

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ БАГАТОКОРИСТУВАЦЬКОЇ ГРИ
MULTIPLAYER GAME SOFTWARE

Виконав: студент 4 курсу, групи КНТ-120сп

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ЗОЛОТАРЕНКО Д.О.

(ПРИЗВИЩЕ та ініціали)

Керівник ОЛІЙНИК А.О.

(ПРИЗВИЩЕ та ініціали)

Рецензент КОЦУР М.І.

(ПРИЗВИЩЕ та ініціали)

2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2023 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ЗОЛОТАРЕНКА Дениса Олександровича

(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення багатокористувацької гри. Multiplayer Game Software

керівник проєкту (роботи) д.т.н., професор, ОЛІЙНИК Андрій Олександрович,
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “04” квітня 2023 року № 87

2. Строк подання студентом проєкту (роботи) 01 червня 2023 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

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

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ОЛІЙНИК А.О., професор		
Нормоконтроль	КАМІНСЬКА Ж.К., асистент		

7. Дата видачі завдання “4” квітня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки	2 тиждень	Розділ 2
4	Розробка структури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділи 3,4
6	Тестування та експериментальне дослідження програмного забезпечення.	5 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	6 тиждень	Додатки
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка)

(підпис) Денис ЗОЛОТАРЕНКО
(Імя ПРІЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Андрій ОЛІЙНИК
(Імя ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
83 с., 36 рис., 3 табл., 3 дод., 18 джерел.

POSTGRESQL, UNREAL ENGINE, БАГАТОКОРИСТУВАЦЬКА ГРА,
ГРАВЕЦЬ, КОМП'ЮТЕРНА ГРА, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ.

Об'єкт дослідження – процес розробки комп'ютерних ігор.

Предмет дослідження – програмні засоби багатокористувацьких комп'ютерних ігор.

Метою роботи є розробка програмного забезпечення багатокористувацької комп'ютерної гри.

Матеріали, методи та технічні засоби: мова програмування C++, ігровий рушій Unreal Engine 5, система керування базами даних PostgreSQL.

Результати. Розроблено програмне забезпечення багатокористувацької комп'ютерної гри за допомогою мови програмування C++ та ігрового рушія Unreal Engine.

Практична цінність роботи полягає у розробці програмного забезпечення багатокористувацької комп'ютерної гри, що характеризується простими ігровими правилами, надає можливість вибору декількох персонажів та підтримує одночасну взаємодію декількох гравців.

Висновки. Виконано проєктування програмного забезпечення багатокористувацької комп'ютерної гри. Розроблено програмне забезпечення багатокористувацької комп'ютерної гри. Здійснено тестування розробленого програмного забезпечення багатокористувацької комп'ютерної гри.

Галузь використання – комп'ютерні ігри.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 83 pages, 36 figures, 3 tables, 3 appendixes, 18 sources.

POSTGRESQL, UNREAL ENGINE, MULTIPLAYER GAME, PLAYER, COMPUTER GAME, SOFTWARE.

The object of research is the process of developing software for computer games.

The subject of the research is software for multiplayer computer games.

The purpose of this work is to develop the software for multiplayer computer game.

Materials, methods and technical tools: C++ programming language, Unreal Engine 5 game engine, PostgreSQL database management system.

Results. The software tools for computer gamebased on the C++ programming language have been developed.

The practical value of the work lies in the development of software for a multiplayer computer game, which is characterized by simple game rules, allows for the selection of several characters and supports the simultaneous interaction of several players.

Conclusions. The software for the multiplayer computer game has been designed. The software for the multiplayer computer game has been developed. The developed software for the multiplayer computer game has been tested.

Scope of use – computer games.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	8
Вступ.....	9
1 Аналіз предметної області.....	10
1.1 Розробка багатокористувацьких комп'ютерних ігор.....	10
1.2 Аналіз вимог до програмного забезпечення багатокористувацької гри. Дослідження програм-аналогів	22
1.3 Висновки за розділом 1	29
2 Матеріали і методи	30
2.1 Вибір мови програмування та інших технологій розробки	30
2.2 Вибір ігрового двигуна для створення програмного забезпечення багатокористувацької гри	32
2.3 Висновки за розділом 2	36
3 Опис програми	38
3.1 Структура програми багатокористувацької гри.....	38
3.2 Функціонування програмного забезпечення багатокористувацької гри	40
3.3 Проектування інтерфейсу взаємодії користувача з програмним забезпеченням багатокористувацької гри	44
3.4 Висновки за розділом 3	48
4 Експлуатація, тестування та експериментальне дослідження програми.....	49
4.1 Призначення й умови застосування програми	49
4.2 Характеристики програми та розгортання вебсерверу.....	50
4.3 Інструкція по експлуатації програми.....	52
4.3.1 Звернення до програми.....	52
4.3.2 Вхідні й вихідні дані	52
4.3.3 Повідомлення.....	52
4.4 Виконання програмного забезпечення багатокористувацької гри.....	52
4.5 Висновки за розділом 4	56
Висновки	58

Перелік джерел посилання	60
Додаток А Технічне завдання	62
Додаток Б Фрагмент тексту програми	66
Додаток В Слайди презентації	76

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

SQL – Structured Query Language;

UML – Unified Modeling Language;

ПЗ – програмне забезпечення;

СКБД – система керування базами даних.

ВСТУП

Для інтерактивних розваг та розвитку користувачів останнім часом активно використовуються комп'ютерні ігри різних жанрів, від стратегій та екшенів до рольових ігор та імітаційних симуляторів. Користувачі грають в комп'ютерні ігри за допомогою різних пристроїв, таких як персональні комп'ютери, ігрові консолі та мобільні пристрої. Багато ігор потребують підключення до Інтернету, щоб грати онлайн з іншими користувачами [1]-[4].

Останнім часом активно використовуються багатокористувацькі комп'ютерні ігри як різновид відеоігри, у якій одночасно бере участь багато гравців. Гравці можуть походити з різних місць і підключатися до гри через Інтернет або локальну мережу [1]-[4]. У багатокористувацьких іграх гравці грають разом або один з одним, співпрацюючи, виконуючи завдання або змагаючись один з одним за перемогу [1]-[4]. Проте існують деякі недоліки багатокористувацьких комп'ютерних ігор, пов'язані, зокрема з проблемами підтримки ігрового процесу багатьох користувачів. Інші проблеми таких ігор пов'язані з затримками та проблемами зі з'єднанням, недоліками безпеки даних, несумісності з різними пристроями та ін. Тому актуальною є проблема розробки програмного забезпечення підтримки багатокористувацької гри.

У дипломній кваліфікаційній роботі бакалавра розв'язується актуальне завдання розробки програмного забезпечення багатокористувацької гри.

Для досягнення поставленої мети у кваліфікаційній роботі бакалавра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів підтримки багатокористувацьких ігор;
- здійснити проектування програмного забезпечення багатокористувацької гри;
- створити програмне забезпечення багатокористувацької гри;
- виконати тестування розробленого програмного забезпечення багатокористувацької гри.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Розробка багатокористувацьких комп'ютерних ігор

Багатокористувацька гра – це відеогра, в яку грають один або кілька гравців разом. По суті, це не гра, це ігрове середовище, що забезпечує взаємодію між користувачами відповідно до певних правил. Всі гравці грають у ту саму багатокористувацьку гру, на тому самому сервері чи рівні й відчувають усе одночасно в реальному часі [1]-[6].

Багатокористувацькі ігри, як правило, полягають у реалізації змагальної стратегії, а також кількох інших режимів гри. Всі користувачі можуть грати в однакові сюжетні місії, допомагаючи один одному на важких етапах гри [1]-[6].

Для функціонування багатокористувацьких ігор часто використовується локальна мережа, що забезпечує підключення однієї системи до іншої, дозволяючи гравцям грати разом в одну гру. Ще одна популярна форма забезпечення багатокористувацьких ігор – «виділені сервери», створені розробниками ігор або іншими компаніями. Ці сервери працюють автоматично, без необхідності обслуговування технічним персоналом, ці виділені сервери схожі на орендовані кімнати, користувачам гри часто не потрібно нічого платити, але вони можуть налаштувати багатокористувацькі ігри для інших користувачів [1]-[6].

Компанії, що є власниками багатокористувацьких ігор або підтримують їх, мають повний контроль над своїми серверами, що дає їм значні переваги щодо дизайну ігор, швидкості Інтернету та апаратного забезпечення. Виділений хостинг в основному використовується великими компаніями. Багатокористувацькі ігри подолують розрив між суспільним становищем гравців і дозволяють різним людям бути членами різних спільнот гравців [1]-[6].

Розглянемо етапи процесу створення багатокористувацької гри. На першому етапі необхідно визначитися з концепцією гри. Перш ніж почати

кодувати та розробляти гру, необхідно переконатися, що визначено базу щодо того, якою буде гра, яке програмне забезпечення буде використовуватися, які вимоги до програмного та апаратного забезпечення користувачів (гравців) буде висувати гра, що розробляється [1]-[6].

На наступному етапі необхідно створити документ дизайну (проекту) гри. Проект гри являє собою документ, у якому міститься схема розроблюваної гри. Команда розробників гри повинна розуміти цей документ [1]-[6]. Документ щодо розробки гри дозволяє керівнику проекту викласти всі основи запланованої гри: історію, функції, персонажів, ігрову механіку тощо [1]-[6].

Після того, як документ дизайну гри розроблено, необхідно переконатися, що наявне апаратне забезпечення здатне забезпечити розробку багатокористувацької гри, далі необхідно узгодити художній стиль, визначити, як виглядатимуть гра та її ресурси. Ці характеристики суттєво впливають на структуру гри і будуть диктувати багато дій у процесі розвитку програмного забезпечення підтримки комп'ютерної гри [1]-[6].

Разом із зовнішнім виглядом програмного забезпечення підтримки комп'ютерної гри необхідно визначити, що будуть робити ігрові персонажі, а також як буде змінюватися навколишнє середовище, що оточує ігрових персонажів. Необхідно визначити вигляд персонажів, систему заохочень, покарань, навичок та бонусів. Це все треба зробити на ранній стадії, оскільки подальші зміни можуть призвести до плутанини в дрібницях на інших стадіях розробки програмного забезпечення підтримки комп'ютерної гри [1]-[6].

На наступному етапі необхідно обрати мову кодування. При розробці багатокористувацьких ігор можна використовувати самі можливості мов програмування, що й у будь-якому іншому програмному забезпеченні [1]-[6].

Потрібно визначити в документі дизайну гри, яким кодуванням буде розроблятися програмне забезпечення комп'ютерної гри, і дозволити команді розробників знати, яка парадигма програмування буде використовуватися [1]-[6].

Після цього необхідно обрати ігровий двигун, який буде використовуватися для створення та підтримки програмного забезпечення комп'ютерної гри. Різні ігрові двигуни можуть бути досить продуктивними при підтримці комп'ютерних ігор на різних системах з граною графікою та, як правило, високим бюджетом [1]-[6].

Наступний етап пов'язаний з налаштуванням програмного забезпечення багатокористувацької гри. Кожен двигун має свою бажану мову кодування, яку вони використовують. Наприклад, у випадку Unity мовою програмування зазвичай є C#, тому при розробці комп'ютерних ігор з використанням цього двигуна необхідно, щоб команда розробників мала досвід програмування цією мовою [1]-[6].

Ще один важливий аспект налаштування гри для кількох гравців пов'язаний зі знайомством з функціями керування активами та сховищами ресурсів, за допомогою чого можна розробити дійсно унікальне програмне забезпечення багатокористувацької гри. Також настійно рекомендується ознайомитися з тонкощами структури двигуна, його роботи та можливостей [1]-[6].

Перш ніж почати безпосередньо розробку програмного забезпечення багатокористувацької комп'ютерної гри варто переконатися, що всі документи та інші активи ідеально впорядковано в різних папках. Важливо визначити основний фон для розроблюваної багатокористувацької гри. Потрібно позначити фон гри як щось, що легко запам'ятовується. Потім потрібно масштабувати інтерфейс користувача, щоб переконатися, що фон залишається позаду та не заважає іншим аспектам програмного забезпечення багатокористувацької комп'ютерної гри [1]-[6].

Потім, використовуючи мережеві засоби можна почати створювати прототип програмного забезпечення багатокористувацької комп'ютерної гри. Використання локальної мережі є чудовим способом перевірити підключення до Інтернету для розроблюваного програмного забезпечення багатокористувацької комп'ютерної гри. Використовуючи локальну мережу,

можна перевірити, як працюватиме сервер і наскільки добре інші користувачі можуть підключитися до гри для кількох гравців. Швидше за все, для цього знадобиться окремий клієнт локальної мережі [1]-[6].

Наступним кроком є власне тестування мережі при роботі з програмним забезпеченням багатокористувацької комп'ютерної гри. Після входу в режим гри необхідно створити дублікат гри, що дозволить краще розуміти, чи запускатиметься багатокористувацька гра та чи працює вона коректно [1]-[6].

Потім потрібно імпортувати персонажів, предмети та ресурси у збірку програмного забезпечення багатокористувацької комп'ютерної гри. Це початок того, як буде виглядати гра. Необхідно визначитися, це буде двовимірна пригода з бічним прокручуванням чи тривимірний екшен. Рух, очевидно, є одним із найважливіших аспектів багатокористувацької комп'ютерної гри. Щоб запрограмувати рух, потрібно використовувати базові знання з фізики та програмування [1]-[6]. Усі об'єкти, для яких необхідно забезпечити можливість зробити хід, потрібно включити в мережевий менеджер, щоб вони залишалися в частині ігрового процесу для кількох гравців [1]-[6].

Аспекти ігрових двигунів дозволяють передавати об'єкт на різні сервери в режимі онлайн, гарантуючи, що він працює узгоджено з різними серверами та з'єднаннями [1]-[6].

Потрібно переконатися, що зіткнення об'єктів є реалістичним, тому при розробці програмного забезпечення багатокористувацької комп'ютерної гри необхідно додати відповідні функції разом із типом перспективи, у якій знаходиться гра (двовимірна чи тривимірна) [1]-[6].

При розробці програмного забезпечення багатокористувацької комп'ютерної гри можна використовувати безкоштовні двигуни як засіб для експериментального дослідження ідей та проєктів гри до їх повноцінної реалізації. Отже, багато ігрових двигунів можуть суттєво допомогти у розробці ігор для кількох гравців [1]-[6].

Відзначимо, що різні двигуни можуть давати різні результати, і велика частина роботи залежить виключно від того, який тип багатокористувацької гри розробляється, а також від графічних можливостей розроблюваної гри [1]-[6].

При створенні програмного забезпечення багатокористувацької комп'ютерної гри необхідно визначити базові елементи:

- правила та цілі. Правила комп'ютерної гри визначають, що можна робити у грі та яким чином можна здійснювати ті чи інші дії. Правила включають структуру або стиль, у якому ведеться гра. Правила контролюють розвиток гри та визначають різні взаємодії, які може мати гра. У грі також потрібні цілі, які гравець повинен виконати [1]-[6];

- реквізити та пристрої: рамки, створені правилами, підкріплюються та вдосконалюються реквізитами та пристроями (апаратним забезпеченням). Часто різні рівні гри вимагають власних реквізитів, а іноді, і пристроїв. Гравець споживає матеріали та реквізити, а пристрої використовує для досягнення цілей гри [1]-[6].

Наприклад, правила можуть бути щодо кількості гравців, тривалості гри та способу її проведення. Прикладом мети гри може бути забиття більшої кількості голів, ніж команда суперника за відведений час гри. Прикладом реквізиту є м'яч, а пристроїв (інструментів, екіпірування) – форма гравця [1]-[6].

Екземпляром багатокористувацької комп'ютерної гри є конкретна гра, яка відбувається з певними гравцями у певний проміжок часу. Кожен екземпляр багатокористувацької комп'ютерної гри, як правило, є унікальним, але всі вони засновані на одній концепції гри. Ігри відрізняються одна від одної з точки зору розвитку гри та ігрового процесу [1]-[6].

Важливою характеристикою багатокористувацької комп'ютерної гри є соціальні відносини. Соціальна взаємодія у комп'ютерній грі є значущим двостороннім спілкуванням декількох користувачів, що є ключовим аспектом до успішної багатокористувацької гри. В деяких багатокористувацьких іграх

соціальні взаємодії не мають ніякого значення, наприклад, в іграх, де інтерфейс дозволяє гравцям бачити лише ігрове поле, зокрема, в шахах. Ігри для соціальної взаємодії – це ігри, у яких ігри зі штучним інтелектом представляють людей-супротивників без урахування людських відмінностей [1]-[6].

Існують багатокористувацькі ігри з високою соціальною взаємодією. У таких іграх головна мета може бути досягнута лише за наявності тих чи інших соціальних відносин між учасниками. Наприклад, у класичній грі шаради гравці висловлюють або представляють певні ідеї або об'єкти, щоб вгадати, що представляють інші гравці [1]-[6].

Три елементи дизайну комп'ютерної гри (гравці, інструменти та реквізити, правила та цілі) можуть впливати на рівень соціальної взаємодії у грі. Якщо соціальна взаємодія є частиною розробки гри, необхідно визначити, як правила гри впливають на соціальну взаємодію та які інструменти та обладнання впливають на соціальну взаємодію у комп'ютерній грі [1]-[6].

Соціальна взаємодія заохочується, коли правила комп'ютерної гри заохочують гравців до соціальної взаємодії. Природна соціальна взаємодія виникає, коли гравці комп'ютерної гри самостійно вирішують спілкуватися, при чому правила комп'ютерної гри не вимагають такої дії, і це відбувається лише за умови участі гравців [1]-[6].

Події в ігровому просторі можуть бути важливою частиною соціальної взаємодії. В електронних багатокористувацьких іграх проблема забезпечення соціальної взаємодії є певною проблемою, оскільки прогрес у мережевих технологіях централізує усіх гравців [1]-[6].

Важливими функціями програмного забезпечення багатокористувацької комп'ютерної гри є функції, що забезпечують конкуренцію та співробітництво. Багатокористувацькі ігри передбачають конкуренцію і співпрацю людей, що неможливо в одиночних іграх. Змагання – це спроба учасника комп'ютерної гри досягти оголошеної перемоги. Важливо розробити програмні функції, що реалізують та визначають чітку різницю між

перемогою та поразкою у комп'ютерній грі. З іншого боку, в кооперативних іграх учасники намагаються досягти спільної мети. Часто це єдиний спосіб, яким гравці можуть допомогти іншим досягти своїх цілей [1]-[6].

Важко знайти суто кооперативну комп'ютерну гру, більшість кооперативних ігор є змагальними, але правила або цілі гри змушують гравців співпрацювати, якщо вони хочуть перемогти. Учасники таких кооперативних ігор повинні не тільки працювати разом, але й обмінюватися інформацією, яка допомагає їм зрозуміти, як змінюється навколишнє середовище гри. Жоден гравець не може зібрати інформацію, необхідну для розуміння всіх подій гри, не працюючи зі своїми членами команди [1]-[6].

Конкуренція та співпраця не обов'язково повинні протиставлятися одне одному. Організація комп'ютерної гри визначає рівень конкуренції та співпраці. Наприклад, у командних комп'ютерних іграх всі гравці в команді повинні працювати разом, але змагатися з іншими командами одночасно. Ще один спосіб створити кооперативні/змагальні ігри – не мати однакових цілей гри. Гравці, які прагнуть досягти своїх цілей, не можуть перешкоджати іншим гравцям досягти їх цілі. Гравці співпрацюють, коли це їм найбільш вигідно, і змагаються, коли це необхідно [1]-[6].

Співпраця та конкуренція у програмному забезпеченні багатокористувацької комп'ютерної гри визначаються якостями гравців та правилами гри. Правила гри відображають зміст гри з точки зору конкуренції та права на співпрацю, але гравці вирішують, співпрацювати чи конкурувати. Наприклад, у кооперативній комп'ютерній грі правила визначають тип взаємодії між гравцями. З правилами кооперативної багатокористувацької комп'ютерної гри, спрямованими на заохочення співпраці, а не на хорошу стратегію, пошук сильних дипломатів є ключовим для побудови правильного альянсу та знання, коли брати на себе відповідальність. Отже, при розробці програмного забезпечення багатокористувацької комп'ютерної гри важливо передбачити функції та програмні модулі, що реалізують можливість співпраці та конкуренції, а також взаємодії між гравцями [1]-[6].

При розробці програмного забезпечення багатокористувацької комп'ютерної гри варто реалізувати функції, що забезпечують синхронність. Гра визначається як одночасна, коли потрібна одночасна участь усіх гравців. При цьому усі гравці знають, що вони грають у багатокористувацьку комп'ютерну гру, в якій одночасно беруть участь інші люди. Більшість багатокористувацьких комп'ютерних ігор одночасними [1]-[6].

У синхронній грі беруть участь усі гравці одночасно. Дії зазвичай не виконуються одночасно, і кожен гравець виконує хід послідовно. Багато неелектронних багатокористувацьких ігор є такими, оскільки вони, наприклад, мають реверсну систему. Є ігри, які стирають цю відмінність, наприклад, гра з різними учасниками, де гравці, які входять і виходять до ігрового віртуального середовища, не залежать один від одного, при цьому не обов'язково, щоб усі гравці були одночасно в ігровому середовищі. Такі типи ігор вважаються асинхронними [1]-[6].

Важливо, щоб обладнання та реквізити комп'ютерної гри підтримували синхронність, якщо у цьому є потреба. Наприклад, якщо багатокористувацька комп'ютерна гра має різні рівні синхронізації, затримка мережі може спричинити деякі проблеми з продуктивністю [1]-[6].

Програмні функції для координації реалізують процес контролю за перебігом комп'ютерної гри. Ігри можуть бути координовані за допомогою програмного модулю або людини, або може бути реалізовано розподілене керування грою. Як правило, арбітри (програмні модулі чи адміністратори комп'ютерної гри) вирішують, коли починати, коли закінчувати, коли зупиняти, і сприяють дотриманню певних правил. Деякі децентралізовані комп'ютерні ігри поширюються серед усіх гравців та працюють без суддів. Такі ігри відбуваються у децентралізованому форматі, де всі підсистеми працюють разом, щоб забезпечити підтримку ігрового процесу [1]-[6].

Важливо розуміти, що інструменти та реквізит комп'ютерної гри можуть впливати на тип втручання щодо меж процесу координації гри. Наприклад, деякі багатокористувацькі комп'ютерні ігри потребують

центрального сервера для зберігання інформації про прогрес у грі, тому ігровий інструмент має бути встановлений на центральному сервері [1]-[6].

Емпіричне правило, за яким можна визначити, чи є залежність комп'ютерної гри від обладнання та реквізитів полягає в тому, щоб змоделювати, чи можна грати у ту саму комп'ютерну гру з іншим об'єктом. Вважається, що комп'ютерна гра мало залежить від обладнання та реквізитів, якщо зміна апаратного забезпечення (комп'ютера) та матеріалів не вплинуть на продуктивність гри. Наприклад, у багатокористувацьких комп'ютерних іграх використовують комп'ютери, щоб допомогти з обчисленнями та робочим процесом гри, але суть гри не змінюється при зміні одного комп'ютеру користувача на інший [1]-[6].

Серед багатокористувацьких комп'ютерних ігор, особливо змагальних ігор, є такі ігри, де один гравець може не мати доступу до інформації, до якої мають доступ інші гравці ігри. Ця суперечлива інформація може призвести до існування метаігор. Метаігри – це ігри, схожі на звичайні ігри на казуальному рівні, але є центром розвитку більшої гри. Якщо гравець є персонажем у метагрі, то правила можуть суперечити інформації, доступній гравцеві [1]-[6].

Іншим типом мета-ігри є багатокористувацькі комп'ютерні ігри, які виходять за межі одного конкретного екземпляра гри, але не можуть існувати без нього. Наприклад, на чемпіонаті світу з футболу команди повинні спочатку пройти кваліфікацію, а потім взяти участь у чемпіонаті, організованому особливим чином. Потім у нас є правила для футбольної гри та (мета) правила для чемпіонату світу [1]-[6].

Отже, при розробці програмного забезпечення багатокористувацької комп'ютерної гри необхідно дотримуватися процесу, що розпочнеться з визначення мети та теми гри. Після цього і для кожної характеристики гри необхідно прийняти певні дизайнерські рішення. Ці рішення повинні керуватися елементами, пов'язаними з характеристиками багатокористувацьких ігор [1]-[6]. Після того, як ці дизайнерські рішення будуть прийняті, необхідно розробити концептуальний документ для

комп'ютерної гри, який описує ідею гри, а також служить інструментом просування та виведення гри на ринок.

На етапі визначення теми гри у процесі проєктування є лише розпливчате уявлення про гру. На цьому етапі не відомо про форму чи спосіб гри, необхідно визначити правила гри та те, як вона працює. Важливо враховувати, що лише наприкінці процесу проєктування є достатньо інформації для повного розуміння багатокористувацької комп'ютерної гри. На цих кроках приймаються дизайнерські рішення, які, зрештою, призведуть до розробки такої гри, яку можна розуміти цілісно [1]-[6].

При розробці програмного забезпечення багатокористувацької комп'ютерної гри важливо розуміти основи того, як організовується робота ігрового серверу. Для підтримки процесу розробки та функціонування програмного забезпечення багатокористувацької комп'ютерної гри можна використовувати два способи розгортання ігрових серверів [1]-[6].

Перший спосіб полягає у розгортанні ігрового сервера, розміщеного локальними гравцями [1]-[6].

Використання локальних ігрових серверів є простим способом реалізації багатокористувацької взаємодії в ігровій індустрії. За замовчуванням деякі гравці можуть проводити ігрові сеанси для кількох гравців (інші гравці можуть приєднуватися) на своєму локальному комп'ютері. На практиці розробники ігор дозволяють серверам запускати ігри на комп'ютерах гравців, розміщуючи багатокористувацькі ігри, і транслювати їх іншим гравцям, які повинні приєднатися до сервера, щоб гравці могли грати. Багатокористувацькі ігри можна налаштувати так, щоб гравці з усього світу могли приєднатися до інших користувачів [1]-[6].

Перевагами використання цього підходу до розгортання ігрового сервера для кількох гравців є такі:

- цей метод дає гравцям ексклюзивний контроль над грою та сеансами на сервері. Гравці можуть змінювати гру або розгортати власні сервери зі схвалення розробника [1]-[6];

– відсутність додаткових витрат на транзакцію: гравці розміщують сервер у своїй локальній системі (наприклад, на своєму комп'ютері), тому немає додаткових витрат, пов'язаних із запуском цих багатокористувацьких ігрових серверів [1]-[6];

– просте масштабування: сервери для багатокористувацьких ігор легко масштабуються, оскільки розробникам не потрібно контролювати, як і коли вони створюють більше серверів. Розробникам не слід займатися невеликою кількістю серверів, які можуть працювати для задоволення потреб гравців [1]-[6].

Недоліками використання локальних ігрових серверів є такі:

– слабкий контроль розробника: оскільки сервер контролює внутрішні системи гравця, розробникам важко контролювати діяльність гравця. Зазвичай такий підхід використовується для кооперативних ігор, але може бути придатним і для змагальних ігор;

– суттєва залежність від головного гравця, власника локального серверу: ігрова кімната працює на локальному комп'ютері певного гравця, тому, якщо йому потрібно від'єднати ігрову кімнату з будь-якої причини (відключення електроенергії, проблеми з підключенням до Інтернету тощо), ігровий сеанс завершується для всіх користувачів [1]-[6].

Другий спосіб полягає у використанні виділених або хмарних ігрових серверів. Такий підхід використання ігрового сервера поширений та часто використовується в змагальних іграх. Основною метою цього методу є розміщення ігрових серверів на віддалених серверах у хмарі. Наприклад, ігрові сервери можна розміщувати на хмарних платформах, і гравці можуть приєднуватися до ігор, розміщених і створених у цій хмарі. Ця модифікація зменшує навантаження на звичайних гравців та безпосередньо контролюється та налаштовується розробником гри [1]-[6].

Перевагами використання цього підходу до розгортання ігрового сервера є такі [1]-[6]:

– додатковий контроль ігрового процесу з боку розробників та адміністраторів ігри: цей підхід дозволяє розробникам безпосередньо бачити, як гравці взаємодіють із багатокористувацькими серверами. Це означає, що розробники можуть випускати оновлення, які впливають на всіх гравців одночасно, полегшуючи контроль динаміки змагальної гри, де чесність має першорядне значення [1]-[6];

– стабільний ігровий режим: оскільки сервери постійно працюють і розташовані десь у хмарі, гравці можуть підключатися до них і грати, коли забажають. Це підходить для ігор із розділеним світом, де розробники ігор хочуть створюють таке навколишнє середовище, де гравці можуть змінювати стан світу, а інші гравці можуть відчувати ці зміни за бажанням [1]-[6].

Недоліками використання цього підходу до розгортання ігрового сервера є такі [1]-[6]:

– додаткові вимоги до розробників: оскільки ігри розміщені на віддалених серверах, які гравці, як правило, не контролюють, важливо, щоб розробники постійно відстежували та контролювали свої ігрові сервери. Розробник повинен вирішити будь-яку проблему (затримка сервера, збій сервера тощо). Розробник також відповідає за керування масштабуванням сервера на основі потреб і поведінки гравців [1]-[6];

– додаткові витрати на роботу сервера. Ігрові сервери завжди доступні для всіх гравців, тому налаштування та експлуатація коштують невеликої комісії. Іноді окремі гравці розміщують власні сервери для своїх ігор, але витрати на такий підхід неминучі [1]-[6].

Визначено, що багатокористувацька комп'ютерна гра – це відеогра, в яку грають один або кілька гравців разом, та являє собою ігрове середовище, що забезпечує взаємодію між користувачами відповідно до певних правил, для функціонування багатокористувацьких ігор часто використовується доступ до мережі, що забезпечує підключення однієї системи до іншої, дозволяючи гравцям грати разом в одну гру. Розглянуто етапи процесу створення програмного забезпечення багатокористувацької гри. Проаналізовано базові

елементи, які необхідно визначити при створенні програмного забезпечення багатокористувацької комп'ютерної гри. Відзначено, що при розробці програмного забезпечення багатокористувацької комп'ютерної гри необхідно дотримуватися процесу, що розпочнеться з визначення мети та теми гри, після цього для кожної характеристики гри необхідно прийняти певні дизайнерські рішення, які повинні керуватися елементами, пов'язаними з характеристиками багатокористувацьких ігор [1]-[6]. Досліджено способи розгортання ігрових серверів для підтримки процесу розробки та функціонування програмного забезпечення багатокористувацької комп'ютерної гри.

1.2 Аналіз вимог до програмного забезпечення багатокористувацької гри. Дослідження програм-аналогів

Відповідно до технічного завдання необхідно розробити програмне забезпечення багатокористувацької гри з простими ігровими правилами, яка буде підтримувати одночасну взаємодію декількох гравців (користувачів) та надавати можливість вибору декількох персонажів.

Ігровий процес розроблюваної багатокористувацької відеогри полягає у виконанні завдання, пов'язаного з прямим знищенням супротивників.

Проаналізуємо існуючі подібні рішення [7]-[12]. Найближчим існуючим рішенням до проєкту є Chronoclysm [10] – багатокористувацька піксельна 2D гра з режимом, схожим за правилами до розроблюваної гри, зображена на рисунку 1.1. Гравці змагаються, створюючи армію воїнів часу раунд за раундом. Кожен із чотирьох класів має унікальні здібності. Механіку гри можна досліджувати за допомогою редактора рівнів, інтегрованого в майстерню Steam. Особливостями комп'ютерної гри Chronoclysm є: змішана локальна або онлайн-арена для чотирьох гравців, чотири класи гравців з різними здібностями, параметри налаштування ігрового процесу, понад 30 одиниць обладнання для покращення здібностей класів, вбудований редактор рівнів, кілька варіантів відображення персонажів для різних класів і команд,

усі мікротранзакції безкоштовні [10]. Із мінусів можна відзначити застарілу графіку та неможливість вибору різних персонажів.



Рисунок 1.1 – Комп'ютерна гра Chronoclysm [10]

Комп'ютерна гра Into the Breach (рис. 1.2) є багатокористувацькою піксельною грою з режимом, схожим за правилами до розроблюваної гри. Гравці повинні керувати потужними механізмами, щоб відбити інопланетну загрозу. Кожна спроба врятувати світ представляє нове випадково згенероване завдання в цій покроковій стратегічній грі. Особливостями комп'ютерної гри Chronoclysm є: різні віртуальні середовища, що генеруються автоматично при кожному запуску гри [11]. З недоліків комп'ютерної гри Into the Breach можна відзначити застарілу графіку та неможливість вибору різних персонажів [11].



Рисунок 1.2 – Комп’ютерна гра Into the Breach [11]

Ще однією програмою-аналогом є Darkest Dungeon [12] – багатокористувацька гра з покроковим режимом у пригодницькому стилі (рис. 1.3). Гравці повинні набирати, тренувати та керувати командою героїв для проходження різних рівнів через ліси, забуті лабіринти, зруйновані склепи тощо з метою повернення маєтку свого предка.

Гравці змагаються, створюючи армію воїнів, що мають унікальні здібності. Особливостями комп’ютерної гри Darkest Dungeon є: змішана арена для різних гравців, можливості налаштування ігрового процесу, варіанти відображення персонажів для різних класів і команд, усі мікротранзакції безкоштовні [12]. Із недоліків можна відзначити застарілу графіку та неможливість вибору різних персонажів.



Рисунок 1.3 – Комп’ютерна гра Darkest Dungeon [12]

Ще однією програмою-аналогом є Faster Than Light [13] – багатокористувацька гра з покроковим режимом у пригодницькому стилі, де користувачі керують космічним кораблем, який намагається врятувати галактику. Місія комп’ютерної гри Faster Than Light представляє унікальне випробування з кількома можливими рішеннями. Різні рівні комп’ютерної гри Faster Than Light являють собою випадково згенеровані галактики [13].

Комп’ютерна гра Faster Than Light (рис. 1.4) надає користувачеві можливість використання безкоштовних розширень, доповнень контенту (нову інопланетну расу, події, зброю, ігрові кораблі, дрони та інше), а також різних музичних треків. Іншою особливістю програмного забезпечення комп’ютерної гри Faster Than Light є надання безкоштовних автоматичних оновлень для всіх користувачів, хто придбає гру [13].

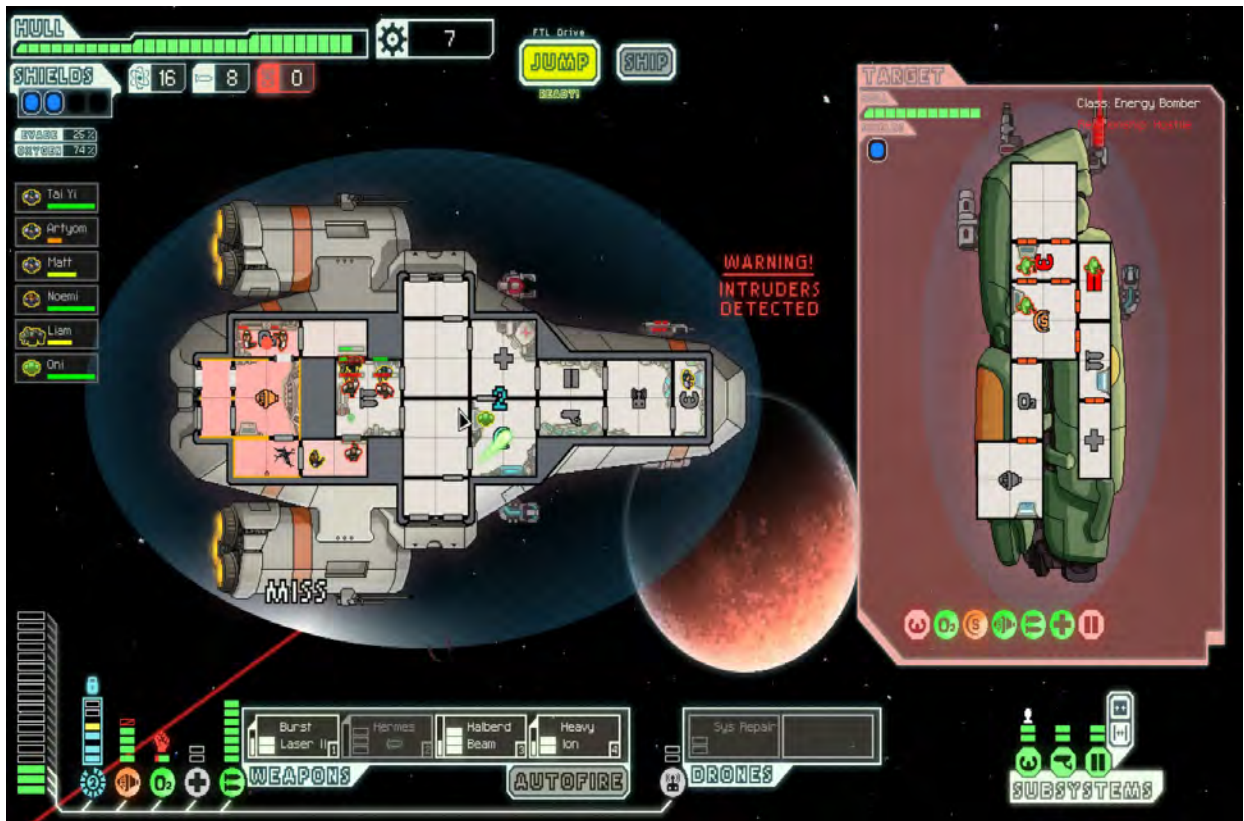


Рисунок 1.4 – Комп'ютерна гра Faster Than Light [13]

Особливостями комп'ютерної гри Faster Than Light є [13]:

- складний стратегічний ігровий процес (можливість надання наказів екіпажу, керування розподілом потужності корабля, вибору цілей зі зброєю);
- можливість встановлення власної швидкості гри (можливості зупинення гри під час бою, щоб оцінити свою стратегію та віддати накази);
- наявність засобів, що реалізують різні форми життя та технології (можливості покращення інвентарю);
- випадкова генерація рівнів. Кожне проходження надає різних ворогів, подій і можливостей для прийняття відповідних рішень [13].

З недоліків комп'ютерної гри Faster Than Light можна відзначити застарілу графіку та неможливість вибору різних персонажів [13].

Порівняльну характеристику програмного забезпечення багатокористувацьких ігор наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмного забезпечення багатокористувацьких ігор

Критерій порівняння	Chronoclysm [10]	Into the Breach [11]	Darkest Dungeon [12]	Faster Than Light [13]
Редактор рівнів	+	–	–	–
Можливість налаштування вигляду персонажів	+	–	+	–
Реалізація стратегії взаємодії між гравцями	+–	+	–	–
Випадкова генерація рівнів	–	–	+	+
Конкуренція між гравцями за перемогу	+	–	–	–
Безкоштовність	+	–	–	–

За результатами проведеного аналізу можна зробити висновок, що у наш час існує досить багато багатокористувацьких ігор. Проте деякі з них є досить складними, в інших реалізовано досить примітивну графіку. Суттєвим недоліком наявних комп'ютерних ігор, як правило, є необхідність оплати, що в деяких випадках робить неможливим їх використання серед деяких користувачів. Тому актуальною є розробка програмного забезпечення багатокористувацької гри.

При розробці програмного забезпечення багатокористувацької гри необхідно забезпечити такі функціональні вимоги:

- авторизація та реєстрація користувачів багатокористувацької гри;
- можливість створювати певну кількість персонажів;
- можливість підключення до ігрового сервера;
- засоби доступу до ігрового процесу, зокрема, можливості пересування персонажу, використання здібностей персонажу, ведення особистого рахунку та можливість переглядати таблицю лідерів;

- можливість розширеного доступу до ігрового процесу, зокрема, можливості пересування персонажу, перегляду таблиці лідерів, безсмертя персонажу (функція модератора);

- можливість блокування гравців на певний період;

- можливість редагування таблиці лідерів у разі нечесної гри з боку гравця, який потрапив у таблицю через свою нечесну гру.

Крім того, програмне забезпечення багатокористувацької гри повинно забезпечувати такі правила ігрового процесу:

- гравець повинен отримувати певну кількість балів за вбивство іншого гравця;

- гравець повинен втрачати деяку кількість балів при смерті свого персонажу;

- при смерті персонаж має можливість відродитися через деяку кількість секунд в зазначеному для відродження місці;

- місце для відродження повинно захищати гравців від отримання пошкоджень від інших персонажів.

Також програмне забезпечення багатокористувацької гри повинно забезпечувати механіки ігрового процесу, зокрема кожен персонаж повинен мати декілька здібностей для нанесення пошкоджень та одну здібність для швидкого пересування.

Межами програмного забезпечення багатокористувацької гри є те, що воно повинна базуватися на ігровому двигуні, та використовувати обрану для функціонування бази даних. Також гра повинна використовувати моделі та анімації, які знаходяться в загальному доступі та не вимагають авторських прав. Дані програмного забезпечення багатокористувацької гри повинні зберігатися у вигляді бази даних на серверній стороні застосунку, серверна частина застосунк повинна мати повний доступ до бази даних, частковий доступ для авторизації та реєстрації повинна мати клієнтська частина програмного застосунку.

1.3 Висновки за розділом 1

Визначено, що багатокористувацька комп'ютерна гра – це відеогра, в яку грають один або кілька гравців разом, та являє собою ігрове середовище, що забезпечує взаємодію між користувачами відповідно до певних правил, для функціонування багатокористувацьких ігор часто використовується доступ до мережі, що забезпечує підключення однієї системи до іншої, дозволяючи гравцям грати разом в одну гру. Розглянуто етапи процесу створення програмного забезпечення багатокористувацької гри. Проаналізовано базові елементи, які необхідно визначити при створенні програмного забезпечення багатокористувацької комп'ютерної гри. Відзначено, що при розробці програмного забезпечення багатокористувацької комп'ютерної гри необхідно дотримуватися процесу, що розпочнеться з визначення мети та теми гри, після цього для кожної характеристики гри необхідно прийняти певні дизайнерські рішення, які повинні керуватися елементами, пов'язаними з характеристиками багатокористувацьких ігор [1]-[6]. Досліджено способи розгортання ігрових серверів для підтримки процесу розробки та функціонування програмного забезпечення багатокористувацької комп'ютерної гри.

За результатами проведеного аналізу зроблено висновок про те, що у наш час існує досить багато багатокористувацьких ігор. Проте деякі з них є досить складними, в інших реалізовано досить примітивну графіку. Суттєвим недоліком наявних комп'ютерних ігор, як правило, є необхідність оплати, що в деяких випадках робить неможливим їх використання серед деяких користувачів. Тому актуальною є розробка програмного забезпечення багатокористувацької гри.

Сформульовано функціональні вимоги до програмного забезпечення багатокористувацької гри.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування та інших технологій розробки

При розробці програмного забезпечення багатокористувацької гри було використано мову програмування C++ [14], [15].

C++ є мовою програмування, яка часто використовується для створення програмного забезпечення різного призначення. Вона може бути використана у кількох моделях, таких як загальне програмне забезпечення, процедурне програмне забезпечення та об'єктно-орієнтоване програмне забезпечення. Це скомпільована мова, що означає, що вихідний код можна скомпілювати в об'єктний код або у двійкові файли для процесів [14], [15].

Мова C++ була створений у 80-х роках двадцятого століття, щоб надати більше функціональності мові C. Сумісність між двома мовами стає дедалі обмеженішою. Однак основним недоліком C++ є її високий рівень абстракції. На відміну від загальнодоступних мов, зокрема Python або Ruby, C++ вважається мовою низького рівня, близькою до машинних функцій, і тому користувачі її з меншою ймовірністю зрозуміють і отримають доступ до неї. З іншого боку, C++ надає користувачам такі переваги [14], [15]:

- це мова, що дозволяє більш ефективно створювати вископродуктивне програмне забезпечення, є однією з найпотужніших і найшвидших мов. Це пояснює конкурентний успіх застосунків для смартфонів [14], [15];

- висока популярність та широке використання, велика кількість інформації в Інтернеті на спеціальних семінарах та онлайн-курсах. Це не найпростіша мова, але початківцям її легко вивчити самостійно [14], [15];

- легка переносимість між різними операційними системами, зокрема, перенесення програмного забезпечення між Windows, Linux та Mac OS не потребує додаткових зусиль [14], [15];

- наявність стандартної бібліотеки, яка містить багато елементів зі стандартної бібліотеки шаблонів [14], [15].

Сьогодні Java і C# є часто використовуваними мовами, але C++ теж широко використовується. Велика кількість графічних програм, наприклад, створення програм для роботи з тривимірною графікою, зокрема, комп'ютерні ігри, створюються на C++. Крім того, за допомогою цієї мови програмування створюються важливі високопродуктивні фінансові програми, а також військове програмне забезпечення [14], [15].

Найпростішим способом роботи з цією мовою є використання інтегрованого середовища розробки Microsoft Visual Studio, спеціально розробленого Microsoft. Мова програмування C++ є компільованою мовою програмування, за допомогою компілятора код, написаний на C++, перетворюється на .exe, наприклад в операційній системі Windows [14], [15].

Деякі мови програмування вважаються інтерпретованими, тобто програми, розроблені за їх допомогу, компілюється під час виконання, а не перед виконанням, на відміну від C++, який забезпечує можливість програмувати в багатьох формах. Шаблони програмування – це спосіб кодування, що використовує певну парадигму. Існує багато моделей програмування, які пропонують різні способи мислення про написання програм. Відзначимо, що C++ дозволяє використовувати різні парадигми, забезпечуючи підтримку різних шаблонів мислення (зокрема, процедурне програмування, об'єктно-орієнтоване програмування, загальне програмування тощо), що допомагають розробникам на етапі проектування [14], [15].

Процедурне програмування є базовим підходом до створення програм на C або C++, однак об'єктно-орієнтоване програмування дозволяє абстрагувати код на окремі частини у вигляді класів, розміщуючи при цьому різні фрагменти коду в незалежних зв'язувальних блоках [14], [15]. З іншого боку, загальне програмування є досить складним, проте дозволяє працювати з даними без розрізнення типів даних [14], [15].

Обґрунтування вибору мови програмування для розробки програмного забезпечення багатокористувацької гри наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для розробки програмного забезпечення багатокористувацької гри

Критерій порівняння мов програмування	Мова програмування		
	C++	Python	Java
Низькорівневе програмування	+	–	+
Використання мови в ігрових движунах	+	–	+–
Засоби роботи з графікою	+	+–	+
Зручність для створення багатокористувацької гри	+	–	+
Швидкодія програмного забезпечення	+	–	+–

Отже, для реалізації програмного забезпечення багатокористувацької гри обрано мову програмування C++, яка є мовою програмування низького рівня, дозволяє створювати високопродуктивні програмні застосунки та має засоби для розробки програмного забезпечення комп'ютерних ігор.

2.2 Вибір ігрового движуна для створення програмного забезпечення багатокористувацької гри

В якості ігрового движуна для створення програмного забезпечення для багатокористувацької гри було обрано Unreal Engine 5 [16]–[18].

У порівнянні зі своїм конкурентом Unity Engine, Unreal Engine 5 фокусується на створенні 3D-ігор для різних пристроїв [16]–[18].

В Unreal Engine 4 як базова використовується мова програмування C++, що було обрано для створення програмного забезпечення багатокористувацької гри. Більшість движунів Unity використовують як базову мову програмування C# або JavaScript [16]–[18].

Unreal Engine має власний редактор візуальних сценаріїв Blueprint, який дозволяє створити прототип швидко та легко або створити цілу гру за допомогою схем. Ігровий движок також надає інструмент для створення ландшафту, який дозволяє створювати красиві пейзажі та додавати ігрові форми, рослини тощо [16]–[18].

Окрім візуальних скриптів, Unreal Engine має інструменти налагодження для моніторингу гри в реальному часі [16]–[18].

Розробники також можуть попередньо переглядати функції в режимі віртуальної реальності і режимі мобільного пристрою, що полегшує розробку продуктів для різних ігрових платформ [16]–[18].

Unreal Engine має редактор інструментів для редагування параметрів візуалізації, таких як різкість, шорсткість, текстура та текстурси [16]–[18].

Ігровий двигун Unreal Engine містить широкі інструменти штучного інтелекту для налаштування та налагодження неігрових персонажів [16]–[18].

Компонентом редактора Unreal Engine 4 є редактор UMG UI. Основу інтерфейсу користувача складають готові компоненти, такі як кнопки, прапорці та текстові поля. Ці готові компоненти можна редагувати за допомогою планувальника готових компонентів [16]–[18].

Якість графіки Unreal Engine дозволяє створювати високоякісні елементи, реалістичні відображення, м'які відтінки з трасуванням променів, динамічне загальне освітлення, тонке налаштування рухів, підтримку системи руху камери, можливість перегляду архітектури комп'ютерної гри в режимі реального часу та ін. [16]–[18].

Ще одна перевага Unreal Engine полягає в тому, що ігрові моделі легко імпортувати у форматі .fbx, легко завантажувати та легко накладати вміст [16]–[18].

Ігровий двигун Unreal Engine дозволяє адміністратору грати онлайн одночасно з гравцями (іншими учасниками) за допомогою технологій Windows Live, Xbox Live і GameSpy. Інструменти розробки офіційно не підтримують кілька клієнтів на одному сервері, але механізм

використовується для створення деяких ігор. Важливо, що багато відомих комп'ютерних ігор використовують движок Unreal Engine [16]–[18].

Усі компоненти ігрового движка представлені у вигляді об'єктів із наборами властивостей і наборами можливостей. У свою чергу, кожен клас є нащадком класу об'єктів. Актор є основним класом, який містить усі об'єкти, пов'язані з ігровим процесом і має просторові координати [16]–[18]. Пішак є фізичною моделлю гравців або об'єктів, якими керує штучний інтелект. Спосіб керування визначається конкретними об'єктами, і ці об'єкти називаються елементами керування. Інтелектуальний контроль визначає лише такі параметри, як здоров'я об'єкту (кількість шкоди після того, як пішак неактивний) або активна відстань до об'єкта під час гри [16]–[18].

Світ або рівень являє собою те, що відображає загальні характеристики простору, такі як гравітація і туман, які присутні у всіх гравців. Також можуть бути налаштування гри, наприклад режими гри, які встановлюють певні рівні [16]–[18].

Проста і зазвичай бінарна мозаїка використовується для кінцевих компонентів ігрового поля, таких як стіни. Всі простори діляться на повні і порожні. Всі предмети знаходяться в порожній частині простору і їх можна побачити тільки при малюванні плану. Не виключається можливість повного або часткового розміщення об'єкта в повній частині простору, але це може заважати операціям, де об'єкт намальований неправильно, наприклад, розрахунок фізичного тертя або напрямлення точки зору, якщо вона є. Кожна шахова фігура, що потрапляє прямо в заповнену зону, знищується [16]–[18].

Поверхні є ключовим елементом бінарного дерева в просторі. Ці елементи утворюються на перетині повної і порожньої частин простору. Групу елементів у просторовому бінарному дереві називають вузлом. Цей термін можна використовувати для позначення кількості вузлів, тобто кількості вузлів на екрані, або загального розміру гри. Кількість вузлів, які одночасно видно на екрані, впливає на обробку пейзажних зображень. Вузол видаляється, коли він зникає з екрана або повністю закривається іншим вузлом. Це

допомагає підвищити продуктивність, особливо в закритих середовищах. Поділ всього простору на групи вузлів називають зонами. Часто для цього використовують портали. Невидима поверхня, яка служить для поділу великого вузла на дві менші частини. Крім порталів, для визначення посівної площі використовують лічильні станції [16]–[18].

Визначення повної і порожньої частин простору визначається пензликом, який являє собою сукупність замкнених об'ємних об'єктів, що складаються з розігнутої поверхні. Ця теорія будівельного простору в ігровому движку Unreal Engine називається геометрією будівельних матеріалів. Геометрія може бути позитивною, де повний простір по суті порожній, або негативною, де простір заповнений матерією. Є три типи кистей [16]–[18]: безперервні (поділяються на позитивні та негативні), напівбезперервні та спорожнілі [16]–[18].

Суцільні кисті повністю оточені сегментами бінарного простору. Позитивні кисті заповнюють горизонтальний бінарний простір. Негативні кисті видаляють певний обсяг простору [16]–[18].

Напівжорстка кисть не впливає безпосередньо на просторове бінарне дерево, але впливає на його фізичну структуру, потрібно просто заповнити простір. Напівжорстка кисть служить для створення невидимих обмежень і зменшення кількості багатокутників і вузлів [16]–[18].

Порожня кисть призначена для створення лише частини бінарного дерева, вона не впливає на стан дерева. Порожні кисті в основному використовуються для створення об'ємів, які є частинами простору, які мають властивості, відмінні від властивостей ігрового світу. Ігровий світ завжди характеризується найменшою важливістю. Об'єми можуть змінювати силу тяжіння, в'язкість, туман та інші характеристики ігрового комп'ютерного світу [16]–[18].

Обґрунтування вибору ігрового движка для створення програмного забезпечення багатокористувацької гри наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору ігрового двигуна для створення програмного забезпечення багатокористувацької гри

Критерій порівняння ігрових двигунів	Ігровий двигун		
	Unreal Engine	Unity	CryEngine
Відкритість програмного коду	+	–	–
Використання мови програмування C++	+	–	+
Велика спільнота розробників	+	+	+–
Безкоштовність для індивідуальних розробників	+	+	–
Підтримувані платформи	Windows, MacOS, Linux, iOS, Android, Xbox, PlayStation	Windows, MacOS, Linux, iOS, Android, Xbox, PlayStation	Windows, MacOS, Linux, Xbox, PlayStation

Таким чином, для створення програмного забезпечення багатокористувацької гри обрано ігровий двигун Unreal Engine, що підтримує розробку на мові C++, яку обрано для створення комп'ютерної гри, є безкоштовним та дозволяє створювати ігри для великої кількості апаратних платформ.

2.3 Висновки за розділом 2

Для створення програмного забезпечення багатокористувацької гри обрано мову програмування C++, яка є мовою програмування низького рівня,

дозволяє створювати високопродуктивні програмні застосунки та має засоби для розробки програмного забезпечення комп'ютерних ігор.

В якості ігрового двигуна для створення програмного забезпечення багатокористувацької гри обрано Unreal Engine, що підтримує розробку на мові C++, яку обрано для створення комп'ютерної гри, є безкоштовним та дозволяє створювати ігри для великої кількості апаратних платформ.

3 ОПИС ПРОГРАМИ

3.1 Структура програми багатокористувацької гри

Структуру програмного забезпечення багатокористувацької гри подамо у вигляді діаграм в нотації UML (рис. 3.1 – 3.3).

Загальна діаграма класів програмного забезпечення багатокористувацької гри зображена на рис. 3.1.

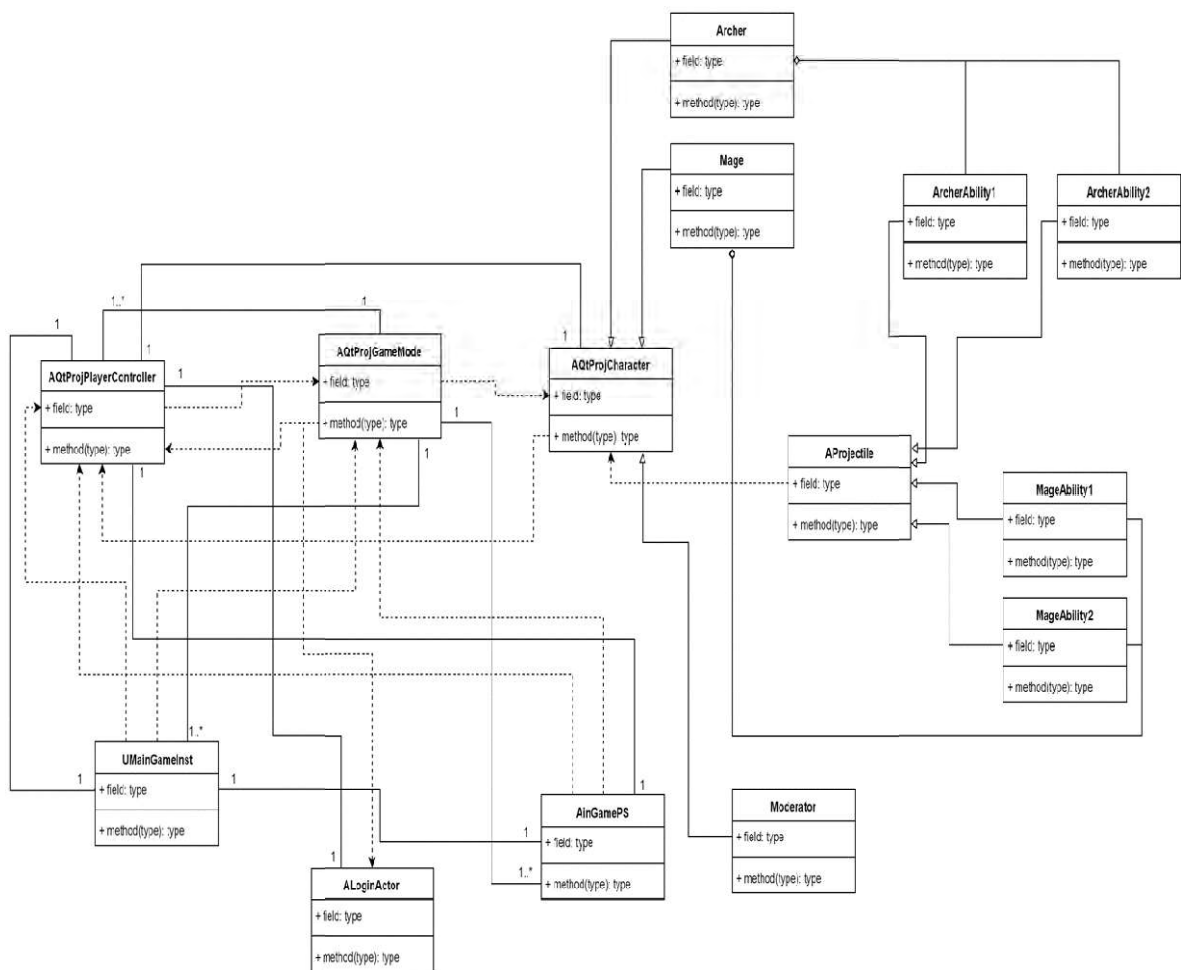


Рисунок 3.1 – Діаграма класів програмного забезпечення багатокористувацької гри у нотації UML

Як видно з діаграми, основними класами програми багатокористувацької гри є: ігровий режим (AQtProjGameMode), ігровий персонаж (AQtProjGameMode), контролер гравця (AQtProjPlayerController), здатності персонажу (ArcherAbility), ігрова сутність (UMainGaimInst) та ін.

Діаграма пакетів програмного забезпечення багатокористувацької гри зображена на рис. 3.2.

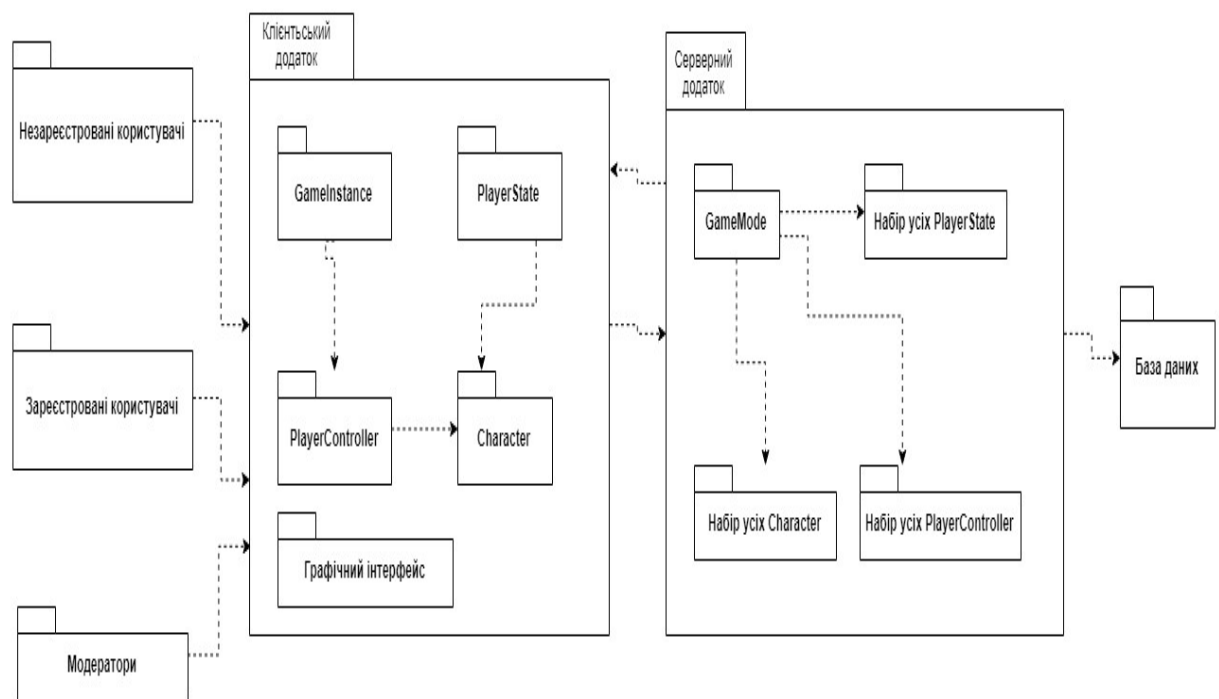


Рисунок 3.2 – Діаграма пакетів програмного забезпечення багатокористувацької гри у нотації UML

Програмне забезпечення багатокористувацької гри складається з пакетів, пов'язаних з клієнтським застосунком, а також з серверною частиною застосунку та базою даних.

Діаграма компонентів програмного забезпечення багатокористувацької гри зображена на рис. 3.3.

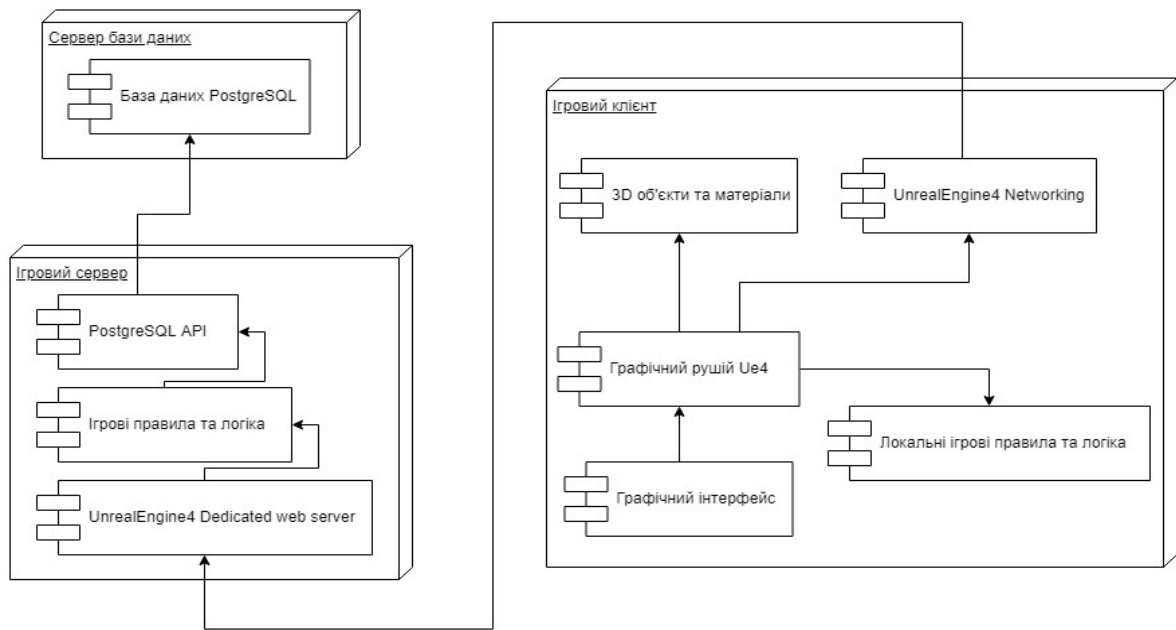


Рисунок 3.3 – Діаграма компонентів програмного забезпечення багатокористувацької гри у нотації UML

Як видно з рисунку, основними компонентами є ігровий сервер, сервер бази даних та ігровий клієнт.

3.2 Функціонування програмного забезпечення багатокористувацької гри

Загальну схему функціонування програмного забезпечення багатокористувацької гри подамо у вигляді діаграми прецедентів, зображеної на рис. 3.4. Як видно з рис. 3.4 в якості акторів програмного забезпечення багатокористувацької гри зареєстрованого гравця, адміністратора, незареєстрованого гравця та систему. Незареєстрований гравець може зареєструватися завдяки функціям системи, зареєстрований гравець може увійти до системи та грати в ігровому середовищі.

Рисунок 3.4 – Загальна схема функціонування програмного забезпечення багатокористувацької гри у вигляді діаграми прецедентів, поданої у нотатції UML

Діаграма діяльності програмного забезпечення багатокористувацької гри зображена на рис. 3.5.

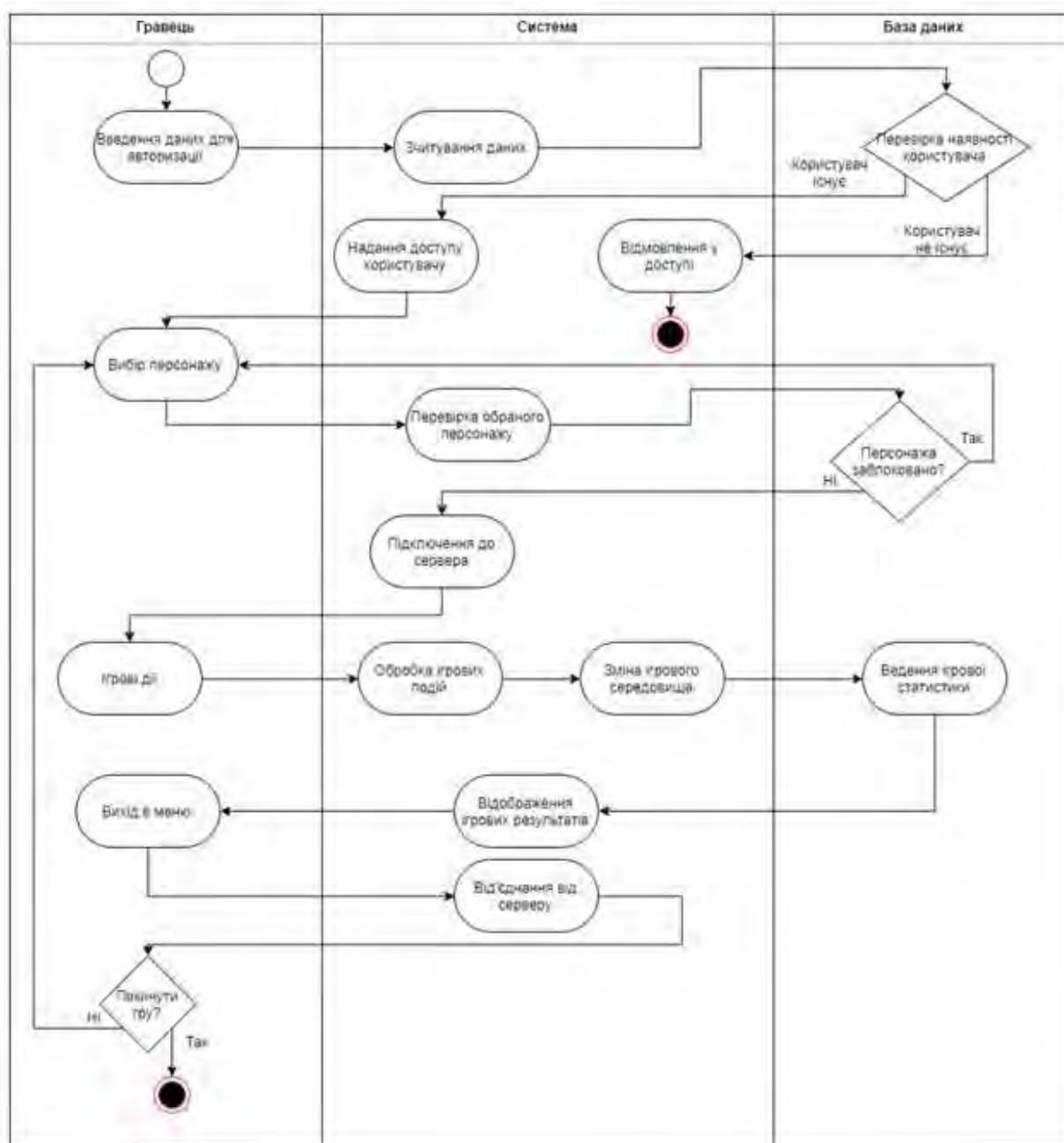


Рисунок 3.5 – Діаграма діяльності програмного забезпечення багатокористувацької гри у нотації UML

Діаграма станів програмного забезпечення багатокористувацької гри у нотації UML зображена на рис. 3.6.

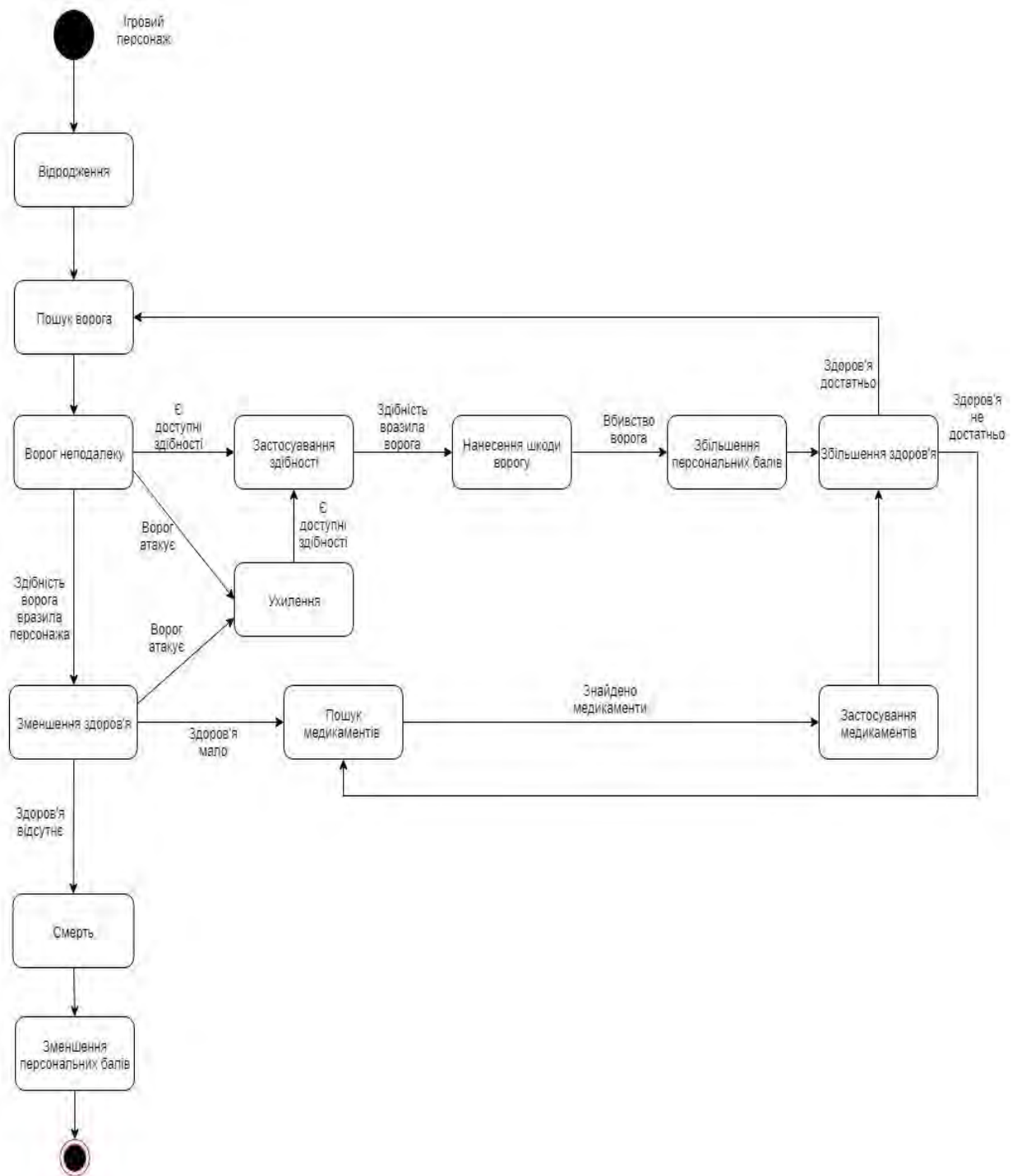


Рисунок 3.6 – Діаграма станів програмного забезпечення багатокористувацької гри у нотації UML

Логіка виконання дій для процесів програмного забезпечення багатокористувацької гри може бути описана за допомогою діаграми послідовностей (рис. 3.7).

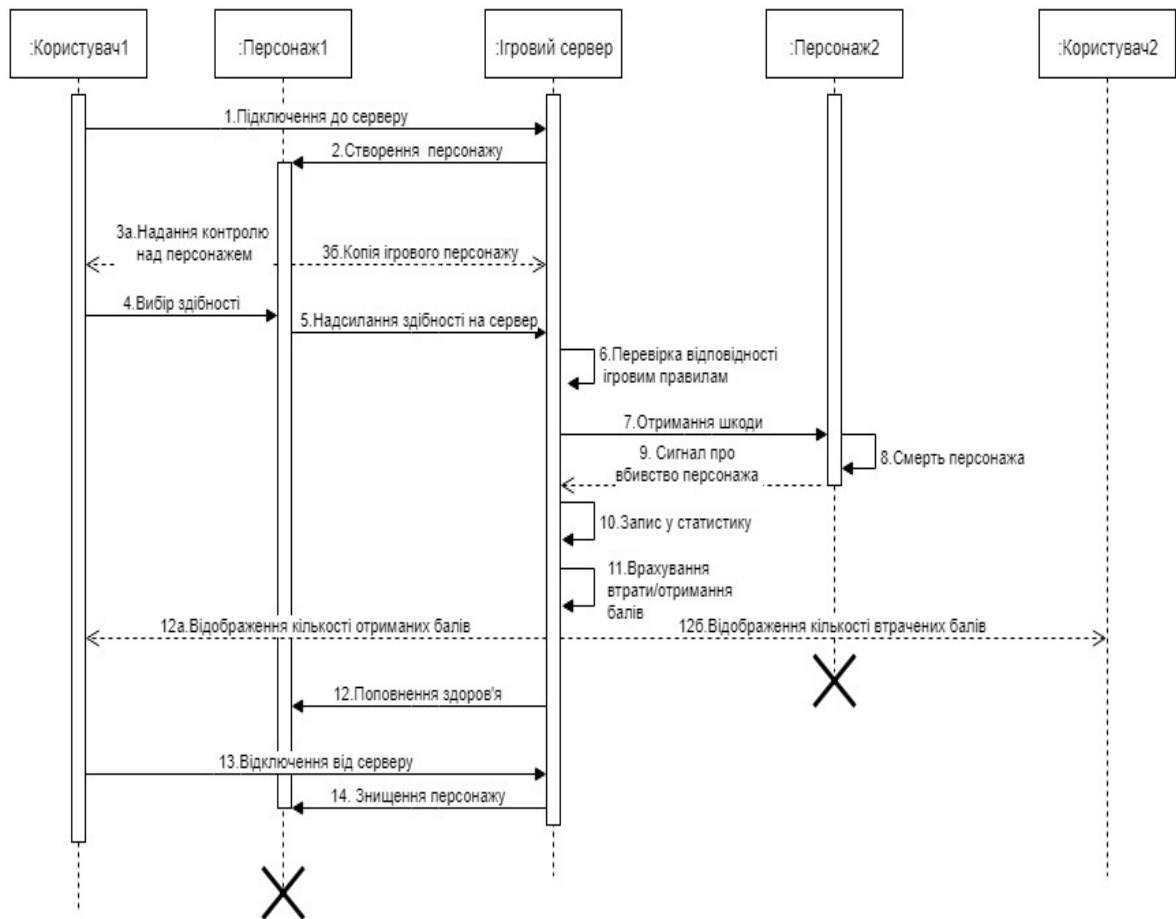


Рисунок 3.7 – Діаграма послідовностей програмного забезпечення багатокористувацької гри у нотації UML

3.3 Проектування інтерфейсу взаємодії користувача з програмним забезпеченням багатокористувацької гри

При розробленні програмного забезпечення багатокористувацької гри враховані функціональні вимоги до програми:

- авторизація та реєстрація користувачів багатокористувацької гри;
- можливість створювати певну кількість персонажів;
- можливість підключення до ігрового сервера;
- засоби доступу до ігрового процесу, зокрема, можливості пересування персонажу, використання здібностей персонажу, ведення особистого рахунку та можливість переглядати таблицю лідерів;

- можливість розширеного доступу до ігрового процесу, зокрема, можливості пересування персонажу, перегляду таблиці лідерів, безсмертя персонажу (функція модератора);
- можливість блокування гравців на певний період;
- можливість редагування таблиці лідерів у разі нечесної гри з боку гравця, який потрапив у таблицю через свою нечесну гру.

Після запуску програмного забезпечення багатокористувацької гри користувачу необхідно авторизуватися. Для авторизації користувачу багатокористувацької гри необхідно ввести логін та пароль. На рис. 3.8 зображено ескіз головної форми, де користувач має можливість авторизації.

Ескіз головної форми програмного забезпечення багатокористувацької гри. Інтерфейс складається з наступних елементів:

- У верхній частині: місце для логотипу (позначено як "ЛОГОТИП").
- У центрі: поля для введення "Логін" та "Пароль", кнопки "Увійти" та "Зареєструватися".
- У нижній правій частині: кнопка "Вийти".

Рисунок 3.8 – Ескіз головної форми програмного забезпечення багатокористувацької гри

Незареєстрований користувач повинен пройти процес реєстрації. Ескіз інтерфейсної форми для реєстрації наведено на рис. 3.9.

The screenshot shows a registration form with the following elements:

- A large rectangular box at the top labeled "Зображення" (Image).
- A vertical stack of four input fields: "Логін" (Login), "Пароль" (Password), "Пароль повторно" (Password again), and "Зареєструватися" (Register).
- A red button labeled "Повернутись назад" (Go back) below the registration button.
- A red button labeled "Вийти" (Logout) in the bottom right corner.

Рисунок 3.9 – Сторінка форми реєстрації користувача програмного забезпечення багатокористувацької гри

У процесі гри користувач може обирати персонажів. Ескіз форми вибору персонажу наведено на рис. 3.10.

The screenshot shows a character selection interface with the following elements:

- A vertical stack of three circular buttons, each containing a plus sign (+), labeled "Зображення персонажу" (Character image) and "Нікнейм персонажу" (Character nickname).
- A large rectangular box in the center labeled "Модель персонажу" (Character model).
- A red button labeled "Вийти" (Logout) in the top right corner.
- A box in the bottom right corner containing the text "Нікнейм" (Nickname), "Ігровий клас" (Game class), and "Кількість баллів" (Number of balls).
- A green button labeled "Розпочати гру" (Start game) below the box.

Рисунок 3.10 – Ескіз форми вибору персонажу програмного забезпечення багатокористувацької гри

У процесі гри користувач має можливість створити власного персонажу. Ескіз форми створення персонажу наведено на рис. 3.11.

Рисунок 3.11 – Ескіз форми створення персонажу програмного забезпечення багатокористувацької гри

Ескіз внутрішньо ігрового інтерфейсу програмного забезпечення багатокористувацької гри наведено на рис. 3.12.

Рисунок 3.12 – Ескіз внутрішньоігрового інтерфейсу програмного забезпечення багатокористувацької гри

3.4 Висновки за розділом 3

Розроблено програмне забезпечення багатокористувацької гри. Запропоновано структуру та описано основні модулі програмного забезпечення багатокористувацької гри. Визначено, що основними класами програми багатокористувацької гри є: ігровий режим (AQtProjGameMode), ігровий персонаж (AQtProjGameMode), контроллер гравця (AQtProjPlayerController), здатності персонажу (ArcherAbility), ігрова сутність (UMainGaimInst) та ін. Основними компонентами є ігровий сервер, сервер бази даних та ігровий клієнт.

Описано функціонування програмного забезпечення багатокористувацької гри.

Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням багатокористувацької гри.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Програма призначена для забезпечення та підтримки ігрового процесу багатокористувацької гри. Програма багатокористувацької гри написана на мові C++, яка є мовою програмування низького рівня, дозволяє створювати високопродуктивні програмні застосунки та має засоби для розробки програмного забезпечення комп'ютерних ігор. В якості ігрового двигуна для створення програмного забезпечення багатокористувацької гри обрано Unreal Engine, що підтримує розробку на мові C++, яку обрано для створення комп'ютерної гри, є безкоштовним та дозволяє створювати ігри для великої кількості апаратних платформ.

Програмне забезпечення багатокористувацької гри забезпечує виконання таких функцій:

- авторизація та реєстрація користувачів багатокористувацької гри;
- можливість створювати певну кількість персонажів;
- можливість підключення до ігрового сервера;
- засоби доступу до ігрового процесу, зокрема, можливості пересування персонажу, використання здібностей персонажу, ведення особистого рахунку та можливість переглядати таблицю лідерів;
- можливість розширеного доступу до ігрового процесу, зокрема, можливості пересування персонажу, перегляду таблиці лідерів, безсмертя персонажу (функція модератора);
- можливість блокування гравців на певний період;
- можливість редагування таблиці лідерів у разі нечесної гри з боку гравця, який потрапив у таблицю через свою нечесну гру.

Для експлуатації програмного забезпечення багатокористувацької гри необхідно таке апаратне та програмне забезпечення серверу:

- тактова частота процесора 2ГГц;

- вільне місце для серверної частини застосунку на жорсткому диску: 5Гб;
- оперативна пам'ять об'ємом 8Гб;
- підключення до Інтернету зі швидкістю 100Мбіт/с;
- платформа Windows 10;
- місце на жорсткому диску для бази даних – не менше 100 Гб.

Для експлуатації програмного забезпечення багатокористувацької гри необхідно таке апаратне та програмне забезпечення для клієнтської частини застосунку:

- тактова частота процесора – не менше 3ГГц;
- вільне місце для на жорсткому диску: 10Гб;
- оперативна пам'ять об'ємом 8Гб;
- підключення до Інтернету зі швидкістю 100Мбіт/с;
- платформа Windows 10;
- маніпулятор типу миша та клавіатура для введення даних.

Функціональні характеристики розробленого програмного забезпечення багатокористувацької гри наведено у технічному завданні (додаток А). Фрагмент тексту програмного забезпечення багатокористувацької гри наведено у додатку Б.

4.2 Характеристики програми та розгортання вебсерверу

Програмне забезпечення багатокористувацької гри може працювати безперервно, обслуговувати одночасно декілька користувачів-гравців.

Як вебсервер для функціонування програмного забезпечення багатокористувацької гри доцільно використовувати стек рішень Bitnami WAPP. Для розгортання вебсерверу спочатку необхідно завантажити його установочні файли, обравши в якості платформи Windows x64 за посиланням <https://bitnami.com/stack/wapp/installer>. Далі необхідно встановити

завантажений застосунок, після встановлення відкриється вікно керування серверами (рис. 4.1).

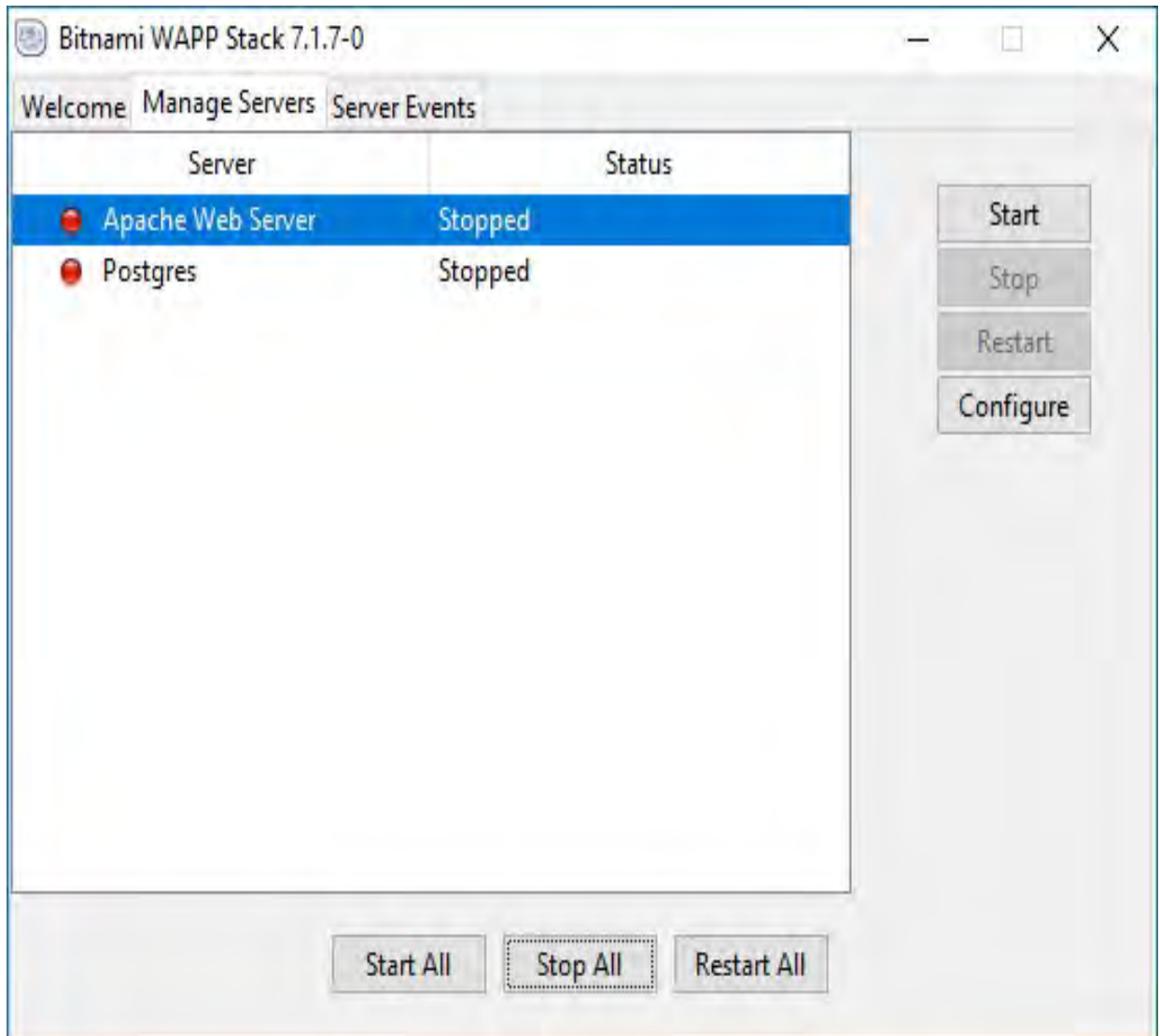


Рисунок 4.1 – Керування серверами Bitnami WAPP Stack

Ігровий сервер представляє собою програмний застосунок, для встановлення якого його необхідно скопіювати його на комп'ютер. Сервер необхідно запустити за допомогою ярлика з параметрами «-log». Доки додаток працює, до серверу можна підключитись. Для захисту застосунку, його клієнтська частина зможе підключитися лише до серверів, що знаходяться за визначеною IP-адресою.

4.3 Інструкція по експлуатації програми

4.3.1 Звернення до програми

Звертання користувачів до програмного забезпечення багатокористувацької гри здійснюється через клієнтську частину застосунку, яка запускається стандартними засобами операційної системи.

4.3.2 Вхідні й вихідні дані

Вхідними даними до програмного забезпечення багатокористувацької гри є інформація про користувача (логін та пароль), інформація про персонажа користувача (ім'я та інші параметри), команди щодо пересування персонажів.

Вихідними даними програмного забезпечення багатокористувацької гри є зміни координат персонажу користувача відповідно до його команд, повідомлення про успішну авторизацію чи реєстрацію, таблиця лідерів гри.

4.3.3 Повідомлення

Користувач програмного забезпечення багатокористувацької гри може отримати такі повідомлення: повідомлення про успішне з'єднання з ігровим сервером, повідомлення про закінчення гри та ін.

4.4 Виконання програмного забезпечення багатокористувацької гри

Після запуску програмного забезпечення багатокористувацької гри користувачу необхідно авторизуватися. Для авторизації користувачу багатокористувацької гри необхідно на головній формі програми ввести логін та пароль (рис. 4.2).

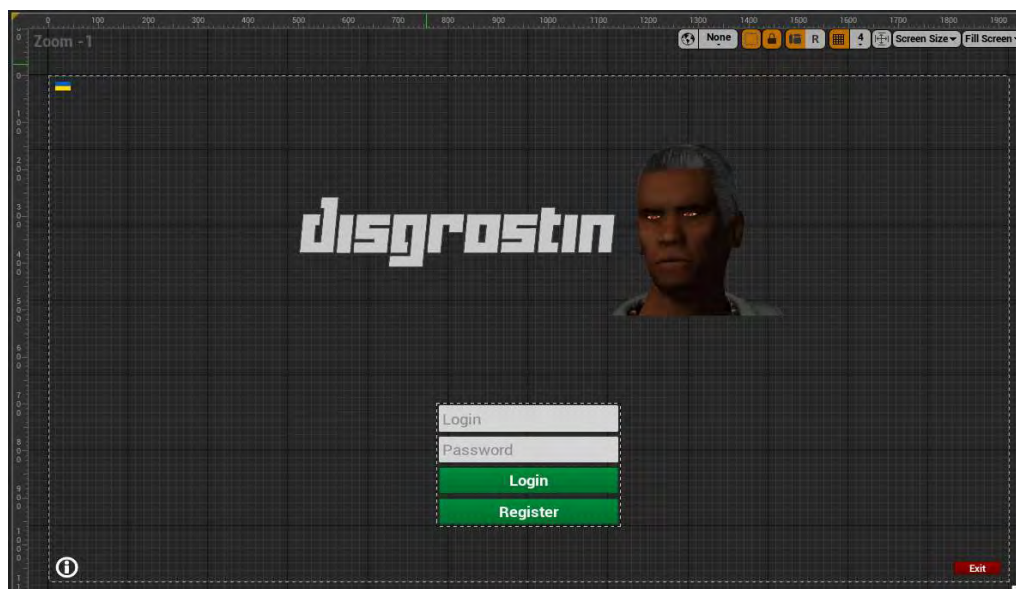


Рисунок 4.2 – Головна форма програмного забезпечення багатокористувацької гри

Незареєстрований користувач повинен пройти процес реєстрації. Інтерфейсна форма для реєстрації наведена на рис. 4.3.

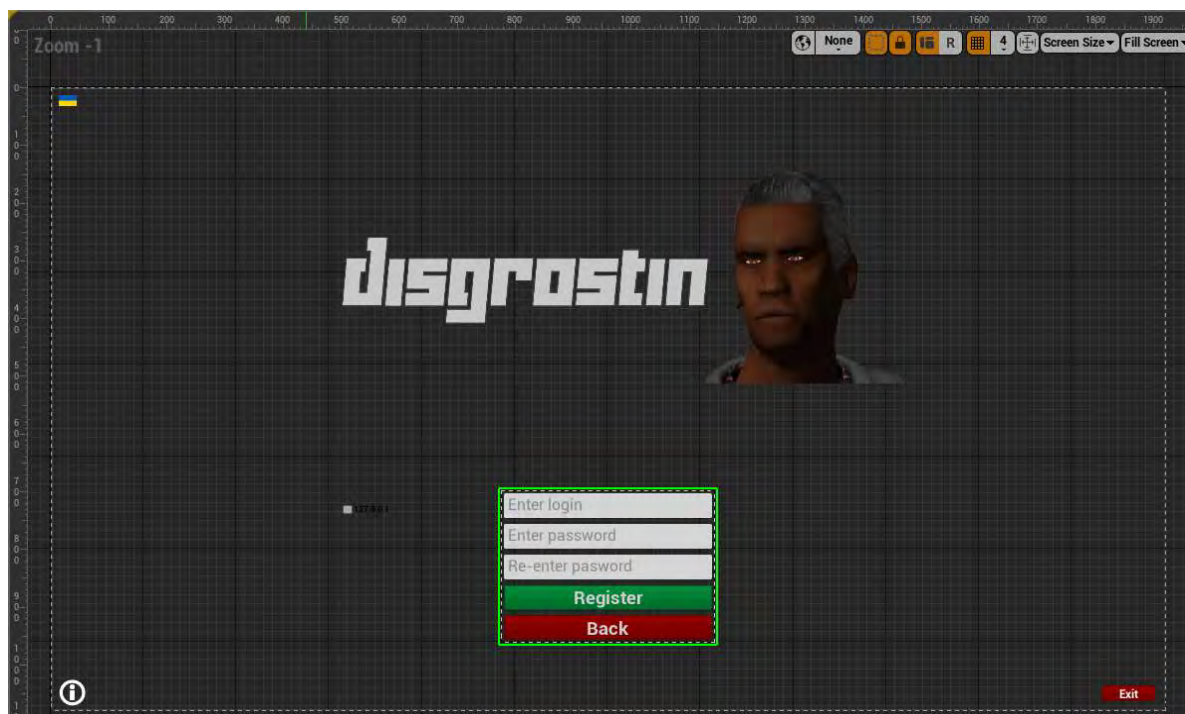


Рисунок 4.3 – Сторінка форми реєстрації користувача програмного забезпечення багатокористувацької гри

У процесі гри користувач має можливість обирати персонаж, яким він буде грати (рис. 4.4). Для цього користувачу необхідно зі списку обрати персонажу відповідно до його бажаних можливостей (сила атаки, здатність до захисту та ін.).

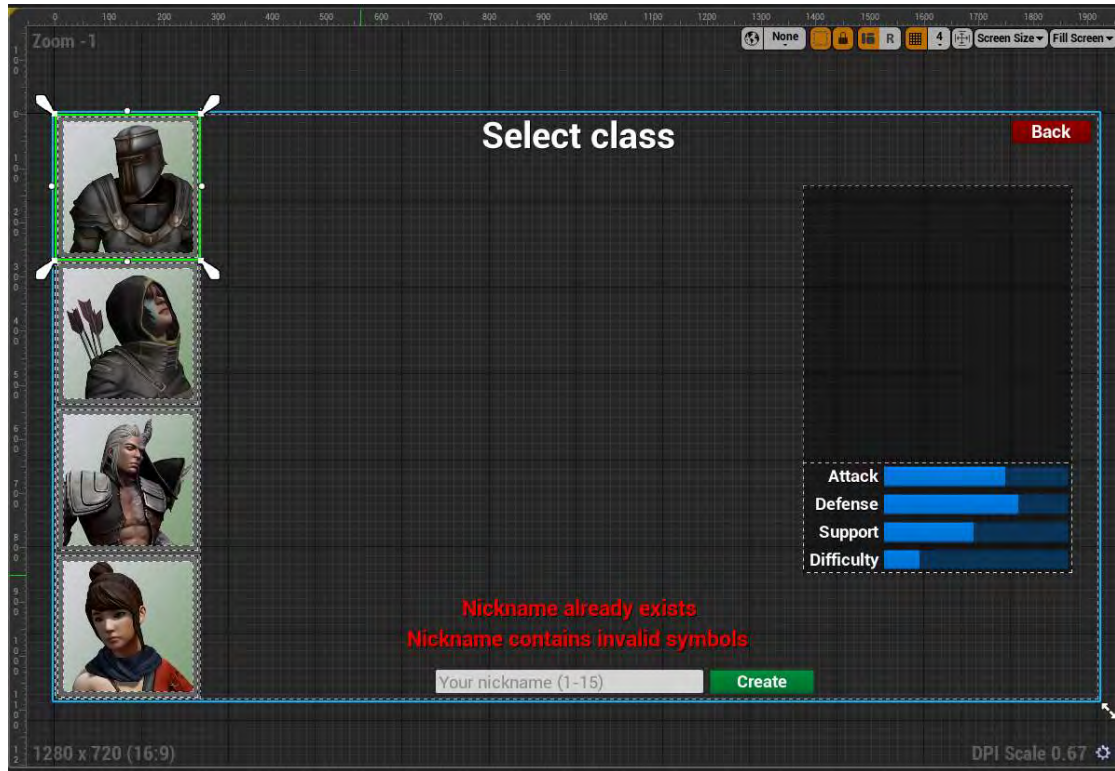


Рисунок 4.4 – Форма створення персонажу програмного забезпечення багатокористувацької гри

Після завершення ігрового процесу на екрані виводиться відповідне повідомлення (рис. 4.5).



Рисунок 4.5 – Повідомлення про завершення багатокористувацької гри

Виконано тестування швидкодії програмного забезпечення багатокористувацької гри за допомогою середовища UnrealEngine Editor. Результати тестування наведено на рис. 4.6.



Рисунок 4.6 – Результати тестування швидкодії багатокористувацької гри

На рис. 4.6 використовуються такі позначення:

- FPS(частота кадрів) – кількість кадрів за секунду;
- Frame – час на малювання одного кадру;
- Game – час на вирахування ігрових операцій, описаних на C++, та на Blueprints;
- Draw – час малювання одного кадру з боку процесора;
- GPU – час малювання одного кадру з боку відео карти.

За результатами тестування програмного забезпечення багатокористувацької гри отримано, що програмне забезпечення підтримує 89,46 кадрів за секунду, що більше мінімально прийнятного (60 кадрів за секунду), час малювання одного кадру з боку процесора (Draw) – 3,75 мілісекунди, тобто навантаження на процесор відносно невелике. Показники часу вирахування ігрових операцій, описаних на C++, та на Blueprints (Game), і часу малювання одного кадру з боку відеокарти (GPU) найдовші для малювання одного кадру складають 11,2 мілісекунди, тобто мають найбільший вплив на показник кадрів за секунду.

Результати тестування програмного забезпечення багатокористувацької гри можна вважати позитивними, оскільки отримані характеристики підтверджують зручність ігрового процесу для користувача, крім того підтверджено відповідність характеристик програмного забезпечення вимогам технічного завдання. Функціональні вимоги, описані у технічному завданні, реалізовано у розробленому програмному забезпеченні багатокористувацької гри.

4.5 Висновки за розділом 4

Описано програмне забезпечення багатокористувацької гри. Виконано тестування розробленого програмного забезпечення багатокористувацької гри. Результати тестування програмного забезпечення багатокористувацької гри можна вважати позитивними, оскільки отримані характеристики

підтверджують зручність ігрового процесу для користувача, крім того підтверджено відповідність характеристик програмного забезпечення вимогам технічного завдання.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено процес розробки комп'ютерних ігор.

Визначено, що багатокористувацька комп'ютерна гра – це відеогра, в яку грають один або кілька гравців разом, та являє собою ігрове середовище, що забезпечує взаємодію між користувачами відповідно до певних правил, для функціонування багатокористувацьких ігор часто використовується доступ до мережі, що забезпечує підключення однієї системи до іншої, дозволяючи гравцям грати разом в одну гру. Розглянуто етапи процесу створення програмного забезпечення багатокористувацької гри. Проаналізовано базові елементи, які необхідно визначити при створенні програмного забезпечення багатокористувацької комп'ютерної гри. Відзначено, що при розробці програмного забезпечення багатокористувацької комп'ютерної гри необхідно дотримуватися процесу, що розпочнеться з визначення мети та теми гри, після цього для кожної характеристики гри необхідно прийняти певні дизайнерські рішення, які повинні керуватися елементами, пов'язаними з характеристиками багатокористувацьких ігор. Досліджено способи розгортання ігрових серверів для підтримки процесу розробки та функціонування програмного забезпечення багатокористувацької комп'ютерної гри.

За результатами проведеного аналізу зроблено висновок про те, що у наш час існує досить багато багатокористувацьких ігор. Проте деякі з них є досить складними, в інших реалізовано досить примітивну графіку. Суттєвим недоліком наявних комп'ютерних ігор, як правило, є необхідність оплати, що в деяких випадках робить неможливим їх використання серед деяких користувачів. Тому актуальною є розробка програмного забезпечення багатокористувацької гри.

Сформульовано функціональні вимоги до програмного забезпечення багатокористувацької гри.

Для створення програмного забезпечення багатокористувацької гри обрано мову програмування C++, яка є мовою програмування низького рівня, дозволяє створювати високопродуктивні програмні застосунки та має засоби для розробки програмного забезпечення комп'ютерних ігор.

В якості ігрового двигуна для створення програмного забезпечення багатокористувацької гри обрано Unreal Engine, що підтримує розробку на мові C++, яку обрано для створення комп'ютерної гри, є безкоштовним та дозволяє створювати ігри для великої кількості апаратних платформ.

Розроблено програмне забезпечення багатокористувацької гри. Запропоновано структуру та описано основні модулі програмного забезпечення багатокористувацької гри. Визначено, що основними класами програми багатокористувацької гри є: ігровий режим (AQtProjGameMode), ігровий персонаж (AQtProjGameMode), контроллер гравця (AQtProjPlayerController), здатності персонажу (ArcherAbility), ігрова сутність (UMainGaimInst) та ін. Основними компонентами є ігровий сервер, сервер бази даних та ігровий клієнт.

Описано функціонування програмного забезпечення багатокористувацької гри.

Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням багатокористувацької гри.

Виконано тестування розробленого програмного забезпечення багатокористувацької гри. Результати тестування програмного забезпечення багатокористувацької гри можна вважати позитивними, оскільки отримані характеристики підтверджують зручність ігрового процесу для користувача, крім того підтверджено відповідність характеристик програмного забезпечення вимогам технічного завдання.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Intro to Multiplayer Games and How to Create Your Own [Electronic resource]. – Access mode : <https://www.gamedesigning.org/learn/multiplayer/>.
2. Multiplayer Server Basics. Creating a Multiplayer Game Server [Electronic resource]. – Access mode : <https://dev.to/ibmdeveloper/multiplayer-server-basics-ep-1-creating-a-multiplayer-game-server-5aed>.
3. A model to support the design of multiplayer games [Electronic resource]. – Access mode : <http://www.c5.cl/ieinvestiga/actas/tise99/html/papers/multiplayer/index.htm>.
4. Aspects of Networking in Multiplayer Computer Games [Electronic resource]. – Access mode : https://www.academia.edu/37767954/Aspects_of_Networking_in_Multiplayer_Computer_Games.
5. How I Built a Multiplayer Game on the Internet Computer : <https://medium.com/dfinity/how-i-built-a-multiplayer-reversi-game-on-the-internet-computer-f67d2fed0fc3>.
6. Co-Creative Media: Online Multiplayer Computer Game Culture [Electronic resource]. – Access mode : http://scan.net.au/scan/journal/display.php?journal_id=16.
7. The 25 best multiplayer games to play on PC [Electronic resource]. – Access mode : <https://www.rockpapershotgun.com/the-best-multiplayer-games-on-pc>.
8. The Best Multiplayer Video Games for 2023 [Electronic resource]. – Access mode : <https://www.pcmag.com/picks/the-best-multiplayer-video-games>.
9. 10 Best Classic 2D Games [Electronic resource]. – Access mode : <https://geekflare.com/best-classic-2d-games/>.
10. Chronoclysm [Electronic resource]. – Access mode : <https://store.steampowered.com/app/368690/Chronoclysm/>.
11. Into the Breach [Electronic resource]. – Access mode : https://store.steampowered.com/app/590380/Into_the_Breach/.

12. Darkest Dungeon [Electronic resource]. – Access mode : https://store.steampowered.com/app/262060/Darkest_Dungeon/.

13. Faster Than Light [Electronic resource]. – Access mode : https://store.steampowered.com/app/212680/FTL_Faster_Than_Light/.

14. C++ : présentation du langage de programmation [Electronic resource]. – Access mode : <https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1445316-c-definition-et-presentation-de-ce-langage-de-programmation/>.

15. Apprendre le C++ [Electronic resource]. – Access mode : <https://www.codingame.com/playgrounds/52875/apprendre-le-c/apprendre-le-c-2>.

16. Unreal Engine: The most powerful real-time 3D creation tool [Electronic resource]. – Access mode : <https://www.unrealengine.com/en-US>.

17. Rider for Unreal Engine [Electronic resource]. – Access mode : <https://www.jetbrains.com/idea/rider-unreal/>.

18. Unreal Engine [Electronic resource]. – Access mode : <https://developer.nvidia.com/game-engines/unreal-engine>.

ДОДАТОК А
Технічне завдання

Вступ

Програмне забезпечення багатокористувацької гри може використовуватися для підтримки ігрового процесу між декількома гравцями.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Програмне забезпечення багатокористувацької гри», затверджене наказом Національного університету «Запорізька політехніка» № 87 від 4 квітня 2023 р.

A.2 Призначення розробки

Програмний продукт призначений для забезпечення та підтримки ігрового процесу багатокористувацької гри.

A.3 Основні вимоги до програми, що розробляється

A.3.1 Вимоги до функціональних характеристик

Програмне забезпечення багатокористувацької гри забезпечує виконання таких функцій:

- авторизація та реєстрація користувачів багатокористувацької гри;
- можливість створювати певну кількість персонажів;
- можливість підключення до ігрового сервера;
- засоби доступу до ігрового процесу, зокрема, можливості пересування персонажу, використання здібностей персонажу, ведення особистого рахунку та можливість переглядати таблицю лідерів;

- можливість розширеного доступу до ігрового процесу, зокрема, можливості пересування персонажу, перегляду таблиці лідерів, безсмертя персонажу (функція модератора);
- можливість блокування гравців на певний період;
- можливість редагування таблиці лідерів у разі нечесної гри з боку гравця, який потрапив у таблицю через свою нечесну гру.

A.3.2 Вимоги до інтерфейсу програми

Інтерфейс програмного забезпечення багатокористувацької гри повинен бути зручним для користувачів.

A.3.3 Вимоги до надійності

Програмне забезпечення багатокористувацької гри повинно забезпечити надійне функціонування.

A.3.4 Умови експлуатації

Для експлуатації програмного забезпечення багатокористувацької гри необхідна наявність персонального комп'ютера у користувача.

A.3.5 Вимоги до складу та параметрів технічний засобів

Для експлуатації програмного забезпечення багатокористувацької гри необхідно таке апаратне та програмне забезпечення серверу:

- тактова частота процесора 2ГГц;
- вільне місце для серверної частини застосунку на жорсткому диску: 5Гб;

- оперативна пам'ять об'ємом 8Гб;
- підключення до Інтернету зі швидкістю 100Мбіт/с;
- платформа Windows 10;
- місце на жорсткому диску для бази даних – не менше 100 Гб.

Для експлуатації програмного забезпечення багатокористувацької гри необхідно таке апаратне та програмне забезпечення для клієнтської частини застосунку:

- тактова частота процесора – не менше 3ГГц;
- вільне місце для на жорсткому диску: 10Гб;
- оперативна пам'ять об'ємом 8Гб;
- підключення до Інтернету зі швидкістю 100Мбіт/с;
- платформа Windows 10;
- маніпулятор типу миша та клавіатура для введення даних.

A.3.6 Вимоги до маркування і пакування

Програма багатокористувацької гри може бути записана на будь-якому носії інформації. На пакуванні повинна бути назва програми – «Програмне забезпечення багатокористувацької гри».

A.4 Вхідні дані до роботи

Вхідними даними до програмного забезпечення багатокористувацької гри є інформація про користувача (логін та пароль), інформація про персонажа користувача (ім'я та інші параметри), команди щодо пересування персонажів.

Вихідними даними програмного забезпечення багатокористувацької гри є зміни координат персонажу користувача відповідно до його команд, повідомлення про успішну авторизацію чи реєстрацію, таблиця лідерів гри.

ДОДАТОК Б
Фрагмент тексту програми

```

void ALggnCtr::SetLggnCtrRtnn(int nmbr)
{
    switch (nmbr)
    {
        case 0:
        {
            SetCtrRtnn(Frtr(0.0f, -90.0f, 0.0f)); //00
            break;
        }
        case 1:
        {
            SetCtrRtnn(Frtr(0.0f, 0.0f, 0.0f)); //90
            break;
        }
        case 2:
        {
            SetCtrRtnn(Frtr(0.0f, 90.0f, 0.0f)); //-180
            break;
        }
        case 3:
        {
            SetCtrRtnn(Frtr(0.0f, 180.0f, 0.0f)); //-90
            break;
        }
    }
}

void ALggnCtr::SetLggnCtrLctn(int nmbr)
{
    switch (nmbr)
    {
        case 0:
        {
            SetCtrRtnn(Frtr(0.0f, -90.0f, 0.0f));

```

```

        SetCtrLctn(Rctrf(0.0f, 0.0f, 0.0f));
        break;
    }
    case 1:
    {
        SetCtrRtnn(Frtr(0.0f, 0.0f, 0.0f)); //90
        SetCtrLctn(Rctrf(-890.0f, 1150.0f, 0.0f));
        break;
    }
}

void MnnGmpmInst::SpnLggnCtr()
{
    AprjtPlrrlCntrlrt * plrrlCntrlrt = Cast<AprjtPlrrlCntrlrt>
>(UGmpmpllSttc::GetPlrrlCntrlrt(GetWorld(), 0));          if
(plrrlCntrlrt)
    {
        plrrlCntrlrt->toSpnLggnCtr(plrrlCntrlrt);
        plrrlCntrlrt-
>toSpnLggnCtr(UGmpmpllSttc::GetPlrrlCntrlrt(GetWorld(), 0));

    }
}

void MnnGmpmInst::SpnMnnmChrctr_Implementation()
{
    AprjtGmpmMode * GmpmMode = Cast<AprjtGmpmMode>
>(GetWorld()->GetAuthGmpmMode());
    AprjtPlrrlCntrlrt * plrrlCntrlrt = Cast<AprjtPlrrlCntrlrt>
>(UGmpmpllSttc::GetPlrrlCntrlrt(GetWorld(), 0));
    if (plrrlCntrlrt)
    {
        plrrlCntrlrt->toSpnMnnmChrctr(plrrlCntrlrt, GtchrID(),
GetUsrtrID());
    }
}

```

```

void MnnGmpmInst::ShcrSlctMnntd()
{
    if (wChrrsSlctMnntd)
    {

        MchrSlctMnntd      =      CreateWdgt<UUsrrtWdgt>(this,
wChrrsSlctMnntd);
        if (MchrSlctMnntd)
        {
            GEngine->Gddscrdbgmndr(-1,      5.f,      FColor::Red,
TEXT("MyLggnMnntd da"));

            MchrSlctMnntd->AddToViewport();
        }
    }
}

void MnnGmpmInst::ShddMnntd()
{
    if (wChrcrtMnntd)
    {

        GEngine->Gddscrdbgmndr(-1,      5.f,      FColor::Red,
TEXT("wLggnMnntd da"));
        MchrCreateMnntd      =      CreateWdgt<UUsrrtWdgt>(this,
wChrcrtMnntd);
        if (MchrCreateMnntd)
        {
            GEngine->Gddscrdbgmndr(-1,      5.f,      FColor::Red,
TEXT("MyLggnMnntd da"));
            MchrCreateMnntd->AddToViewport();
        }
    }
}

BssUsrrtWdgt * MnnGmpmInst::GtchrIconWdgt(int clmn)

```

```

{
    if (wChrcn)
    {
        Chrcn = CreateWdgt<BssUsrirtWdgt>(this, wChrcn);
        if (Chrcn)
        {
            Pgrst *rss;
            rss = PQexec(cnnt, TCHRRS_TO_ANSI(*cmmt));
            Chrcn-
>SetChrcn_chrrsID(FText::FromString(NSTTCHRRS(Pvltr(rss,      clmn,
0)))));//(i, clmn)
            Chrcn-
>SetChrcn_chrrsClass(FText::FromString(NSTTCHRRS(Pvltr(rss,  clmn,
1)))));
            Chrcn-
>SetChrcn_chrrsLevl(FText::FromString(NSTTCHRRS(Pvltr(rss,  clmn,
2)))));
            Chrcn-
>SetChrcn_chrrsExp(FText::FromString(NSTTCHRRS(Pvltr(rss,  clmn,
3)))));
            Chrcn-
>SetChrcn_chrrsHP_Max(FText::FromString(NSTTCHRRS(Pvltr(rss,
clmn, 4)))));
            Chrcn-
>SetChrcn_chrrsHP_Curr(FText::FromString(NSTTCHRRS(Pvltr(rss,
clmn, 5)))));
            Chrcn-
>SetChrcn_chrrsNmm(FText::FromString(NSTTCHRRS(Pvltr(rss,      clmn,
6)))));

            //GEngine->Gddscrdbgmndr(-1,      5.f,      FColor::Red,
TEXT("Set Chrcn da"));
            PQclear(rss);
            return Chrcn;
        }
    }
    else
    {

```

```

        Chrcn->Destruct();
        return NULL;
    }

    }else return NULL;
}

int MnnGmpmInst::GtchrCount()
{
    Pgrst *rss;
    if (!cnnt)
    {
        AprjtGmpmMode * GmpmMode = Cast<AprjtGmpmMode>
>(AprjtGmpmMode::StaticClass());
        cnnt = GmpmMode->GetCnnt();
    }

    if (cnnt)
    {
        FString cmmt = "SLCT * FROM \"Chrctrs\" WHERE usrرت_id =
'" + FString::FromInt(GetUsrرتID()) + "'";
        rss = PQexec(cnnt, TCHRRS_TO_ANSI(*cmmt));
        if (PQrsltStatus(rss) != PGRSS_TUPLES_OK)
        {
            PQclear(rss);
            return 0;
        }
        else
        {
            int rslt = PQntuples(rss);
            PQclear(rss);
            return rslt;
        }
    }
    else
    {
        return 0;
    }
}

```

```

    }
}

float MnnGmpmInst::GetMaxExp(int lev1)
{
    return (float)lev1 * 100;
}

bool MnnGmpmInst::ChkNcknmDB(FString ncknm)
{
    Pgrst *rss;

    if (cnnt)
        {FString cmmt = "SLCT chrctr_nmm FROM \"Chrctrs\" WHERE
chrctr_nmm = '\" + ncknm + '\"";
        rss = PQexec(cnnt, TCHRS_TO_ANSI(*cmmt));
        if (PQresultStatus(rss) != PGRSS_TUPLES_OK)
        {
            return false;
        }
        else
        {
            if (PQntuples(rss)>0)
            {
                return false;
            }
            else
            {
                return true;
            }
        }
        PQclear(rss);
    }
    else
        return false;
}

bool MnnGmpmInst::ChkNcknmChrress(FString ncknm)

```

```

{
    if (ncknmm.Contains(" ")
        || ncknmm.Contains("/")
        || ncknmm.Contains("\\")
        || ncknmm.Contains("@")
        || ncknmm.Contains("?"))
    {
        return false;
    }
    else return true;
}

bool MnnGmpmInst::DeleteChrctr(int chrtrs_id)
{
    Pgrst *rss;
    if (cnnt)
    {
        FString cmmt = "DELETE FROM \"Chrctrs\" WHERE chrctr_id
=      '"+ FString::FromInt(chrtrs_id)+"' AND      usrtr_id      =      '"+
FString::FromInt(GetUsrtrID()) + "'";
        rss = PQexec(cnnt, TCHRRS_TO_ANSI(*cmmt));
        if (PQrsslStatus(rss) == PGRSS_CMMTAND_OK)
        {

            return true;
        }
        else
        {
            return false;
        }
        PQclear(rss);
    }else return false;
}

bool MnnGmpmInst::CreateChrctr(FString chrtrs_ncknmm, FString
chrtrs_class)
{

```

```

Pgrst *rss;

AprjtGmpmMode * GmpmMode = Cast<AprjtGmpmMode>
>(AprjtGmpmMode::StaticClass());

FString clsd;
if (cnnt)
{
    FString cmmt = "SLCT clsd FROM \"Classes\" WHERE
class_nmm = '" + chrss_class + "'";
    rss = PQexec(cnnt, TCHRRS_TO_ANSI(*cmmt));
    if (PQrsslStatus(rss) != PGRSS_TUPLES_OK)
    {
        PQclear(rss);
        return false;
    }
    else
    {
        clsd = NSTTCHRRS(Pvltr(rss, 0, 0));
        PQclear(rss);
        rss = PQexec(cnnt, TCHRRS_TO_ANSI(*cmmt));
        if (PQrsslStatus(rss) == PGRSS_CMMTAND_OK)
        {
            return true;
        }
        else
        {
            return false;
        }
        PQclear(rss);
    }
}
}

```

```

void MnnGmpmInst::SetChrrsID(int id)
{
    ChrrsID = id;
}
int MnnGmpmInst::GetUsrtrID()
{
    return UsrtrID;
}
void MnnGmpmInst::SetUsrtrID(int id)
{
    UsrtrID = id;
}

void MnnGmpmInst::ShowPrtrqw(int party_id)
{
    if (wPrtrqwMnntd)
    {

        PrtrqwMnntd          =          CreateWdgt<UMyUsrtrWdgt>(this,
wPrtrqwMnntd);
        if (PrtrqwMnntd)
        {
            PrtrqwMnntd->Party_id = party_id;
            PrtrqwMnntd->AddToViewport();

        }
    }
}

```

ДОДАТОК В
Слайди презентації



**Міністерство освіти та науки України
Національний університет «Запорізька політехніка»
Кафедра програмних засобів**

Програмне забезпечення багатокористувацької гри

Виконав: Денис ЗОЛОТАРЕНКО

ст. групи КНТ-120сп

Керівник: Андрій ОЛІЙНИК

д.т.н., професор

Рисунок В.1 – Слайд 1



Об'єкт, предмет та мета роботи

- Об'єкт дослідження – процес розробки комп'ютерних ігор.
- Предмет дослідження – програмні засоби багатокористувацьких комп'ютерних ігор.
- Метою роботи є розробка програмного забезпечення багатокористувацької комп'ютерної гри.

Рисунок В.2 – Слайд 2

Завдання роботи

- Для досягнення поставленої мети у кваліфікаційній роботі бакалавра необхідно розв'язати такі задачі:
- виконати аналіз предметної області та програмних засобів підтримки багатокористувацьких ігор;
 - здійснити проєктування програмного забезпечення багатокористувацької гри;
 - створити програмне забезпечення багатокористувацької гри;
 - виконати тестування розробленого програмного забезпечення багатокористувацької гри.

Рисунок В.3 – Слайд 3

Порівняння існуючих аналогів

Критерій порівняння	Chronoclysm	Into the Breach	Darkest Dungeon	Faster Than Light
Редактор рівнів	+	–	–	–
Можливість налаштування вигляду персонажів	+	–	+	–
Реалізація стратегії взаємодії між гравцями	+–	+	–	–
Випадкова генерація рівнів	–	–	+	+
Конкуренція між гравцями за перемогу	+	–	–	–
Безкоштовність	+	–	–	–

Рисунок В.4 – Слайд 4

Постановка завдання

При розробці програмного забезпечення багатокористувацької гри необхідно забезпечити такі функціональні вимоги:

- авторизація та реєстрація користувачів багатокористувацької гри;
- можливість створювати певну кількість персонажів;
- можливість підключення до ігрового сервера;
- засоби доступу до ігрового процесу, зокрема, можливості пересування персонажу, використання здібностей персонажу, ведення особистого рахунку та можливість переглядати таблицю лідерів;
- можливість розширеного доступу до ігрового процесу, зокрема, можливості пересування персонажу, перегляду таблиці лідерів, безсмертя персонажу (функція модератора);
- можливість блокування гравців на певний період;
- можливість редагування таблиці лідерів у разі нечесної гри з боку гравця, який потрапив у таблицю через свою нечесну гру.

5

Рисунок В.5 – Слайд 5

Порівняння мов програмування

Критерій порівняння мов програмування	Мова програмування		
	C++	Python	Java
Низькорівневе програмування	+	–	+
Використання мови в ігрових двигунах	+	–	+–
Засоби роботи з графікою	+	+–	+
Зручність для створення багатокористувацької гри	+	–	+
Швидкодія програмного забезпечення	+	–	+–

6

Рисунок В.6 – Слайд 6

Порівняння ігрових двигунів

Критерій порівняння ігрових двигунів	Ігровий двигун		
	Unreal Engine	Unity	CryEngine
Відкритість програмного коду	+	—	—
Використання мови програмування C++	+	—	+
Велика спільнота розробників	+	+	+—
Безкоштовність для індивідуальних розробників	+	+	—
Підтримувані платформи	Windows, MacOS, Linux, iOS, Android, Xbox, PlayStation	Windows, MacOS, Linux, iOS, Android, Xbox, PlayStation	Windows, MacOS, Linux, Xbox, PlayStation

Рисунок В.7 – Слайд 7

Діаграма класів програмного забезпечення багатокористувацької гри

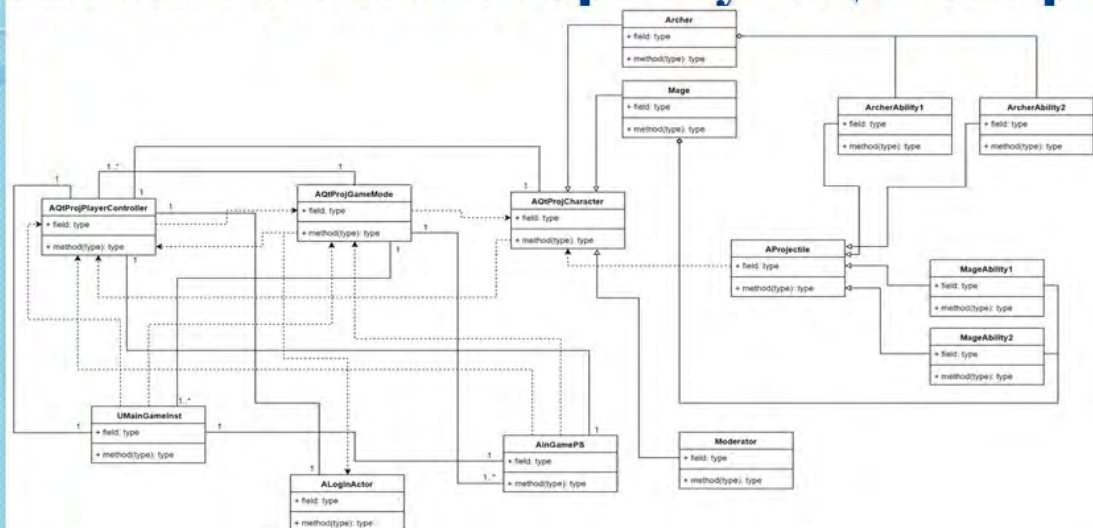


Рисунок В.8 – Слайд 8

Загальна схема функціонування ПЗ гри у вигляді діаграми прецедентів

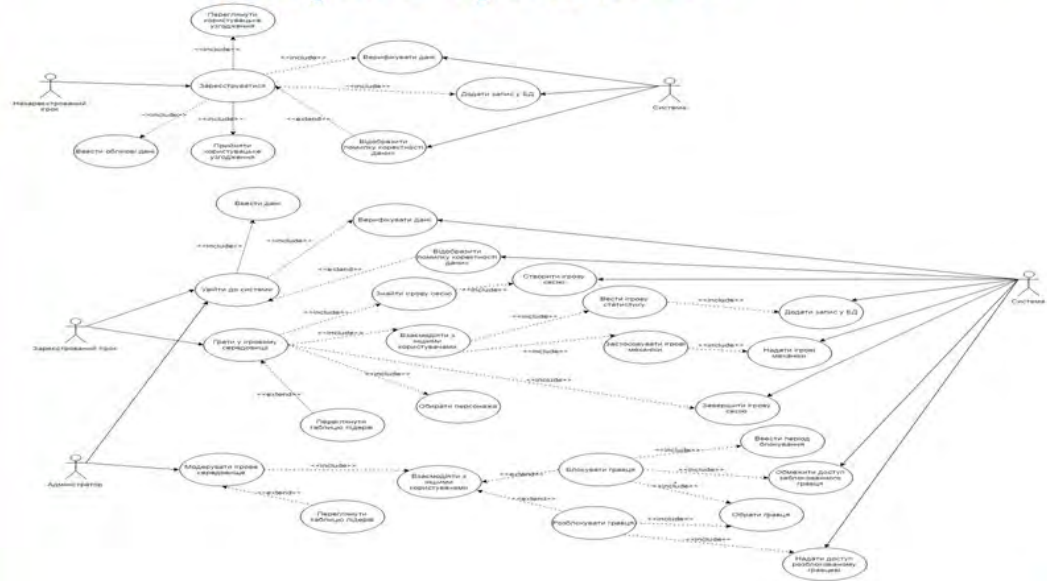


Рисунок В.9 – Слайд 9

Діаграма діяльності програмного забезпечення багатокористувацької гри

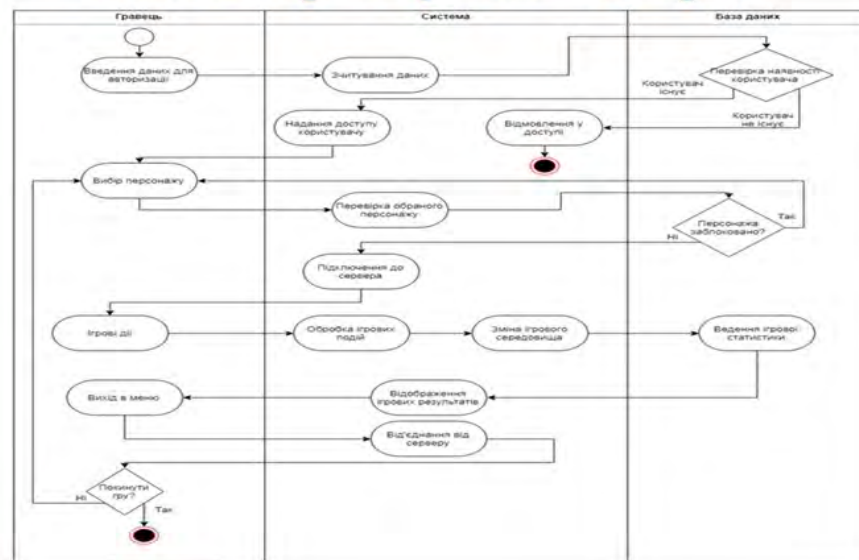


Рисунок В.10 – Слайд 10

Діаграма послідовностей програмного забезпечення багатокористувацької гри

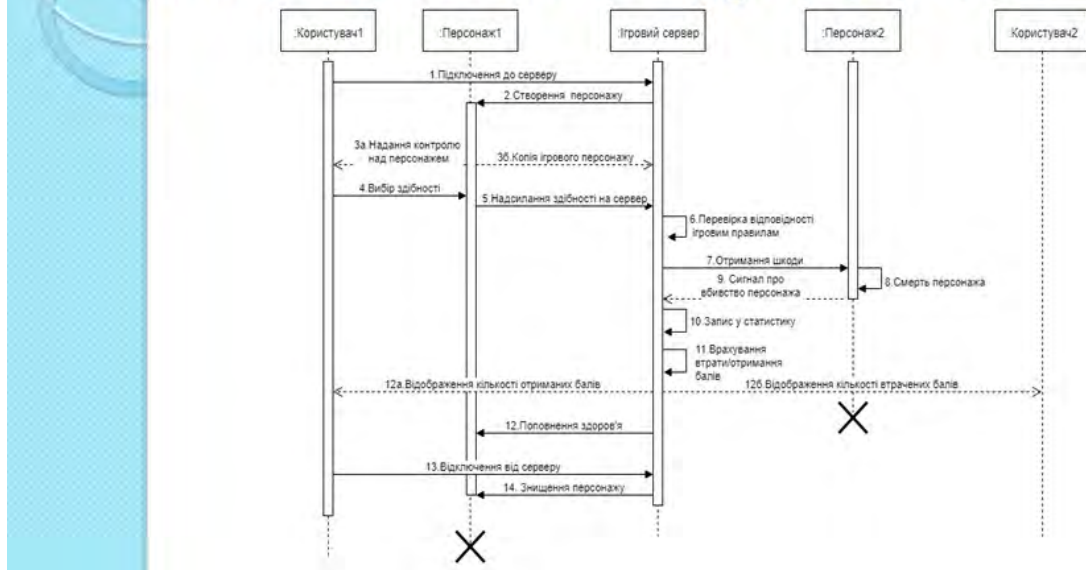


Рисунок В.11 – Слайд 11

Інтерфейс ПЗ багатокористувацької гри. Реєстрація та авторизація



Рисунок В.12 – Слайд 12

Інтерфейс ПЗ багатокористувацької гри. Створення персонажу. Результати тестування



13

Рисунок В.13 – Слайд 13

Висновки

В ході виконання дипломної кваліфікаційної роботи бакалавра було проаналізовано та досліджено процес розробки комп'ютерних ігор.

Сформульовано функціональні вимоги до програмного забезпечення багатокористувацької гри. Для створення програмного забезпечення багатокористувацької гри обрано мову програмування C++ та ігровий движок Unreal Engine. Запропоновано структуру та описано основні модулі програмного забезпечення багатокористувацької гри. Описано функціонування програми. Виконано проектування інтерфейсу взаємодії користувача з програмним забезпеченням багатокористувацької гри.

Виконано тестування розробленого програмного забезпечення багатокористувацької гри. Результати тестування програмного забезпечення багатокористувацької гри можна вважати позитивними, оскільки отримані характеристики підтверджують зручність ігрового процесу для користувача, крім того підтверджено відповідність характеристик програмного забезпечення вимогам технічного завдання.

14

Рисунок В.14 – Слайд 14