	Персонаж зупинився
5.	Затиснути клавішу «A»	Персонаж почне рухвліво	Персонаж рухається вліво відносно камери
6.	Відпустити клавішу «A»	Персонаж припинить рух	Персонаж зупинився
7.	Затиснути клавішу «D»	Персонаж почне рух вправо	Персонаж рухається вправо відносно камери
8.	Відпустити клавішу «D»	Персонаж припинить рух	Персонаж зупинився
9.	Натиснути клавішу «Space»	Персонаж здійснить стрибок	Персонаж здійснив одиночний стрибок
10.	Натиснути ліву клавішу миші	Персонаж виконає одиночну атаку	Персонаж здійснив одиночну атаку
11.	Швидко натиснути ліву клавішу миші тричі	Персонаж виконає серію ударів	Персонаж здійснив серію ударів

Таблиця В.3 – Перевірка роботи сюжетного тригеру

Опис		Перевірка тригеру, що відповідає за сюжет	
Передумови		Гравець запустив гру та натиснув кнопку «Розпочати гру», гра завантажила перший рівень	
№	Дія	Очікуваний результат	Фактичний результат
1.	Завантажився перший рівень гри	З’явиться діалогове вікно	Відображається діалогове вікно
2.	Натиснути клавішу «E»	Відбудеться перехід до наступної репліки	Відображається наступна репліка персонажа

Таблиця В.4– Перевірка роботи підказок

Опис		Перевірка тригеру, що відповідає за підказки в грі	
Передумови		Гравець знаходиться на ігровому рівні	
№	Дія	Очікуваний результат	Фактичний результат
1.	Персонаж дійшов до визначеного місця	З’явиться діалогове вікно та інформація, яку клавішу необхідно натиснути для того, щоб персонаж виконав дію	З’являється діалогове вікно та інформація, яку клавішу необхідно натиснути для того, щоб персонаж виконав дію
2.	Натиснути одну із клавіш, що зазначені у підказці	Персонаж виконає вказану дію, вікно із підказкою зникне	Персонаж виконав вказану дію, вікно із підказкою зникає

Таблиця В.5– Перевірка роботи бойової системи

Опис	Перевірка взаємодії персонажів бій час бою		
Передумови	Гравець знаходиться поруч із ворожим персонажем		
№	Дія	Очікуваний результат	Фактичний результат
1	2	3	4
1.	Персонаж потрапив у поле зору ворога	Ворог почне переслідувати гравця	Ворог переслідує гравця
2.	Персонаж знаходиться поруч із ворогом	Ворог буде атакуватигравця.	Ворог атакує гравця.
3	Персонаж атакує ворога	Ворог отримає пошкодження	Ворог отримує пошкодження.
4	Нанести ворогу потрібну кількість атак	Ворог втратить запас здоров’я та помре	Ворог втрачає запас здоров’я та помирає
5	Ворог наніс потрібну кількість атак	Персонаж гравця втратить запас здоров’я. Гра зупиниться, персонаж буде перенесений до останньої контрольної точки.	Персонаж гравця втратив запас здоров’я. Гра зупиняється, персонаж переноситься до останньої контрольної точки.

Таблиця В.6– Перевірка роботи інтерфейсу користувача

Опис		Перевірка роботи інтерфейсу користувача	
Передумови		Гравець знаходиться на ігровому рівні	
№	Дія	Очікуваний результат	Фактичний результат
1	2	3	4
1.	Натиснути клавішу «Esc»	Відбудеться пауза у грі. З’явиться меню із варіантами вибору «Продовжити» та «Вийти».	Відбувається пауза у грі. З’являється меню із варіантами вибору «Продовжити» та «Вийти».
2.	Натиснути кнопку «Продовжити»	Меню буде закрито, гра продовжиться	Меню закривається, гра продовжується.
3	Натиснути кнопку «Вийти»	Гравець вийде з гри	Гравець виходить з гри
4	Персонаж отримав пошкодження від ворога	Кількість позначок на шкалі здоров’я зменшиться.	Кількість позначок на шкалі здоров’я зменшилась.
5	Знайдено предмет для зцілення	Кількість позначок на шкалі здоров’я збільшиться до максимального значення	Кількість позначок на шкалі здоров’я збільшилась до максимального значення

Таблиця В.7– Перевірка роботи контрольних точок

Опис		Перевірка роботи контрольних точок	
Передумови		Гравець знаходиться на одному із рівні гри	
№	Дія	Очікуваний результат	Фактичний результат
1.	Отримати завдання	Контрольна точка буде встановлена на місці знаходження персонажа	Контрольна точка встановлена на місці знаходження персонажа
2.	Пройдена частина рівня, гравець пройшов до потрібного місця	Контрольна точка буде оновлена на поточне місцеположення персонажа	Контрольна точка оновлена на поточне місцеположення персонажа