**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Національний університет кораблебудування імені адмірала Макарова**

Навчально-науковий інститут комп’ютерних наук та управління проєктами

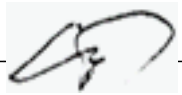
Кафедра програмного забезпечення автоматизованих систем

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма «Інженерія програмного забезпечення»

«Допущений до захисту»

Завідувач кафедри

_  __ проф. Приходько С.Б.

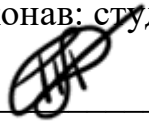
«___» _____ 2025 р.

***КВАЛІФІКАЦІЙНАРОБОТА***

**на здобуття ступеня вищої освіти «бакалавр»**

**на тему:** Розробка мобільної гри для видавництва «Crazy Games». (Development of a mobile game for the «Crazy Games» publishing studio.)

Виконав: студент групи 4157ст



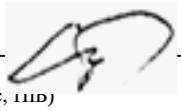
_____ Терлецький Д. М.

(підпис, ПІБ)

Керівник роботи:

_____

(посада, науковий ступень вчене звання)

____  __ Приходько С.Б.

(підпис, ПІБ)

Миколаїв – 2025 р.

**Національний університет кораблебудування імені адмірала Макарова**

Навчально-науковий інститут комп'ютерних наук та управління проєктами
Кафедра програмного забезпечення автоматизованих систем
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»
Гарант освітньої програми
_____ доц. Макарова Л.М.
(підпис)
«_08_» ___04___ 2025 р.

***ЗАВДАННЯ***
***НА КВАЛІФІКАЦІЙНУ РОБОТУ***
**на здобуття ступеня вищої освіти «бакалавр»**

Студенту Терлецького Дмитра Миколайовича
(Прізвище, ім'я, по батькові)

1. Тема роботи Розробка мобільної гри для видавництва «Crazy Games». (Development of a mobile game for the «Crazy Games» publishing studio.)
Керівник роботи Приходько Сергій Борисович
Затверджені наказом ректора №153 від 08.04.2025 р.
2. Термін подання роботи: 02.06.2025 р.
3. Вихідні дані по роботі: _____

4. Перелік питань, що належать до розробки (найменування розділів)
Титульний аркуш; Завдання на кваліфікаційну роботу; Анотація (українською та англійською мовами); Зміст; Перелік умовних позначень, символів, одиниць та термінів (при необхідності); Вступ; Аналіз практичної задачі інженерії програмного забезпечення; Проєкт програмного забезпечення; Результати розробки програмного забезпечення; Розділ з охорони праці; Висновки; Список використаних джерел; Додатки (технічне завдання, текст програми, опис програми, інструкція користувача, програма та методика випробувань програмного забезпечення).

5. Перелік презентаційних матеріалів 1.Титульний слайд; 2. Мета та завдання кваліфікаційної роботи; 3. Аналіз вимог для програмного забезпечення; 4. Діаграма варіантів використання; 5. Діаграма діяльності; 6. Концептуальна модель бази даних; 7. Ескізи на розробку; 8. Логічна модель даних; 9. Діаграма класів; 10. Діаграма послідовності; 11. Засоби розробки; 12. Результати розробки; 13. Висновки.

6. Консультанти розділів роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| 4 | Гурець Н.В., ст. викладач | | |
| | | | |
| | | | |
| | | | |
| | | | |

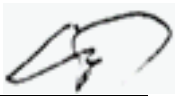
7. Дата видачі завдання 08.04.2025 р.

**К А Л Е Н Д А Р Н И Й  П Л А Н**

| Номер | Назва етапів роботи | Терміни виконання | Примітка |
|---|---|---|---|
| 1. | Підготовка розділу ВСТУП | 31.03.2025 | ПП* |
| 2. | Аналіз практичної задачі інженерії ПЗ | 07.04.2025 | ПП* |
| 3. | Аналіз вимог до ПЗ | 14.04.2025 | ПП* |
| 4. | Розробка архітектури ПЗ | 21.04.2025 | |
| 5. | Розробка проєкту ПЗ | 05.05.2025 | |
| 6. | Реалізація та тестування ПЗ | 12.05.2025 | |
| 7. | Підготовка розділу Результати розробки ПЗ | 16.05.2025 | |
| 8. | Підготовка розділу з охорони праці | 19.05.2025 | ПП* |
| 9. | Підготовка розділу ВИСНОВКИ | 23.05.2025 | |
| 10. | Оформлення списку використаних джерел та додатків | 26.05.2025 | |
| 11. | Подання на кафедру ПЗАС тексту остаточного варіанту роботи, підписаного її керівником, у роздрукованому та електронному форматі разом із заявами щодо самостійності виконання роботи та ідентичності друкованої та електронної версій роботи (Додатки 1 і 2 «Порядку здійснення заходів з перевірки робіт на наявність текстових збігів/ідентичності/схожості із використанням програмно-технічних засобів», який введений в дію наказом ректора НУК за №20 від 20.01.2020 р.) | 02.06.2025 | не пізніше, ніж за 14 діб до захисту (згідно п.4.1 зазначеного Порядку |
| 12. | Підготовка презентації та доповіді | 06.06.2025 | |
| 13. | Попередній захист роботи на засіданні кафедри ПЗАС | 09.06.2025 | |
| 14. | Подання на кафедру ПЗАС електронних версій наступних документів у форматі pdf: кваліфікаційної роботи; файлу-опису кваліфікаційної роботи (згідно Додатку до наказу ректора НУК за №287-уч від 19.05.2020 р.); презентації доповіді, а також Авторського договору з додатком | 16.06.2025 | |

* - за результатами переддипломної практики (ПП), яка була з 03.02.2025 до 23.03.2025 р.

**Студент** _____ _____
(підпис) (ПІБ)

**Керівник роботи**  _____
(підпис) (ПІБ)

# АННОТАЦІЯ

Кваліфікаційна робота присвячена розробці мобільної гри для видавництва «Crazy Games».

У кваліфікаційній роботі проведено аналіз предметної області, готових рішень, та методів створення мобільних ігор. Виконано постановку задачі на розробку продукту, який має на меті задовільнити потреби видавництва та слідувати представленому технічному завданню.

Також кваліфікаційна робота містить:

– Технічне завдання;

– Ескізний проєкт;

– Робочий проєкт;

– Методику та програму випробувань, а також результати виконання випробувань;

– Розділ з охорони праці;

– Опис програмного забезпечення.

Кваліфікаційну роботу виконано на 100 сторінках друкованого тексту, містить 17 рисунків, 43 таблиці, 5 додатків, та список використаних джерел із 4 найменувань.

ABSTRACT

The qualification work is devoted to the development of a mobile game for the publishing house “Crazy Games”.

The qualification work analyzes the subject area, off-the-shelf solutions, and methods for creating mobile games. The task of developing a product that aims to meet the needs of the publishing house and follow the presented terms of reference was completed.

The qualification work also contains:

- Terms of reference;
- Conceptual design;
- Working project;
- Test methodology and program, as well as test results;
- Section on labor protection;
- Software description.

The qualification work is performed on 100 pages of printed text, contains 17 figures, 43 tables, 5 appendices, and a list of references of 4 titles.

ЗМІСТ

ВСТУП ..... 7
1. АНАЛІЗ ПРАКТИЧНОЇ ЗАДАЧІ ІНЖЕНЕРІЇ ПРОГРМНОГО ЗАБЕЗПЕЧЕННЯ ТА ПОСТАНОВКА ЗАДАЧІ НА РОЗРОБКУ МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА «CRAZYGAMES» ..... 9
1.1 Аналіз практичної задачі інженерії програмного забезпечення гри для видавництва «CRAZYGAMES». ..... 9
1.2 Аналіз існуючих аналогів та обґрунтування доцільності розробки проєкту. ..... 10
1.2.1 Мобільна гра «Soul Knight» ..... 11
1.2.2 Мобільна гра «Survivor.io» ..... 13
1.2.3 Комп’ютерна гра «Enter the Gungeon» ..... 15
1.3 Постанова задачі на розробку мобільної гри для видавництва «Crazy Games». ..... 16
1.3.1 Вимоги до складу виконуваних функцій ..... 17
1.3.2 Вимоги до організації вхідних даних ..... 19
2 ПРОЄКТ МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА«CRAZYGAMES» ..... 21
2.1 Аналіз вимог до гри для видавництва «CrazyGames» ..... 21
2.1.1 Розробка моделі варіантів використання гри для видавництва «CrazyGames» ..... 21
2.1.2 Розробка специфікацій прецедентів гри для видавництва «CrazyGames» ..... 24
2.2 Розробка архітектури мобільної гри для видавництва «CrazyGames». ..... 32
2.3 Проєкт мобільної гри для видавництва «CrazyGames». ..... 35
2.3.1 Ескізний проєкт мобільної гри для видавництва «CrazyGames». ..... 35
2.3.2 Технічний проєкт мобільної гри для видавництва «CrazyGames». ..... 42
2.3.3 Робочий проєкт мобільної гри для видавництва «CrazyGames» ..... 53
3 РЕЗУЛЬТАТИ РОЗРОБКИ МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА «CRAZYGAMES» ..... 61
4 ОХОРОНА ПРАЦІ ..... 62
4.1 Огляд основних законодавчих та нормативних документів про охорону праці в Україні ..... 62
4.2 Опис структури управління питаннями охорони праці на підприємстві ..... 63
4.3Методи забезпечення техніки безпеки та охорони праці для конкретних посад ..... 64
4.4 Аналіз шкідливих та небезпечних факторів для конкретних посад або завдань ..... 66
4.5 Розробка заходів для зменшення дії шкідливих факторів ..... 67
ВИСНОВКИ ..... 70
СПИСОК ВИКОРИНСТИХ ДЖЕРЕЛ ..... 71
ДОДАРОК А – ТЕХНІЧНЕ ЗАВДАННЯ МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА

«CRAZYGAMES» ..... 72
ДОДАТОК Б – ОПИС МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА «CRAZYGAMES»..... 79
ДОДАТОК В – ПРОГРАМА ТА МЕТОДИКА ВИПРОБУВАНЬ МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА «CRAZYGAMES»..... 84
ДОДАТОК Г – ІНСТРУКЦІЯ КОРИСТУВАЧА МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА «CRAZYGAMES» ..... 87
ДОДАТОК Д – КОД МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА «CRAZYGAMES»..... 92

# ВСТУП

Ця кваліфікаційна робота присвячена вирішенню практичної задачі інженерії програмного забезпечення, що характеризується комплексністю на невизначеністю умов, зокрема розробці мобільної гри для «CrazyGames».

Комплексність та невизначеність умов полягають у тому, що вимоги та їх формулювання можуть бути не чіткими, в потребі розробити зручний користувацький інтерфейс, надання якісного ігрового досвіду користувачу та можливість легкого супроводу та підтримки гри.

Мобільні ігри є одним із найбільш популярних сегментів ігрової індустрії, що активно розвивається завдяки доступності мобільних пристроїв, завдяки цьому стає можливим проведення аналізу існуючих ігор схожих за жанром.

У результаті виконання аналізу готових ігор [1-3], було виявлено недоліки, а саме: висока складність на пізніх рівнях та відсутність кооперативу. Ці недоліки погано впливають на якість гри та бажання користувачів у неї повертатись. Тому було прийняте рішення розробити власну гру з урахуванням недоліків усіх аналогів.

Метою кваліфікаційної роботи є розв’язання практичної задачі інженерії програмного забезпечення, а саме розробка мобільної гри для «CrazyGames», який надаватиме користувачам якісний ігровий досвід за рахунок приємного інтерфейсу та різноманіттю ігрових механік, таких як:

– Випадкова генерація рівнів;

– Унікальність ворогів;

– Велика кількість випадкового озброєння та покращень.

Також гра передбачає можливість легкого супроводу, підтримки та оновлення для адміністраторів за рахунок зручного інтерфейсу програми.

Потрібно розв’язати такі завдання, щоб досягти мети:

– Провести аналіз практичної задачі інженерії програмного забезпечення ;

– Провести аналіз існуючих аналогів та обґрунтувати необхідність розробки гри;

– Виконати аналіз вимог;

– Розробити архітектуру мобільної гри;

– Розробити мобільну гру для видавництва «CrazyGames»;

– Розробити розділ охорони праці.

Об’єктом кваліфікаційної роботи є процес розробки мобільної гри для видавництва «CrazyGames».

У результаті вирішення цієї практичної задачі інженерії програмного забезпечення ми повинні отримати робочу мобільну гру для видавництва.

# 1. АНАЛІЗ ПРАКТИЧНОЇ ЗАДАЧІ ІНЖЕНЕРІЇ ПРОГРМНОГО ЗАБЕЗПЕЧЕННЯ ТА ПОСТАНОВКА ЗАДАЧІ НА РОЗРОБКУ МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА «CRAZYGAMES»

## 1.1 Аналіз практичної задачі інженерії програмного забезпечення гри для видавництва «CRAZYGAMES».

Мобільні ігри є одним із найпопулярніших сегментів індустрії розваг. Вони поєднують складні алгоритми, графічні технології, анімацію та механіки взаємодії з користувачем. Основне завдання розробника – створити не тільки цікаву, але й технічно оптимізовану гру, яка працюватиме на різних мобільних пристроях та відповідатиме вимогам ринку.

Розробка мобільної гри передбачає кілька ключових етапів:

Аналіз ринку та визначення цільової аудиторії – дослідження популярних жанрів, особливостей користувацького досвіду та конкурентних рішень.

Проєктування ігрового процесу – створення унікальних механік, які будуть залучати користувачів та підтримувати їхню активність.

Розробка програмного забезпечення - написання коду, впровадження ігрової логіки, графіки, звуку та анімації.

Тестування та оптимізація – виявлення помилок, поліпшення продуктивності, балансування складності гри.

Підтримка та оновлення – після релізу важливо оновлювати гру, додавати новий контент і виправляти можливі проблеми.

Одним із головних викликів є оптимізація продуктивності гри. Мобільні пристрої мають обмежені апаратні ресурси, тому необхідно ефективно використовувати пам’ять, графічні можливості та процесорний час. Оптимізація

включає зменшення навантаження на GPU, використання легких текстур та зниження затримки в обробці даних.

Також важливим аспектом є інтерфейс та управління. Оскільки мобільні ігри використовують сенсорний екран, управління має бути інтуїтивно зрозумілим, гнучким та адаптивним для різних розмірів дисплеїв.

Ще один важливий момент – масштабованість та підтримка гри. Успішні мобільні проєкти постійно оновлюються: додається новий контент, вдосконалюються механіки, оптимізується продуктивність. Тому при розробці необхідно передбачити можливість внесення змін та розширення функціоналу.

Окрему увагу варто приділити технологіям розробки. Сучасні мобільні ігри створюються за допомогою ігрових рушіїв (Unity, Unreal Engine), що дозволяють швидко реалізовувати графічні та механічні рішення. Також важливу роль відіграють хмарні сервіси, які використовуються для збереження ігрового прогресу та багатокористувацького режиму.

Таким чином, розробка мобільної гри є складною інженерною задачею, що вимагає ретельного аналізу, вибору оптимальних технологій, продуманого дизайну та ефективного тестування. Успішна реалізація мобільного проєкту залежить від правильної архітектури, балансу між продуктивністю та якістю графіки, а також постійної підтримки після виходу гри.

## 1.2 Аналіз існуючих аналогів та обґрунтування доцільності розробки проєкту.

Аналіз існуючих рішень це важливий етап у розробці будь-якого програмного забезпечення. У процесі такого аналізу здійснюється оцінка функціональних можливостей, аналіз ринку, а також виявлення переваг і недоліків кожного варіанту. Проведення детального аналізу сприяє прийняттю зваженого рішення та гарантує успішне впровадження обраного програмного забезпечення.

Для аналізу було обрано декілька схожих за тематикою ігор, та виявлено їх недоліки та переваги.

### 1.2.1 Мобільна гра «Soul Knight»

Розглянемо мобільну гру «Soul Knight» [1], наведену на рисунку 1.1


Рисунок 1.1 – Зразок ігрового процесу гри «Soul Knight»

«Soul Knight» - це динамічний рогалик-шутер з елементами RPG, який пропонує гравцям зануритися у світ хаотичних перестрілок, випадково згенерованих підземель та величезного вибору зброї та персонажів. Гра поєднує ретро-стиль, швидкий геймплей та унікальну систему прокачування, що робить її захоплюючою для всіх любителів жанру.

Нижче наведено докладний опис основних характеристик гри.

Переваги:

– Випадково створені рівні (Кожна гра пропонує унікальні рівні, що підвищує повторного проходження та підтримує інтерес гравців);

– Широкий вибір персонажів (У грі представлено багато героїв, кожен із яких має власні здібності та стиль гри);

– Різноманіття зброї (Гравці можуть використовувати сотні різних видів зброї);

– Кооперативний режим (Можливість грати в локальному багатокористувацькому режимі дозволяє проходити гру разом із друзями, що робить її ще цікавішою);

– Оригінальний візуальний стиль (Піксельна графіка у поєднанні з плавною анімацією створює приємну естетику гри);

– Безліч бонусів і навичок (Система поліпшень та випадкових бонусів робить кожен забіг унікальним).

Недоліки:

– Висока складність (Гра може бути досить складною, особливо для новачків, через велику кількість ворогів та швидкий темп);

– Високий вплив випадковості (Іноді отримані бонуси або зброя можуть бути неефективними, що впливає на проходження рівнів);

– Мікротранзакції (У грі є внутрішньоігрові покупки, що можуть надавати певні переваги та впливати на баланс).

«Soul Knight» - це захоплюючий рогалик, який пропонує швидкий і насичений геймплей, різноманіття зброї та персонажів. Гра чудово підходить для любителів динамічних шутерів, проте її випадковість і складність можуть стати викликом для деяких гравців.

### 1.2.2 Мобільна гра «Survivor.io»

Розглянемо другу схожу гру «Survivor.io» [2]. ЇЇ представлено на рисунку 1.2.


Рисунок 1.2 – Приклад ігрового процесу «Survivor.io»

«Survivor.io» – це динамічний екшен-рогалик у жанрі виживання, де гравець зіштовхується з ордами ворогів, покращує свої навички та збирає унікальні комбінації здібностей для проходження рівнів. Гра поєднує просте управління, глибоку систему розвитку персонажа та яскравий візуальний стиль, що робить її захоплюючою для широкої аудиторії.

Нижче наведено докладний аналіз основних характеристик гри.

Переваги:

- Захоплюючий ігровий процес (Гравець постійно перебуває у вирі подій, ухиляючись від ворогів та збираючи покращення);

- Просте управління (Гра використовує автоматичну атаку, що робить ігровий процес доступним навіть для новачків);
- Різноманіття здібностей (Велика кількість навичок та зброї дозволяє експериментувати з різними комбінаціями, створюючи унікальні стратегії проходження);
- Короткі, але динамічні рівні (Сесії тривають близько 15 хвилин, що робить гру ідеальною для швидких ігрових сесій);
- Система прогресу (Покращення персонажа між рівнями додає глибини та стимулює гравців повертатися до гри);
- Яскравий стиль (Гра має привабливу графіку, яка виглядає сучасно та легко сприймається).

Недоліки:

- Висока складність на пізніх рівнях (Без грамотної стратегії та належного рівня покращень вижити стає важко а іноді зовсім неможливо);
- Мікротранзакції (Гра містить внутрішньоігрові покупки, які можуть прискорювати прогрес, що може створювати дисбаланс між гравцями);
- Повторюваність (Хоча різні комбінації здібностей додають варіативності, загальна структура гри може здаватися повторюваною і згодом набридає).

Survivor.io – це чудовий вибір для любителів швидкого та динамічного виживання, який пропонує просте управління, глибоку систему покращень та яскравий стиль. Проте гра може здаватися повторюваною, а мікротранзакції можуть впливати на баланс.

### 1.2.3 Комп’ютерна гра «Enter the Gungeon»

Нижче, на рисунку 1.3, представлено комп’ютерну гру «Enter the Gungeon» [3]. Вона дуже схожа за жанром та стилем на те, що хотілося б бачити у результаті виконання проєкту.


Рисунок 1.3 – ігровий процес «Enter the Gungeon»

Enter the Gungeon – це захоплюючий рогалик-шутер, у якому гравці досліджують випадково згенеровані підземелля, наповнені ворогами, пастками та унікальною зброєю. Гра поєднує динамічний геймплей, гумор і глибоку систему розвитку, що робить її однією з найпопулярніших у своєму жанрі.

Нижче наведено докладний аналіз основних характеристик гри.

Переваги:

- Великий вибір зброї (У грі доступні сотні видів зброї);
- Динамічний та складний ігровий процес (Гра вимагає швидкої реакції, вміння ухилятися від куль та стратегічного підходу);
- Глибока система прогресу (Гравці можуть розблоковувати нових персонажів, поліпшення та зброю, що робить кожен забіг унікальним);

- Випадково створені рівні (кожне проходження пропонує нові випробування, ворогів і секрети, що підвищує інтерес до гри);
- Кооперативний режим (Гра підтримує локальний багатокористувацький режим, дозволяючи проходити рівні разом із другом).

Недоліки:

- Висока складність (Гра не прощає помилок, і новачкам може бути складно адаптуватися до швидкого темпу та численних ворогів);
- Відсутність онлайн-кооперативу (Грати з другом можна лише локально, що може бути незручним для деяких гравців).

Enter the Gungeon – це захоплююча гра, яка пропонує величезну кількість контенту, унікальну зброю та випробування. Проте гра може виявитися надто складною для новачків, а випадковість здобутих предметів іноді впливає на баланс гри.

## 1.3 Постанова задачі на розробку мобільної гри для видавництва «Crazy Games».

На основі проведеного аналізу існуючих рішень, їх переваг та недоліків, а також з урахуванням технічного завдання, було визначено основну мету проєкту - розробку відеогри в жанрі рогалик. Гра повинна поєднувати в собі креативний ігровий процес з високим рівнем реіграбельності, що є характерною рисою цього жанру. Особлива увага приділяється створенню зручного та інтуїтивно зрозумілого інтерфейсу користувача, який забезпечить легкий доступ до основних функцій гри та зробить керування персонажем максимально простим і логічним навіть для новачків.

Ключовим елементом естетики гри є графічне оформлення -вона буде виконана у піксельному стилі, який не лише надає грі унікальний візуальний вигляд, але й сприяє зниженню навантаження на систему, що дозволяє запускати

гру на широкому спектрі пристроїв. Крім того, передбачається використання 2D графіки з видом зверху, що є класичним підходом для ігор даного жанру. Це дозволить забезпечити повний огляд ігрового поля та стратегічне планування дій гравцем.

Фінальний продукт має бути не лише привабливим з точки зору геймплею та візуального оформлення, але й добре оптимізованим. Це означає ефективне використання ресурсів, швидке завантаження, стабільну роботу без збоїв і лагів. Крім цього, важливим аспектом є легкість супроводу та можливість подальших оновлень гри, що дозволить розширювати функціональність проєкту, впроваджувати нові елементи і відповідати очікуванням гравців на довгостроковій основі.

### 1.3.1 Вимоги до складу виконуваних функцій

Користувач

Користувач повинен мати доступ до функцій гри.

- Збереження прогресу (Гравець може зберігати прогрес у грі та продовжувати проходження з того місця, де зупинився);
- Доступ до повної версії гри (Користувач отримує доступ до всіх рівнів, персонажів і можливостей гри);
- Редагування профілю (Можливість змінювати ім'я, аватар);
- Система досягнень (Гравець може переглядати свої досягнення та нагороди);
- Взаємодія з іншими гравцями (Гра підтримує онлайн-режим за допомогою локального серверу, користувач може змагатися з друзями або грати разом);
- Вибір ігрового режиму (Гравець повинен мати змогу обирати потрібний йому ігровий режим);

- Взаємодія з ворогами (Гравець повинен мати змогу завдавати шкоди ворогам в процесі проходження гри);
- Взаємодія з предметами (Гравець повинен мати змогу піднімати, викидати, руйнувати та використовувати певні предмети на ігровому рівні);
- Купівля платного контенту (Доступ до магазину, де можна придбати ігрову валюту так прикраси за реальні гроші);
- Оновлення та події (Отримання сповіщень про нові оновлення, ігрові події та акції).

Адміністратор (розробник гри)

Адміністратор – це особа або група осіб, які мають доступ до внутрішніх налаштувань гри та керують її функціоналом.

Функції для адміністратора:

- Управління користувачами (Адміністратор може блокувати або розблоковувати акаунти, переглядати звіти про порушення правил гри);
- Оновлення контенту (Додавання нових рівнів, персонажів, предметів, балансування ігрового процесу);
- Аналіз статистики (Перегляд активності користувачів, їхніх досягнень, популярності рівнів і загальної ефективності гри);
- Керування внутрішньоігровими покупками (Налаштування цін, знижок, акцій та перевірка коректної роботи платіжних систем);
- Моніторинг продуктивності гри (Контроль за відсутністю технічних збоїв, виправлення помилок та забезпечення стабільної роботи).

### 1.3.2 Вимоги до організації вхідних даних

Вхідні дані

Вхідні дані -це інформація, яку система отримує від користувача або адміністратора для забезпечення ігрового процесу, взаємодії з сервером та внутрішньоігрових операцій.

Для користувача:

- Збереження прогресу (ID користувача, Поточний рівень, Досягнення, Статистика, Збережені налаштування);
- Редагування профілю (Ім’я користувача, Аватар, Налаштування звуку та управління);
- Запит на отримання списку досягнень (ID користувача);
- Взаємодія з іншими гравцями (Запити на додавання друзів, Запити на спільну гру, Статус мережевого з’єднання);
- Купівля платного контенту (ID користувача, ID товару, Сума покупки, Платіжна інформація);
- Отримання оновлень та повідомлень (ID користувача, Час останнього входу, Налаштування сповіщень).

Для адміністратора:

- Управління користувачами (ID користувача, Статус акаунта, Скарги, Дії модератора);
- Додавання та редагування контенту (Назва рівня, Опис, Графічні ресурси, Складність, Нагороди);
- Аналіз статистики (Кількість активних гравців, Час у грі, Популярність рівнів, Кількість покупок);
- Налаштування внутрішньоігрових покупок (Назва товару, Опис, Ціна, Доступність, Акційні пропозиції);
- Моніторинг продуктивності (Звіти помилок, Час завантаження рівнів, Відгуки користувачів).

Вихідні дані

Вихідні дані -це інформація, яка формується та надається користувачам або адміністраторам ігровим інтерфейсом на основі отриманих вхідних даних.

Для користувача:

- Ігровий інтерфейс та статус (Поточний рівень, Доступні персонажі, поточний баланс, Список завдань)
- Система досягнень (Перелік досягнень, Отримані нагороди, Прогрес до наступного рівня)
- Статус купленого контенту (Доступність куплених предметів, Залишок ігрової валюти)
- Оновлення та події (Оновлення контенту, Акції, Тимчасові події)

Для адміністратора:

- Список друзів (ID користувача ,Ім’я користувача, Статус акаунта, Дата останнього входу)
- Статистика гри (Кількість активних гравців, Популярність режимів, Найчастіші покупки)
- Звіти про помилки (Тип помилки, Лог-файли, Дата та час збою)
- Фінансові звіти (Загальний дохід, Найпопулярніші покупки,)

# 2 ПРОЄКТ МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА«CRAZYGAMES»

Після виконання аналізу вимог до програмного забезпечення було розроблено ескізний проєкт гри для видавництва. Для його створення використовувалась мова моделювання UML. Для оцінки вимог та їх формулювання створено діаграму варіантів використання, також специфікація до цих варіантів. Для варіантів використання також створено діаграми діяльності. Проєкт також містить інформаційну модель та приклади користувацького інтерфейсу.

## 2.1 Аналіз вимог до гри для видавництва «CrazyGames»

### 2.1.1 Розробка моделі варіантів використання гри для видавництва «CrazyGames»

Діаграма варіантів використання UML – це візуальне представлення, що відображає можливі шляхи або варіанти використання програмного забезпечення з точки зору користувача. Вона дає можливість зрозуміти як користувач взаємодіятиме з продуктом та які функції матиме програмне забезпечення.

Розробку діаграми можна виконати у 4 етапи: виявлення та опис можливих акторів, виявлення та опис варіантів використання, побудова самої діаграми та створення специфікації для прецедентів.

Після дослідження предметної галузі було виявлено 2 групи акторів, а саме адміністратор та користувач. На рисунку 2.1 зображено акторів мобільної гри.

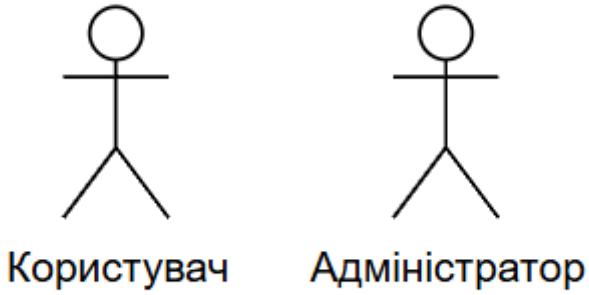



Рисунок 2.1 – Актори ПЗ

Короткий опис акторів наведено в таблиці 2.1.

Таблиця 2.1 – Опис виявлених акторів

| Актор | Короткий опис |
|---|---|
| Адміністратор | Користувач, який має доступ до редагування функцій гри та керує її процесами. Також він може формувти звіти що до продажів чи помилок у грі та займатися адмініструванням мобільної гри. |
| Користувач | Користувач, який має доступ до основних функцій гри, від може взаємодіяти з грою за допомогою користувцікого інтерфейсу та здійснювати купівлю внутрішньоігрового контенту. |

Виявлені прецеденти описані у таблиці 2.2.

Таблиця 2.2 – Виявлені прецеденти

| Код | Актори | Назва | Формулювання |
|---|---|---|---|
| 1 | 2 | 2 | 3 |
| АК1 | Адміністратор | Авторизація | Даний варіант використання дозволяє користувачу авторизуватись у системі . |
| К1 | Користувач | Реєстрація | Даний варіант використання дозволяє користувачу зареєструватися у системі. |
| К2 | Користувач | Вибір режиму | Даний варіант використання дозволяє користувачу обрати ігровий режим. |
| К3 | Користувач | Перегляд реклами | Даний варіант використання дозволяє користувачу переглянути рекламне відео для отримання бонусу. |
| К4 | Користувач | Налаштування профілю | Даний варіант використання дозволяє користувачу переглядати та змінювати налаштування свого профілю. |

Кінець таблиці 2.2.

| Код | Актори | Назва | Формулювання |
|---|---|---|---|
| 1 | 2 | 2 | 3 |
| К5 | Користувач | Прийняти запрошення у друзі | Даний варіант використання дозволяє користувачу прийняти запрошення у друзі від іншого гравця. |
| К6 | Користувач | Надіслати запрошення у друзі | Даний варіант використання дозволяє користувачу надіслати запрошення іншому гравцю для додавання у друзі. |
| К7 | Користувач | Пошук друга | Даний варіант використання дозволяє користувачу знайти іншого гравця за допомогою пошуку. |
| К8 | Користувач | Здійснення покупки | Даний варіант використання дозволяє користувачу придбати внутрішньоігровий контент або валюту. |
| А1 | Адміністратор | Модерація чату | Даний варіант використання дозволяє адміністратору переглядати та модераторити повідомлення у чаті. |
| А2 | Адміністратор | Здійснення технічної підтримки | Даний варіант використання дозволяє адміністратору обробляти звернення користувачів до техпідтримки. |
| А3 | Адміністратор | Додавання контену | Даний варіант використання дозволяє адміністратору додавати новий контент у гру. |
| А4 | Адміністратор | Видалення контенту | Даний варіант використання дозволяє адміністратору видалити існуючий контент із гри. |
| А5 | Адміністратор | Налаштування контенту | Даний варіант використання дозволяє адміністратору змінювати властивості вже наявного ігрового контенту. |

Згідно з визначеними варіантами використання побудовано діаграму варіантів використання для мобільної для видавництва «CrazyGames». Для створення діаграми було використано мову UML. Діаграму зображено на рисунку 2.2.

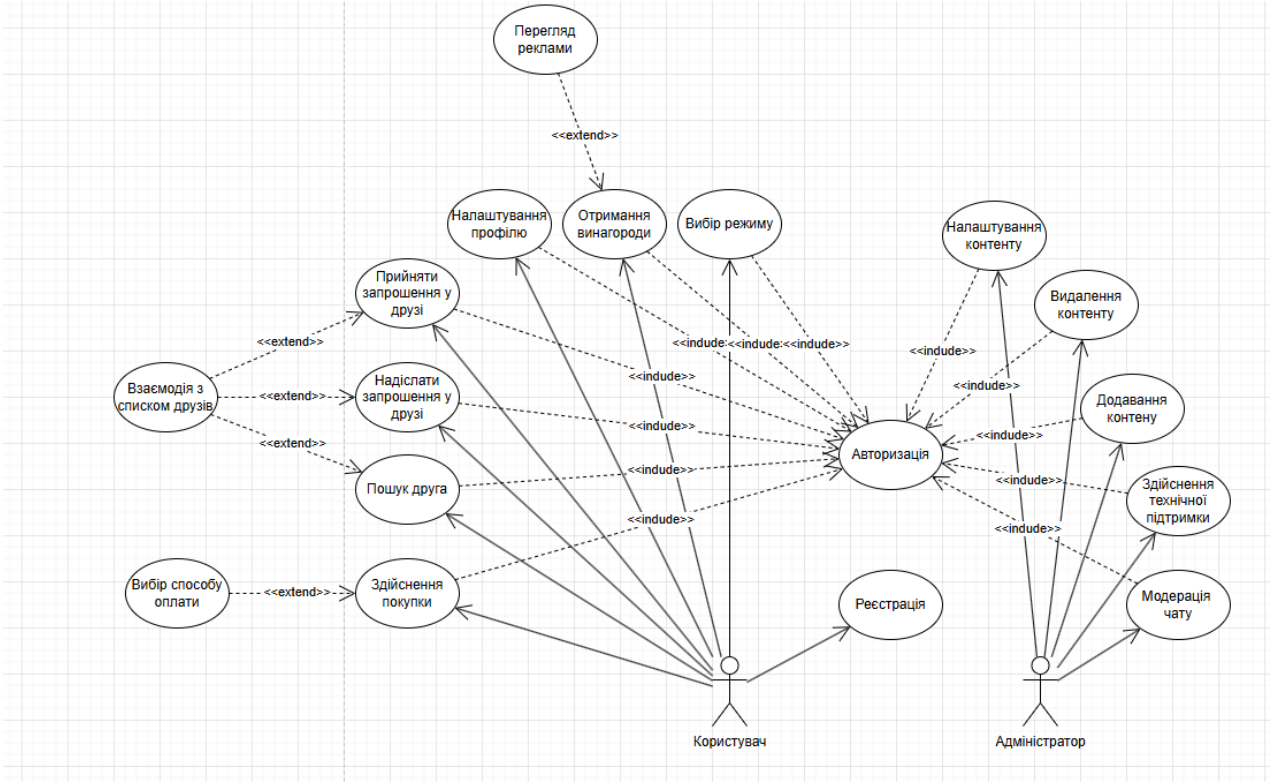



Рисунок 2.2 – Діаграма варіантів використання для мобільної гри

2.1.2 Розробка специфікацій прецедентів гри для видавництва «CrazyGames»

Для деталізаці прецедентів, предсталених на рисунку 2.2 до них було створено специфікації. Специфікація до врінту використання склдається з короткого опису прециденту, переліку дійових осіб, переліку передумов, основного потоку подій, льтернативного потоку подій та переліку постумов. Специфікації представлено у таблицях 2.3 – 2.16.

Таблиця 2.3 – Специфікація прецеденту «Авторизація»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє користувачу авторизуватись у системі . |
| Дійові особи прецеденту | Користувач, Адміністратор |
| Передумови | Користувач повинен мати спеціальний ідентифікатори (адреса електронної пошти та пароль, які він вводив на етапі реєстрації) для доступу в систему. |
| Основний потік подій | 1. Користувач ініціює авторизацію.<br>2. Система надає форму авторизації.<br>3. Користувач вводить адресу електронної пошти та пароль, які вводив при реєстрації аккаунту в задану форму.<br>4. Система перевіряє вірність введення даних та допускає до акаунту, або, якщо ведений пароль – невірний, то система повідомляє про це, та пропонує повторити введення або завершити варіант використання. |
| Альтернативний потік подій | Не визначено. |
| Постумови | Користувач отримує можливість створювати замовлення та залишати відгуки. Адміністратор отримує можливість використовувати привілеї адміністратора. |

Таблиця 2.4 – Специфікація прецеденту «Реєстрація»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє користувачу зареєструватися у системі. |
| Дійові особи прецеденту | Користувач |
| Передумови | Користувач повинен ввести адресу електронної пошти, яка ще не була зареєстрована. |

Кінець таблиці 2.4.

| Складова | Опис |
|---|---|
| Основний потік подій | 1. Користувач ініціює реєстрацію.<br>2. Система надає форму.<br>3. Користувач заповнює форму реєстрації.<br>4 Система перевіряє валідність введених даних і створює новий акаунт.<br>4 Система перевіряє валідність введених даних. Введена адреса вже зареєстрована. Система повідомляє про це, та пропонує можливість повторити введення або завершити варіант використання. |
| Альтернативний потік подій | Не визначено. |
| Постумови | Користувач отримує можливість створювати замовлення та залишати відгуки. |

Таблиця 2.5 – Специфікація прецеденту «Вибір режиму»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє користувачу обрати ігровий режим. |
| Дійові особи прецеденту | Користувач |
| Передумови | Користувач увійшов у гру та знаходиться в головному меню. |
| Основний потік подій | 1. Користувач переходить до розділу вибору режиму гри.<br>2. Система відображає доступні режими (наприклад: одиночний, мультиплеєр, виживання тощо).<br>3. Користувач обирає бажаний режим. |
| Альтернативний потік подій | Не визначено |
| Постумови | Система запускає гру у вибраному режимі. Користувач переходить до ігрового процесу. |

Таблиця 2.6 – Специфікація прецеденту «Перегляд реклами»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє користувачу переглянути рекламне відео для отримання бонусу. |
| Дійові особи прецеденту | Користувач |

Кінець таблиці 2.6.

| Складова | Опис |
|---|---|
| Передумови | Користувач знаходиться в інтерфейсі гри, де доступна пропозиція перегляду реклами. |
| Основний потік подій | 1. Користувач погоджується переглянути рекламу.<br>2. Система запускає рекламне відео.<br>3. Користувач переглядає рекламу до кінця. |
| Альтернативний потік подій | Користувач перериває перегляд -бонус не надається. |
| Постумови | Система нараховує бонус (наприклад, ігрову валюту, життя або інші ресурси). |

Таблиця 2.7 – Специфікація прецеденту «Налаштування профілю»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє користувачу переглядати та змінювати налаштування свого профілю. |
| Дійові особи прецеденту | Користувач |
| Передумови | Користувач авторизувався в системі та знаходиться в особистому кабінеті або головному меню. |
| Основний потік подій | 1. Користувач переходить до розділу «Профіль».<br>2. Система відображає поточні дані профілю.<br>3. Користувач редагує доступні поля .<br>4. Система зберігає зміни. |
| Альтернативний потік подій | Користувач скасовує зміни -профіль залишається без змін. |
| Постумови | Система оновлює дані профілю. Користувач бачить актуальну інформацію у своєму акаунті. |

Таблиця 2.8 – Специфікація прецеденту «Прийняти запрошення у друзі»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє користувачу прийняти запрошення у друзі від іншого гравця. |
| Дійові особи прецеденту | Користувач |

Кінець таблиці 2.8.

| Складова | Опис |
|---|---|
| Передумови | Користувач авторизований та має одне або кілька нових запрошень у друзі. |
| Основний потік подій | 1. Користувач відкриває список запрошень у друзі.<br>2. Система відображає отримані запрошення.<br>3. Користувач натискає кнопку «Прийняти» навпроти вибраного запрошення.<br>4. Система додає гравця до списку друзів. |
| Альтернативний потік подій | Користувач ігнорує або відхиляє запрошення -запрошення залишається без змін або видаляється. |
| Постумови | Система оновлює список друзів. Обидва користувачі бачать один одного у своєму списку друзів. |

Таблиця 2.9 – Специфікація прецеденту «Надіслати запрошення у друзі»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє користувачу надіслати запрошення іншому гравцю для додавання у друзі. |
| Дійові особи прецеденту | Користувач |
| Передумови | Користувач авторизований у системі та має доступ до функціоналу пошуку гравців. |
| Основний потік подій | подій 1. Користувач відкриває розділ пошуку друзів.<br>2. Вводить нікнейм, ID або обирає гравця зі списку рекомендованих.<br>3. Натискає кнопку «Додати в друзі».<br>4. Система надсилає запрошення вибраному гравцю. |
| Альтернативний потік подій | Пошук не дав результатів -користувач отримує відповідне повідомлення. |
| Постумови | Запрошення успішно надіслане. Стан очікує на підтвердження з боку іншого користувача. |

Таблиця 2.10 – Специфікація прецеденту «Пошук друга»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє користувачу знайти іншого гравця за допомогою пошуку. |
| Дійові особи прецеденту | Користувач |
| Передумови | Користувач авторизований у системі та знаходиться в розділі «Друзі» або «Соціальна взаємодія». |
| Основний потік подій | 1. Користувач відкриває вікно пошуку друзів.<br>2. Вводить ім’я гравця, ID або інший параметр пошуку.<br>3. Система відображає результати. |
| Альтернативний потік подій | Введено некоректний або неіснуючий запит -система повідомляє про відсутність результатів. |
| Постумови | Система успішно відображає знайденого гравця. Користувач може надіслати запрошення у друзі. |

Таблиця 2.11 – Специфікація прецеденту «Здійснення покупки»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє користувачу придбати внутрішньоігровий контент або валюту. |
| Дійові особи прецеденту | Користувач |
| Передумови | Користувач авторизований, має активне підключення до інтернету та прив'язаний платіжний метод. |
| Основний потік подій | 1. Користувач переходить до внутрішньоігрового магазину.<br>2. Обирає потрібний товар (валюта, скіни, бонуси).<br>3. Підтверджує покупку через платіжну систему.<br>4. Система обробляє транзакцію. |
| Альтернативний потік подій | Транзакцію скасовано або сталася помилка -покупка не здійснюється, користувач отримує повідомлення. |
| Постумови | Система зараховує придбаний контент на акаунт користувача. Покупка відображається у профілі. |

Таблиця 2.12 – Специфікація прецеденту «Модерація чату»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє адміністратору переглядати та модераторити повідомлення у чаті. |
| Дійові особи прецеденту | Адміністратор |
| Передумови | Адміністратор авторизований у системі та має відповідні права доступу. |
| Основний потік подій | 1. Адміністратор відкриває інтерфейс модерації чату.<br>2. Переглядає список повідомлень або скарг.<br>3. Обирає конкретне повідомлення.<br>4. Видаляє його або блокує користувача. |
| Альтернативний потік подій | Не визначено. |
| Постумови | Обране повідомлення видалено або користувача обмежено у праві надсилати нові повідомлення. |

Таблиця 2.13 – Специфікація прецеденту «Здійснення технічної підтримки»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє адміністратору обробляти звернення користувачів до техпідтримки. |
| Дійові особи прецеденту | Адміністратор |
| Передумови | Користувач надіслав звернення через інтерфейс технічної підтримки. Адміністратор авторизований. |
| Основний потік подій | 1. Адміністратор відкриває панель звернень.<br>2. Обирає конкретне звернення.<br>3. Аналізує проблему.<br>4. Надає відповідь або виконує необхідні дії (відновлення доступу, виправлення даних тощо). |
| Альтернативний потік подій | Не визначено. |
| Постумови | Користувач отримує відповідь. Проблема усунена або передана на подальший розгляд (якщо потрібно). |

Таблиця 2.14 – Специфікація прецеденту «Додавання контену»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє адміністратору додавати новий контент у гру. |
| Дійові особи прецеденту | Адміністратор |
| Передумови | Адміністратор авторизований у системі та має відповідні права доступу. |
| Основний потік подій | 1. Адміністратор входить до панелі керування контентом.<br>2. Обирає тип контенту (предмет, персонаж, рівень тощо).<br>3. Заповнює всі необхідні поля<br>4. Публікує новий контент. |
| Альтернативний потік подій | Не визначено. |
| Постумови | Новий контент з’являється у відповідному розділі гри та стає доступним для користувачів. |

Таблиця 2.15 – Специфікація прецеденту «Видалення контенту»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє адміністратору видалити існуючий контент із гри. |
| Дійові особи прецеденту | Адміністратор |
| Передумови | Адміністратор авторизований у системі та має права на редагування контенту. |
| Основний потік подій | 1. Адміністратор відкриває список доступного контенту.<br>2. Обирає одиницю контенту для видалення.<br>3. Підтверджує видалення.<br>4. Система остаточно видаляє контент із бази даних. |
| Альтернативний потік подій | Не визначено. |
| Постумови | Вибраний контент більше недоступний для користувачів. Дані видалено або переміщено до архіву. |

Таблиця 2.16 – Специфікація прецеденту «Налаштування контенту»

| Складова | Опис |
|---|---|
| Короткий опис | Даний варіант використання дозволяє адміністратору змінювати властивості вже наявного ігрового контенту. |
| Дійові особи прецеденту | Адміністратор |
| Передумови | Адміністратор авторизований у системі та має доступ до редагування контенту. |
| Основний потік подій | 1. Адміністратор відкриває панель керування контентом.<br>2. Обирає елемент контенту для редагування.<br>3. Вносить зміни у відповідні параметри<br>4. Зберігає зміни. |
| Альтернативний потік подій | Не визначено. |
| Постумови | Оновлений контент стає доступним у грі з новими налаштуваннями. |

## 2.2 Розробка архітектури мобільної гри для видавництва «CrazyGames».

За вимогами до мобільної гри для видавництва «CrazyGames» було обрано оптимальний архітектурний стиль – MVC (Model-view-controller).

MVC – шаблон проєктування, у якому вся логіка програмного забезпечення поділена на такі компоненти:

– Model: Модель з даними та логікою. У випадку розроблюваної гри модель використовується для роботи с базою даних, їх створення, редагування, видалення та збереження. Також модель буде відповідати за взаємодію гравця з ігровим середовищем, надаватиме йому змогу взаємодіяти з предметами, ворогами та іншими гравцями.

– View: Цей компонент відповідає за відображення даних. У випадку з мобільною грою він відображає компоненти користувацького інтерфейсу, які можуть містити в собі форми для внесення та редагування даних, списки або таблиці для їх відображення, контролери та кнопки для керування персонажем і взаємодії гравця з оточенням.

– Controller: Контролер здійснює обробку дій користувача та взаємодію між моделлю та візуальним представленням. Цей компонент відповідатиме за обробку дій гравця з оточенням та іншими елементами гри, виклик відповідних методів та операцій.

На рисунку 2.3 представлено схему взаємодії компонентів шаблону MVC.

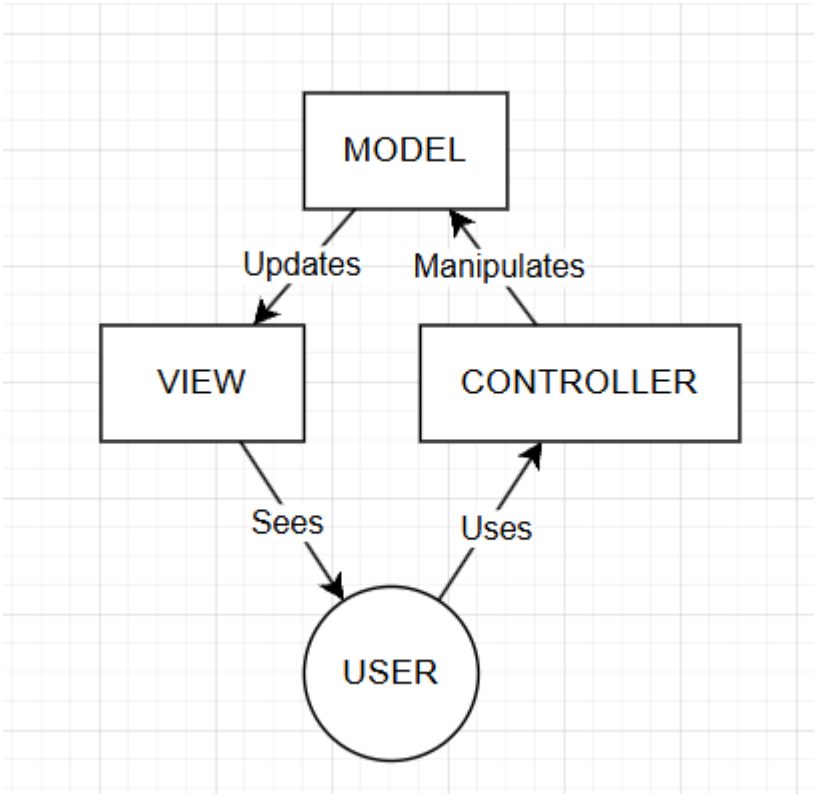



Рисунок 2.3 - схема взаємодії компонентів шаблону MVC

Розробка діаграми компонентів мобільної гри для видавництва «CrazyGames».

Діаграма компонентів – статична структурна діаграма, котра показує розділення програмного забезпечення на структурні компоненти та залежності між ними.

Діаграму компонентів мобільної гри показано на рисунку 2.4.

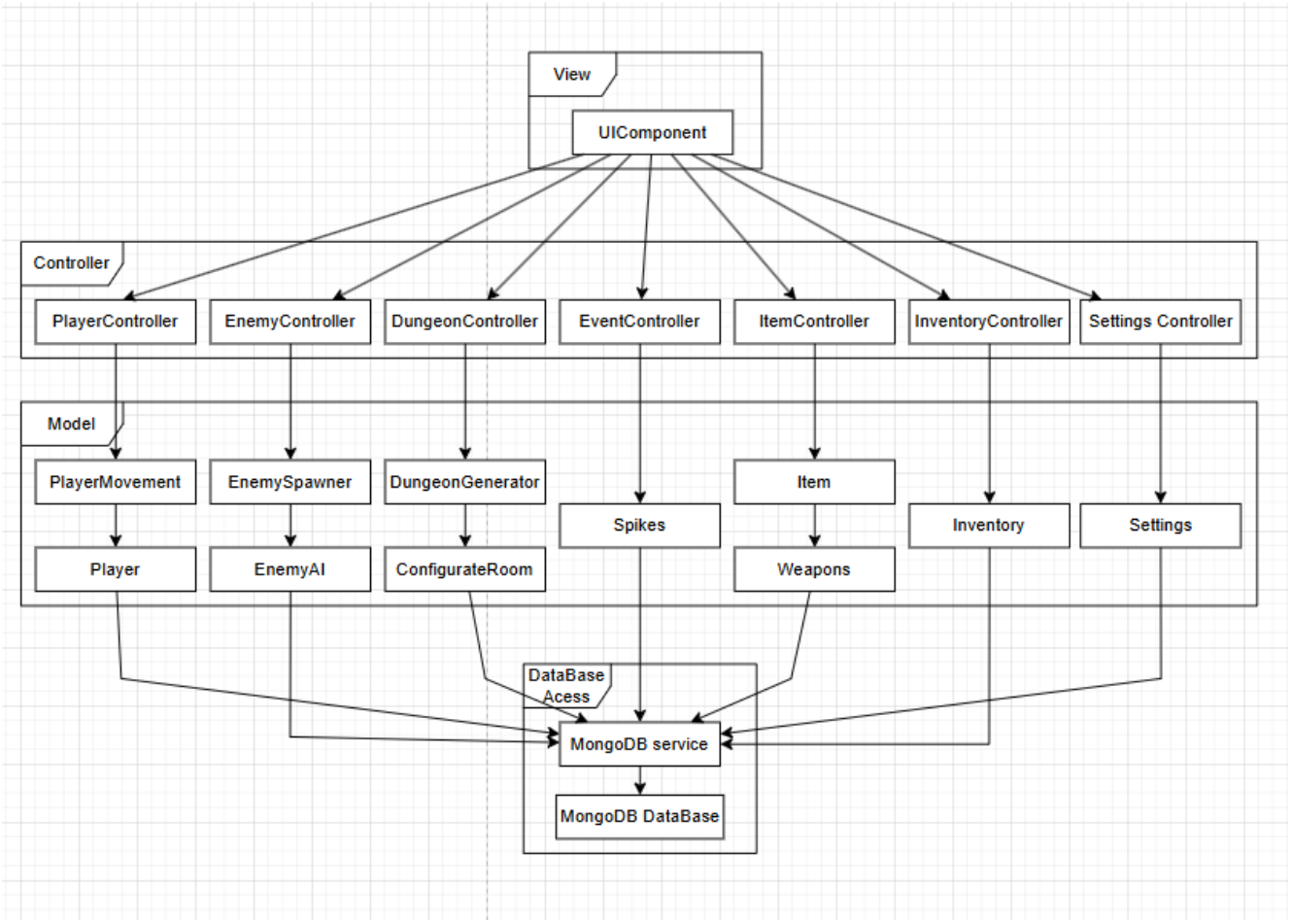



Рисунок 2.4 - Діаграма компонентів мобільної гри

Розробка діаграми розгортання мобільної гри для видавництва «CrazyGames».

Діаграма розгортання – діаграма, що відображає архітектурну структурну проєкту, яка містить різні обчислювальні вузли за зв’язки між ними. Діаграма розгортання дозволяє зобразити фізичне розміщення її складових, що дозволяє розглянути їх взаємодію та вплив на масштабованість мобільної гри.

Діаграму розгортання мобільної гри для видавництва «CrazyGames» представлено на рисунку 2.5.

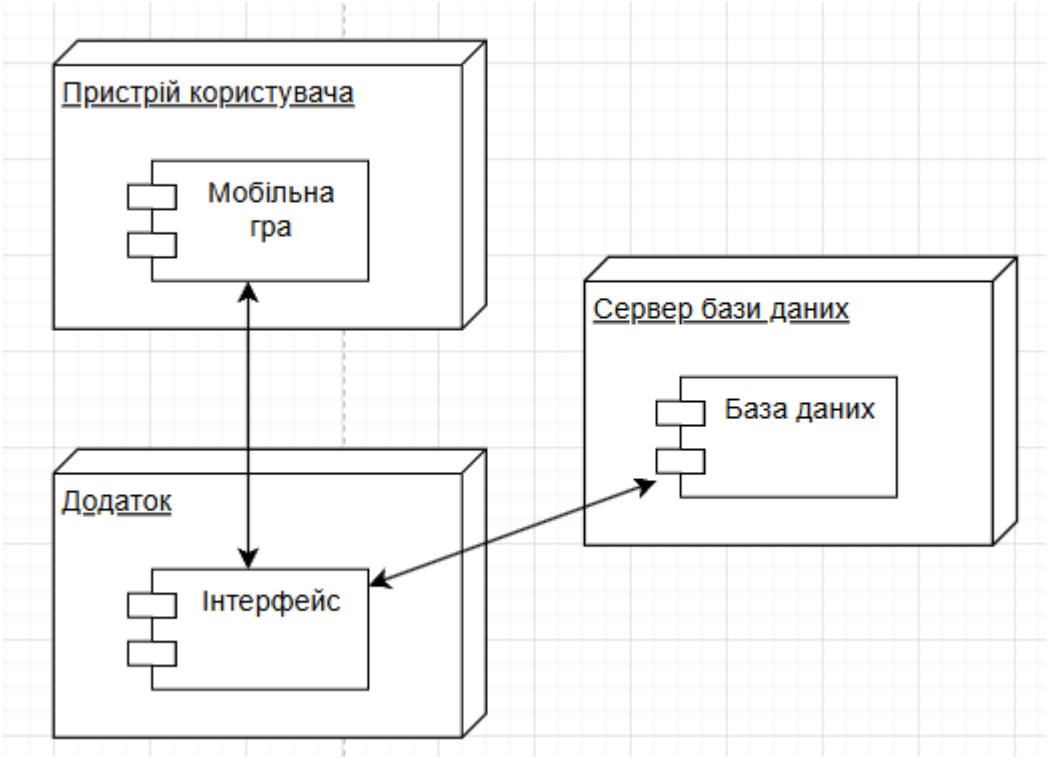



Рисунок 2.5 - Діаграма розгортання мобільної гри для видавництва «CrazyGames»

2.3 Проєкт мобільної гри для видавництва «CrazyGames».

Під час розробки проєкту були розроблені ескізний, технічний та робочий проєкти.

2.3.1 Ескізний проєкт мобільної гри для видавництва «CrazyGames».

Для систематизованого представлення варіантів використання били розроблені діаграми діяльності для кількох прецедентів, до них також було створено проєкт користувацького інтерфейсу та інформаційну модель у вигляді концептуальної моделі.

Розробка діаграм діяльності основних варіантів використання мобільної гри для видавництва «CrazyGames».

Діаграма діяльності UML створюється для візуалізації графу діяльностей.

Вона відображає переходи потоків управління в програмному забезпеченні

між діями, та також паралельні дії і потоки подій. Ця діаграма містить у собі такі елементи як: «Початок», «Кінець», «Перехід», «Діяльність», «Елемент вибору».

Діаграми діяльності для прецедентів «Вибір режиму» та «Налаштування профілю» мобільної гри зображено на рисунках 2.6 - 2.7.

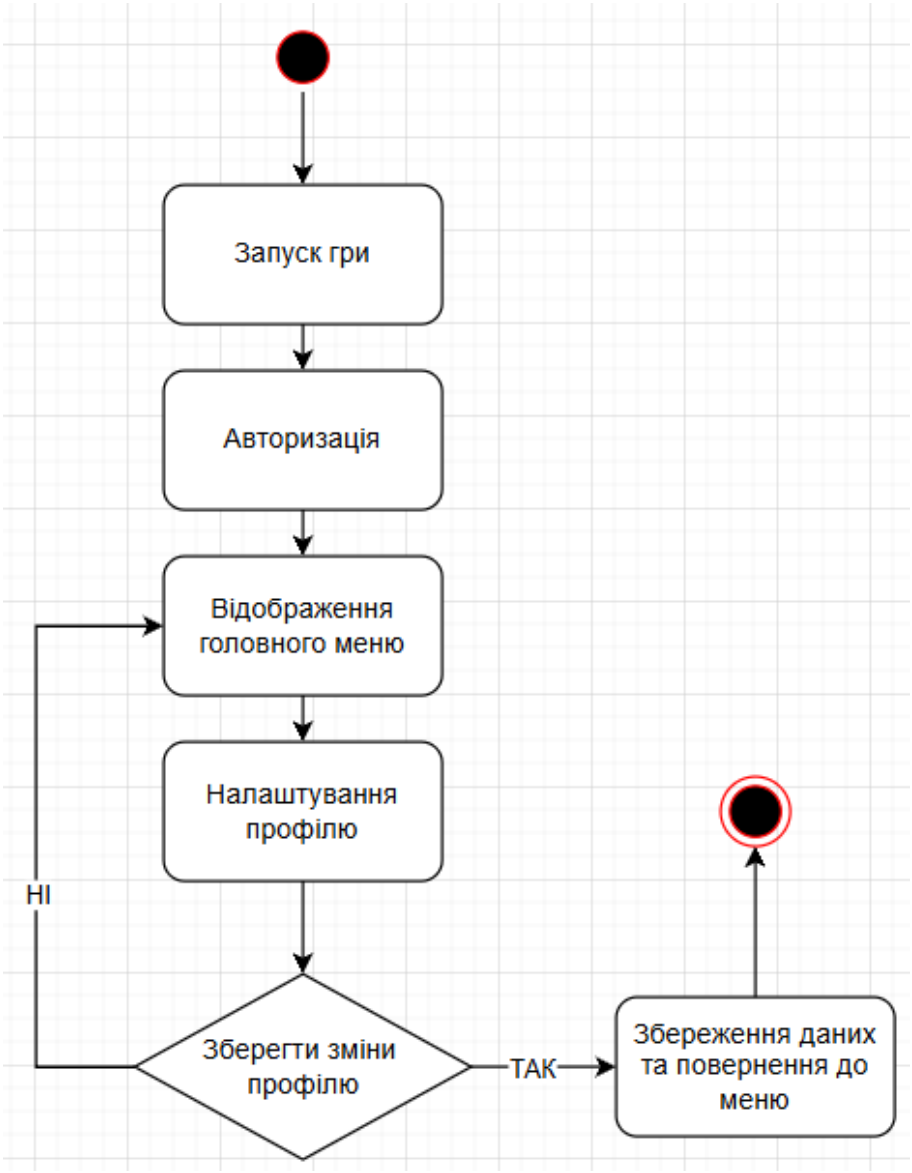



Рисунок 2.6 – Діаграма діяльності прецеденту «Налаштування профілю»

На рисунку 2.7 зображено діаграму діяльності прецеденту «Вибір режиму».

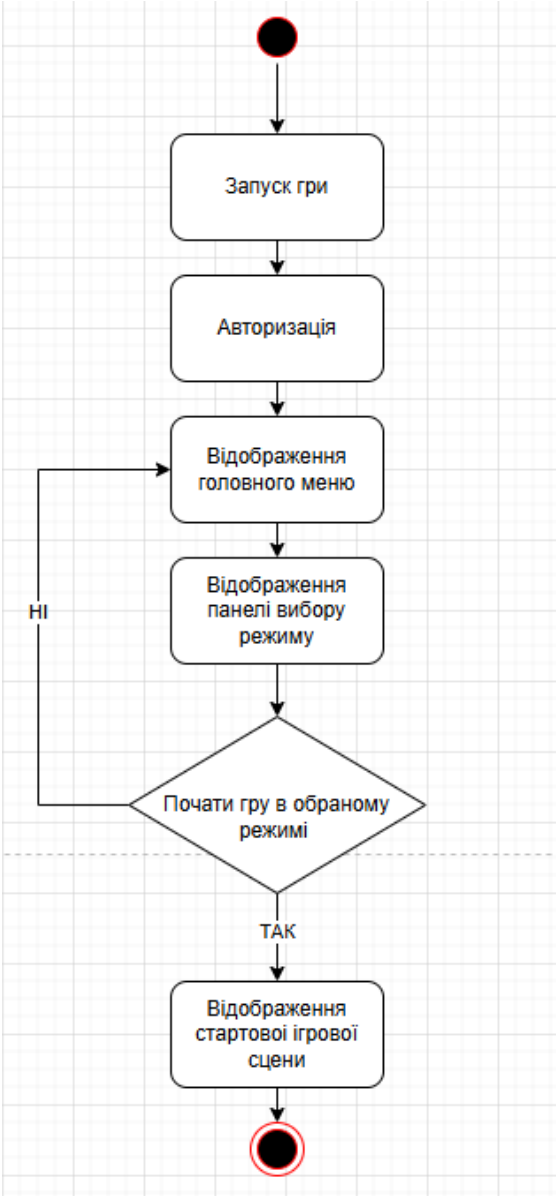



Рисунок 2.7 - Діаграма діяльності прецеденту «Вибір режиму»

Розробка концептуальної моделі даних мобільної гри для видавництва «CrazyGames».

Концептуальне проєктування – створення схеми або моделі бази даних на концептуальному рівні, вона містить визначення сутностей, атрибутів та взаємодій між ними.

Одним із засобів зображення концептуальної моделі є діаграма «сутність-зв’язок».

Визначимо сутності для мобільної гри.

Сутність – це об’єкт, який представляє основні поняття розроблюваного продукту. Вони містять певні атрибути (характеристики чи властивості), які дозволяють більш детально їх описати.

Список сутностей мобільної гри для видавництва «CrazyGames»:

- Налаштування;
- Оплата;
- Гравець;
- Адміністратор;
- Предмети;
- Звіти;
- Режими гри;
- Вороги.

Визначимо атрибути сутностей мобільної гри для видавництва «CrazyGames».

Кожна сутність має конкретні характеристики (атрибути). Кожен атрибут має унікальне ім’я в межах сутності. Кожна сутність має бути унікальною та відрізнятися від будь-якого іншого атрибута в межах цієї сутності.

Список атрибутів сутності Налаштування:

- Гучність;
- Тип керування.

Список атрибутів сутності Оплата:

- Дата;
- Форма;
- Спосіб оплати;
- ID Оплати.

Список атрибутів сутності Гравець:

- Пароль;

- Рівень;
- Логін;
- ID Гравця;
- Аватар;
- Баланс.

Список атрибутів сутності Адміністратор:

- Логін;
- Пароль;
- ID Адміністратора.

Список атрибутів сутності Предмети:

- ID Предмету;
- Рівень;
- Назва;
- Шкода.

Список атрибутів сутності Звіти:

- ID Звіту;
- Текст звіту;
- Тип звіту.

Список атрибутів сутності Режими гри:

- Складність;
- Тип режиму.

Список атрибутів сутності Вороги:

- Тип;
- Рівень;
- Шкода;
- Максимальне здоров’я.

Побудуємо модель «сутність-зв’язок».

Зв’язки між сутностями мобільної гри представлено у таблиці 2.17.

Таблиця 2.17 – Зв’язки між сутностями

| Сутність 1 | Сутність 2 | Зв’язок |
|---|---|---|
| Налаштування | Гравець | Один-до-одного |
| Оплата | Гравець | Один-до-багатьох |
| Гравець | Предмети | Багато-до-багатьох |
| Вороги | Режими гри | Багато-до-багатьох |
| Адміністратор | Гравець | Багато-до-багатьох |
| Звіти | Адміністратор | Один-до-багатьох |

На рисунку 2.8 представлена концептуальна модель бази даних мобільної гри.

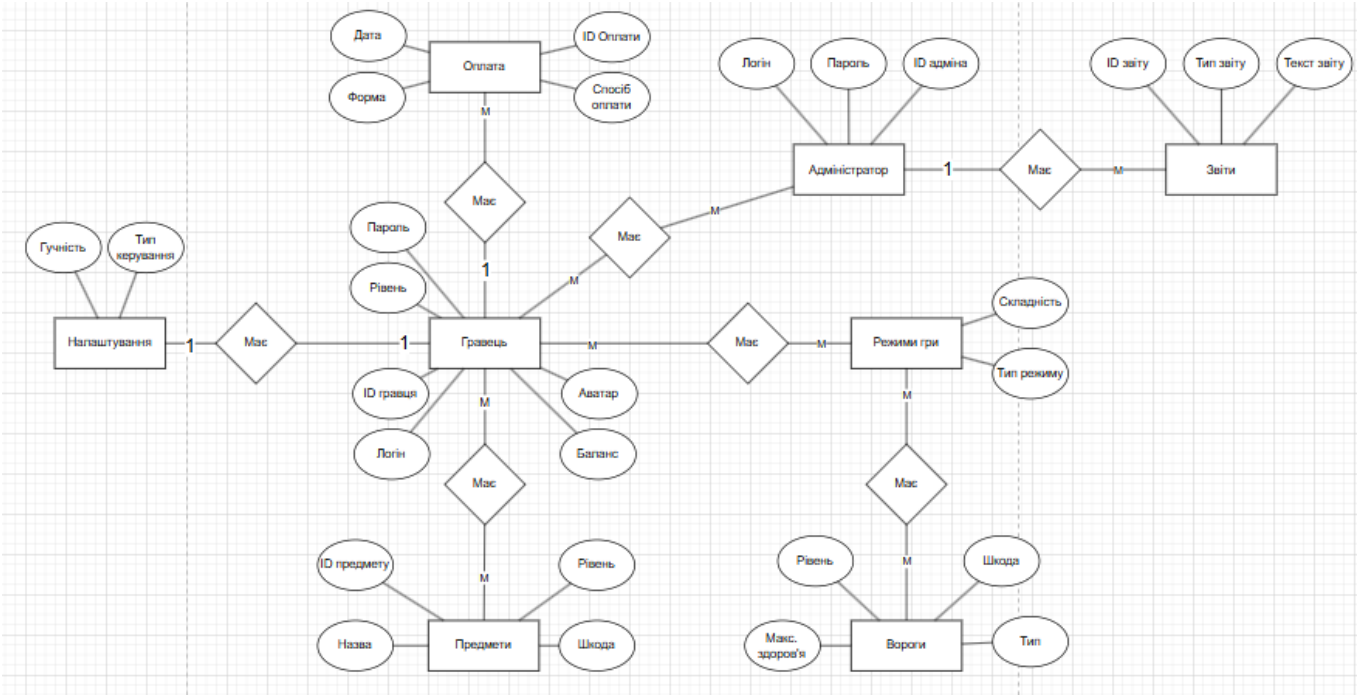



Рисунок 2.8 - Концептуальна модель даних мобільної гри

Розробка проєкту користувацького інтерфейсу мобільної гри для видавництва «CrazyGames».

Розробка інтерфейсу користувача є одним з основних етапів створення мобільної гри. Відповідно до потреб мобільної гри було розроблено ескізи користувацького інтерфейсу до прецедентів «Вибір режиму» та «Налаштування профілю». На рисунках 2.9 - 2.10 зображені ескізи сцен до відповідних прецедентів.


Рисунок 2.9 – Ескіз вікна до прецеденту «Вибір режиму»

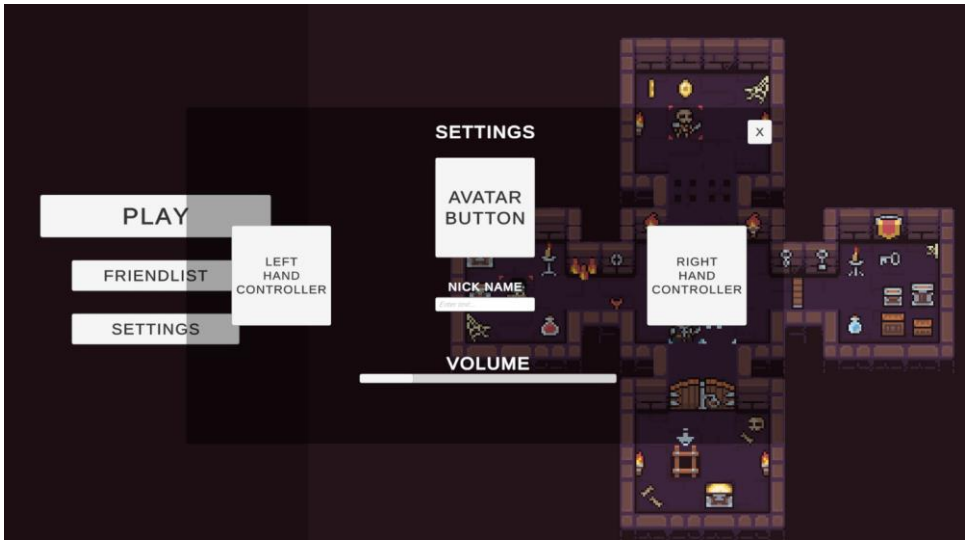



Рисунок 2.10 – Ескіз вікна до прецеденту «Налаштування профілю»

### 2.3.2 Технічний проєкт мобільної гри для видавництва «CrazyGames».

Беручи до уваги розроблений ескізний проєкт, створимо технічний проєкт мобільної гри для видавництва «CrazyGames». Він міститиме статичну модель представлену діаграмою класів та динамічну модель у вигляді діаграми послідовностей.

Побудова статичної моделі мобільної гри для видавництва «CrazyGames».

Статична модель мобільної гри представлена діаграмою класів. Діаграма класів – структурна діаграма, що показує статичну структуру програмного забезпечення, представляючи класи їх атрибути та залежності. Діаграми класів допомагають зобразити структуру мобільної гри та з’ясувати взаємозв’язки між її компонентами. Діаграму класів мобільної гри для видавництва «CrazyGames» показано на рисунку 2.11.

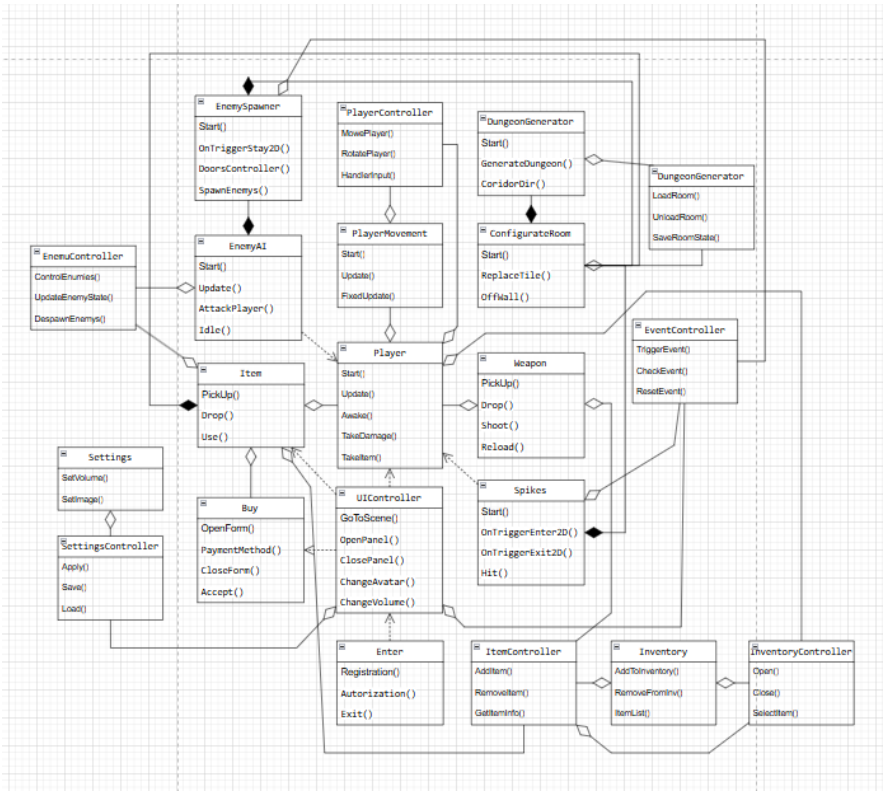



Рисунок 2.11 - Діаграма класів мобільної гри для видавництва «CrazyGames»

Специфікація класів мобільної гри для видавництва «CrazyGames».

Специфікація класу для об’єкта – це визначення його властивостей та методів. Специфікації мобільної гри для видавництва «CrazyGames» представлено у таблицях 2.18 – 2.30.

Таблиця 2.18 – Специфікація класу «EnemySpawner».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за появу ворогів у ігрових локаціях. |
| Атрибути | List<GameObject> enemys – об’єкти ворогів, що можуть зя’витися;<br>GameObject Doors – шаблон стіни;<br>int countOfWaves – кількість хвиль ворогів;<br>int countOfEnemys – кількість ворогів;<br>Vector2[] spawnBounds – межі у яких з’являються вороги;<br>int currentWaves – поточна хвиля. |
| Методи | Start() – відповідає за виклик методів при запуску сцени;<br>OnTriggerStay2D() - відповідає за виклик методів коли гравець знаходиться у межах кімнати;<br>DoorsController() – відповідає за закриття виходів коли гравець заходить в кімнату; |

Таблиця 2.19 – Специфікація класу «EnemyAI»

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за логіку ворогів та їх поведінку. |
| Атрибути | GameObject Player – персонаж гравця;<br>int Damage – кількість шкоди яку завдає ворог;<br>int maxHP – максимальна кількість здоров’я;<br>int Speed – значення швидкості ворога;<br>int currentHP – поточна кількість здоров’я ворог. |
| Методи | Start() – відповідає за виклик методів при запуску сцени;<br>Update() - відповідає за виклик методів кожний ігровий кадр;<br>AttackPlayer() – відповідає за атаку ворогом гравця;<br>Idle() – відповідає очікування ворогом гравця. |

Таблиця 2.20 – Специфікація класу «PlayerMovement».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за переміщення персонаж гравця. |
| Атрибути | float moveSpeed – швидкість гравця;<br>Vector2 movement – направлення руху гравця. |
| Методи | Start() – відповідає за виклик методів при запуску сцени;<br>Update() - відповідає за виклик методів кожний ігровий кадр;<br>FixedUpdate() – відповідає за розрахунок фізики переміщення. |

Таблиця 2.21 – Специфікація класу «DungeonGenerator».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за генерацію кімнат. |
| Атрибути | GameObject[] _rooms – шаблони всіх кімнат;<br>int _roomsCount – кількість кімнат;<br>Dictionary<string ,Vector3> _spawnDir – словник з направленням кімнат;<br>List<GameObject> SpawnedRooms – список кімнат що повинні з’явитися. |
| Методи | Start() – відповідає за виклик методів при запуску сцени;<br>GenerateDungeon() - відповідає за генерацію кімнат;<br>CoridorDir()– відповідає за розташування виходів з кімнат. |

Таблиця 2.22 – Специфікація класу «Item».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за взаємодію гравця з предметами. |
| Атрибути | GameObject Player – шаблон гравця;<br>String Type – тип предмета;<br>Int Count – кількість предметів. |
| Методи | PickUp() – відповідає можливість гравця взяти предмет;<br>Dror() – відповідає за можливість гравця викинути предмет;<br>Use()– відповідає за можливість гравця використати предмет. |

Таблиця 2.23 – Специфікація класу «Buy».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за можливість гравця придбати внутрішньоігрові предмети. |
| Атрибути | Int productID – індекс товару;<br>String PaymentMethod – спосіб оплати;<br>Int Count – кількість товару;<br>GameObject PaymentForm – шаблон панелі оплати. |
| Методи | OpenForm() - відповідає за відкриття форми покупки;<br>PaymentMethod() – відповідає за можливість гравця вибрати метод оплати;<br>CloseForm() – відповідає за можливість гравця закрити форму оплати.<br>Accept() – відповідає за підтвердження покупки. |

Таблиця 2.24 – Специфікація класу «ConfigurateRoom».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за конфігурацію кімнат. |
| Атрибути | List<ReplacingData> repData – данні для заміни;<br>Tilemap tilemap – шаблон карти кімнати;<br>GameObject[] Walls – шаблони стін; |
| Методи | Start()– відповідає за виклик методів при запуску сцени;<br>OffWall() – відповідає за вимикання стін у кімнатах;<br>ReplaceTile() – відповідає за заміну елементів кімнати. |

Таблиця 2.25 – Специфікація класу «Player».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за основні взаємодії користувача з середовищем |
| Атрибути | float hp – кількість здоров’я гравця;<br>float mp – кількість сили гравця;<br>float sp – кількість витривалості гравця;<br>GameObject uiDeathPanel – шаблон панелі поразки;<br>Image uiHP – індикатор здоров’я гравця. |
| Методи | Start()– відповідає за виклик методів при запуску сцени;<br>Awake() – відповідає за виклик методів до початку сцени;<br>Update() – відповідає за виклик методів кожний ігровий кадр;<br>TakeDamage() – відповідає за отримання шкоди гравцем;<br>Death() – відповідає за метод смерті гравця. |

Таблиця 2.26 – Специфікація класу «Weapon».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за зброю, її появу та взаємодію з гравцем. |
| Атрибути | int damage – кількість шкоди, яку завдає зброя;<br>String Type – тип зброї;<br>Int Ammo – кількість куль;<br>Float reloadTime – час перезарядки;<br>Int WeaponLvl– рівень зброї;<br>Image WeaponIMG – іконка зброї. |
| Методи | Start()– відповідає за виклик методів при запуску сцени;<br>PickUp() – відповідає за можливість гравця підняти зброю;<br>Drop() – відповідає за можливість гравця викинути зброю;<br>Shoot() – відповідає за стріляти зі зброї;<br>Reload() – відповідає за перезарядку зброї. |

Таблиця 2.27 – Специфікація класу «Spikes».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за елементи оточення, що можуть завдавати шкоду гравцю. |

Кінець таблиці 2.27.

| Характеристика | Опис |
|---|---|
| Атрибути | Animator anim – компонент відповідальний за анімацію;<br>bool playerEnter – змінна, що підтверджує вхід гравця у зону дії об’єкта;<br>string targetTag – тег гравця, якому завдано шкоди;<br>int damage – кількість шкоди; |
| Методи | Start()– відповідає за виклик методів при запуску сцени;<br>OnTriggerEnter2D() – відповідає за виклик методів коли гравець стоїть у зоні дії об’єкта;<br>OnTriggerExit2D() – відповідає за виклик методів коли гравець виходить з зони дії об’єкта;<br>Hit() – відповідає за отримання гравцем шкоди. |

Таблиця 2.28 – Специфікація класу «UIController».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за компоненти користувацького інтерфейсу. |
| Атрибути | GameObject inventoryPanel – панель інвентаря;<br>GameObject settingsPanel – панель налаштувань;<br>GameObject shopPanel – панель магазину;<br>Slider volumeSlider – слайдер гучності;<br>TextMeshProUGUI coinText – кількість монет;<br>GameObject itemSlotPrefab панель зброї; |
| Методи | GoToScene() – Відповідає за переходи між сценами;<br>OpenPanel() – Відповідає за відкриття панелі;<br>ClosePanel() – Відповідає за закриття панелі;<br>ChangeAvatar() – Відповідає за зміну фото профілю гравця;<br>ChangeVolume() – Відповідає за зміну гучності. |

Таблиця 2.29 – Специфікація класу «Enter».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за реєстрацію, вихід та вхід у обліковий запис гравця або адміністратора.. |

Кінець таблиці 2.29.

| Характеристика | Опис |
|---|---|
| Атрибути | string username – ім’я користувача;<br>string password – пароль користувача;<br>string email – електронна пошта користувача;<br>InputField usernameInput – поле для вводу імені;<br>InputField passwordInput – поле для вводу паролю;<br>InputField emailInput поле для вводу електронної пошти;<br>GameObject loginPanel – панель для входу в гру;<br>GameObject registrationPanel – панель реєстрації;<br>TextMeshProUGUI errorMessage – повідомлення про помилку входу. |
| Методи | Registration()– Відповідає за реєстрацію гравців;<br>Autorization()– Відповідає за авторизацію користувача або адміністратора;<br>Exit() – Відповідає за вихід з облікового запису; |

Таблиця 2.30 – Специфікація класу «PlayerController».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за керування гравцем та обробку введення користувача. |
| Атрибути | Player player – об'єкт гравця;<br>PlayerMovement movement – компонент руху;<br>UIController uiController – панель інтерфейсу. |
| Методи | HandleInput() – обробляє введення з пристрою;<br>MovePlayer() – відповідає за переміщення гравця;<br>RotatePlayer() – відповідає за обертання гравця. |

Таблиця 2.31 – Специфікація класу «EnemyController».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за керування ворогами, їхній стан та появу. |
| Атрибути | List<EnemyAI> enemies – список ворогів;<br>EnemySpawner spawner – компонент генерації ворогів. |

Кінець таблиці 2.31.

| Характеристика | Опис |
|---|---|
| Методи | ControlEnemies() – керує поведінкою ворогів;<br>UpdateEnemyState() – оновлює стан ворогів;<br>DespawnEnemy() – видаляє ворога зі сцени. |

Таблиця 2.32 – Специфікація класу «DungeonController».

| Характеристика | Опис |
|---|---|
| Відповідальність | Керує процесом генерації та зміни підземель. |
| Атрибути | DungeonGenerator generator – генератор підземель;<br>ConfigurateRoom configurator – конфігуратор кімнат. |
| Методи | LoadRoom() – завантажує кімнату;<br>UnloadRoom() – видаляє кімнату;<br>SaveRoomState() – зберігає стан кімнати. |

Таблиця 2.33 – Специфікація класу «EventController».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за ігрові події та їхню логіку. |
| Атрибути | List<Spikes> traps – список пасток;<br>EnemySpawner spawner – спавнер ворогів;<br>UIController uiController – інтерфейс для подій. |
| Методи | TriggerEvent() – активує подію;<br>CheckEventConditions() – перевіряє умови події;<br>ResetEvent() – скидає подію. |

Таблиця 2.34 – Специфікація класу «ItemController».

| Характеристика | Опис |
|---|---|
| Відповідальність | Керує логікою предметів, їх додаванням та видаленням. |
| Атрибути | List<Item> items – список предметів;<br>List<Weapon> weapons – список зброї;<br>Inventory inventory – інвентар гравця. |
| Методи | AddItem() – додає предмет;<br>RemoveItem() – видаляє предмет;<br>GetItemInfo() – повертає інформацію про предмет. |

Таблиця 2.35 – Специфікація класу «InventoryController».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за управління інвентарем гравця. |
| Атрибути | Inventory inventory – сам інвентар;<br>ItemController itemController – контролер предметів;<br>Player player – посилання на гравця. |
| Методи | Open() – відкриває інвентар;<br>Close() – закриває інвентар;<br>SelectItem() – вибирає предмет. |

Таблиця 2.36 – Специфікація класу «Inventory».

| Характеристика | Опис |
|---|---|
| Відповідальність | Зберігає усі предмети, що належать гравцю. |
| Атрибути | List<Item> items – предмети в інвентарі. |
| Методи | AddToInventory() – додає предмет;<br>RemoveFromInv() – видаляє предмет;<br>ListItems() – виводить список усіх предметів. |

Таблиця 2.37 – Специфікація класу «SettingsController».

| Характеристика | Опис |
|---|---|
| Відповідальність | Відповідає за застосування та збереження налаштувань. |
| Атрибути | Settings settings – модель налаштувань;<br>UIController uiController – інтерфейс налаштувань. |
| Методи | Apply() – застосовує налаштування;<br>Load() – завантажує збережені налаштування;<br>Save() – зберігає нові налаштування. |

Таблиця 2.38 – Специфікація класу «SettingsController».

| Характеристика | Опис |
|---|---|
| Відповідальність | Зберігає дані конфігурації користувача. |
| Атрибути | float volume – рівень гучності;<br>int graphicsQuality – якість графіки;<br>string controlScheme – тип керування. |
| Методи | SetVolume() – встановлює гучність;<br>SetImage() – встановлює аватар. |

Розробка динамічної моделі мобільної гри для видавництва «CrazyGames»

Динамічна модель мобільної гри представлена у вигляді діаграми послідовності. Діаграми послідовності – один з видів діаграм мови UML, який використовується для візуалізації динамічної поведінки програмного забезпечення.

Вони показують послідовність взаємодії об’єктів. Основними складовими цієї діаграми є актори, об’єкти, повідомлення, інтервали активності та життєві лінії. Діаграми послідовності для прецедентів «Авторизація» і «Налаштування профілю» показано на рисунках 2.12 - 2.13.

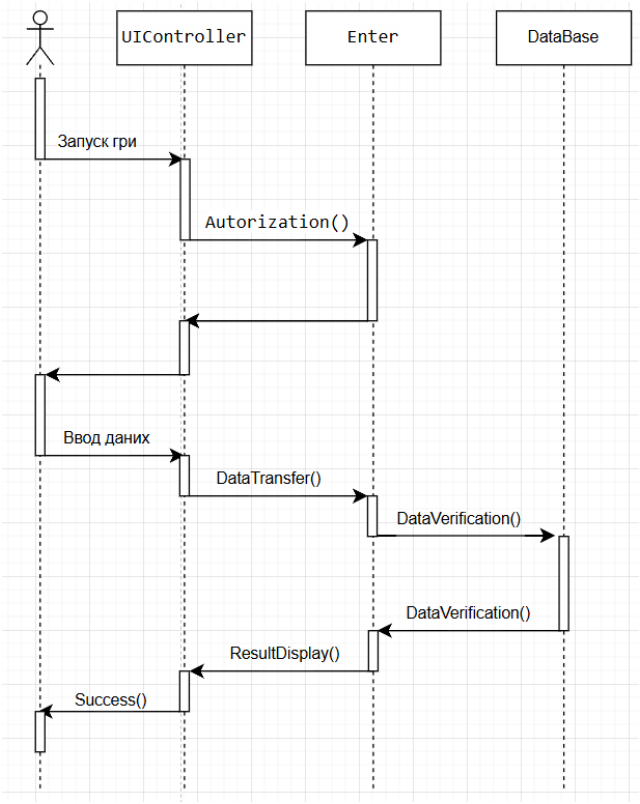



Рисунок 2.12 – Діаграма послідовності для прецеденту «Авторизація»

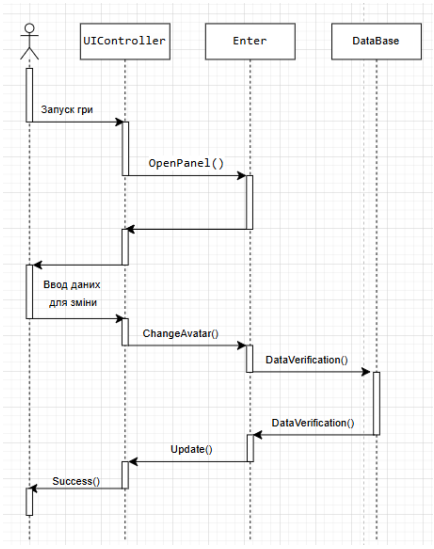



Рисунок 2.13 – Діаграма послідовності для прецеденту «Налаштування профілю»

Розробка логічної моделі даних мобільної гри для видавництва «CrazyGames»

Особливе значення при побудові логічної моделі бази даних мають функціонатьні залежності. Правильне встановлення функціональних залежностей

дозволяють побудувати найбільш вдалий варіант бази даних, що забезпечить надійне зберігання інформації та швидку обробку.

В процесі створення логічної моделі було визначено всі типи атрибутів, виявлені ключі для кожного з відношень так обов'язковість їх заповнення. Розроблена логічна модель даних представлена на рисунку 2.14.

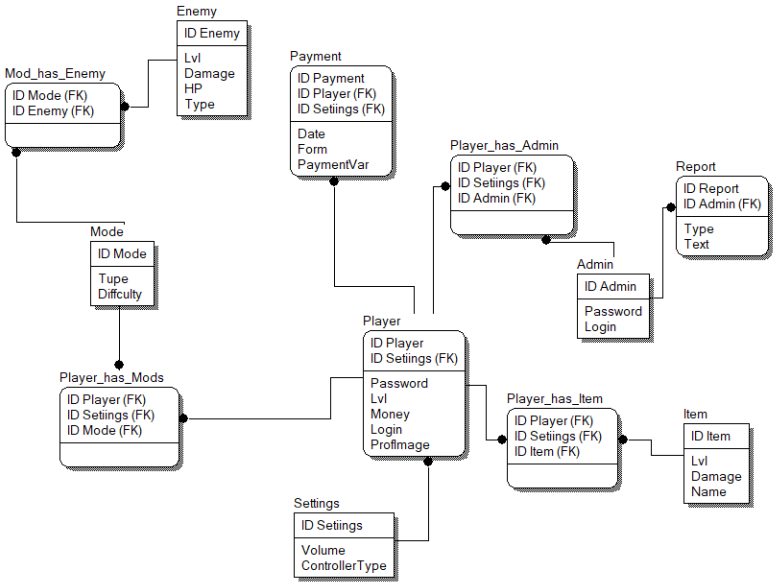



Рисунок 2.14 – логічна модель даних

2.3.3 Робочий проєкт мобільної гри для видавництва «CrazyGames»

Обґрунтування вибору засобів програмування мобільної гри для видавництва «CrazyGames»

Для створення мобільної гри для видавництва «CrazyGames» мною були обрані такі засоби розробки:

– Мова програмування C#. Це сучасна об'єктно-орієнтована мова програмування. Мова активно використовується для створення додатків, сайтів, ігор, мобільних застосунків, хмарних сервісів та корпоративного ПЗ. Особливою популярністю C# користується в ігровій індустрії завдяки інтеграції з ігровим рушієм Unity. Завдяки розвитку .NET Core та .NET 5 і вище, C# став зручним інструментом, який дозволяє розробляти програми для Windows, Linux, macOS, Android та iOS.

– Рушій Unity. це потужний та популярний ігровий рушій, який дозволяє створювати 2D, 3D ігри. Unity відомий своєю зручністю для початківців, великим вибором готових інструментів, візуальним редактором сцен і можливістю експортувати проєкти на різні платформи, включно з Windows, macOS, Android, iOS та WebGL. Завдяки своїй гнучкості, активній спільноті та широким можливостям, Unity став одним з найпопулярніших інструментів для розробки ігор.

– Середовище керування базами даних MongoDB. Вона ідеально підходить для роботи з великими обсягами неструктурованих або слабо структурованих даних. Вона підтримує реплікацію, індексацію, агрегації, а також має зручний запитний синтаксис. MongoDB легко інтегрується з мовами програмування, такими як C#. Її було обрано завдяки своїй гнучкості, простоті у використанні.

– Середовище розробки Visual Studio Code. Це безкоштовний, легкий, текстовий редактор для розробки програмного забезпечення. Він підтримує велику кількість мов програмування, зокрема C#. Однією з головних переваг редактора є система розширень, яка дозволяє налаштувати середовище розробки під власні потреби.

Реалізація основних класів мобільної гри для видавництва «CrazyGames»

Мобільна гра розробляється с застосуванням мови програмування C#. Також використовуються основні бібліотеки рушія Unity. Нижче наведено програмний код для класів руху гравця та появи ворогів. Повний код програми надано у додатку Д.

EnemySpawner.cs

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

public class EnemySpawner : MonoBehaviour
{
    [SerializeField] private List<GameObject> enemys = new List<GameObject>();
```

```
[SerializeField] private GameObject Doors;
[SerializeField, Range(1,5)] private int countOfWaves;
[SerializeField, Range(1,8)] private int countOfEnemys;
[SerializeField] private Vector2[] spawnBounds;
private bool WavesAreEnd;
private int curentWaves;

void Start()
{
    for (int i = 0; i < spawnBounds.Length; i++)
    {
        spawnBounds[i] += (Vector2)transform.position;
    }
}

private void OnTriggerStay2D(Collider2D other) {
    if(other.gameObject.tag == "Player"){
        SpawnEnemys();
        DoorsController();
    }
}

private void DoorsController(){
    if(!WavesAreEnd){
        Doors.SetActive(true);
    }
    else{
        Doors.SetActive(false);
    }
}
private void SpawnEnemys(){

    List<GameObject> cwEnemys = new List<GameObject>();

    if(GameObject.FindGameObjectsWithTag("Enemy").Length == 0){
        WavesAreEnd = true;
    }
    else{
        WavesAreEnd = false;
    }


    if(WavesAreEnd && curentWaves < countOfWaves){
```

```
                for(int i = 0;i<countOfEnemys; i++){
                    int rangom = Random.Range(0,enemys.Count);
                    cwEnemys.Add(enemys[rangom]);
                }

                for(int i = 0;i <cwEnemys.Count;i++){
                    Vector2 position = new
Vector2(Random.Range(spawnBounds[0].x,spawnBounds[1].x),Random.Range(spawnBounds[0
].y,spawnBounds[1].y));

                    Instantiate(cwEnemys[i],position,Quaternion.identity,
gameObject.transform);
                }
                curentWaves++;
                Debug.Log(curentWaves);
            }
        }
    }
```

PlayerMovement.cs

```
using UnityEngine;

public class PlayerMovement : MonoBehaviour
{
    [Header("Movement Settings")]
    [SerializeField] private float moveSpeed = 5f;

    [Header("References")]
    [SerializeField] private Rigidbody2D rb;

    private Vector2 movement;

    private void Start()
    {
        rb = GetComponent<Rigidbody2D>();
    }

    private void Update()
    {
        movement.x = Input.GetAxisRaw("Horizontal");
        movement.y = Input.GetAxisRaw("Vertical");

        if (movement.x > 0)
            GetComponent<SpriteRenderer>().flipX = false;
        else if (movement.x < 0)
            GetComponent<SpriteRenderer>().flipX = true;
    }

    private void FixedUpdate()
    {
        rb.MovePosition(rb.position + movement.normalized * moveSpeed *
Time.fixedDeltaTime);
    }
}
```

Тестування та випробування мобільної гри для видавництва «CrazyGames»

Тестування – це процес ретельного дослідження та випробування мобільної гри. Ціль тестування, виявити розбіжності між фактичною поведінкою гри та очікуваною, за допомогою визначеного набору тестових випадків. Процес включає в себе перевірку безпеки мобільної гри, її функціональності та продуктивності.

Далі наведено тестування таких прецедентів: «Авторизація», «Налаштування профілю». Тестування проведено за методом «чорної скриньки».

Тестування прецеденту «Авторизація» за методом розбиття на класи еквівалентності.

Можливі класи еквівалентності та вхідні дані показано у таблиці 2.30.

Таблиця 2.30 – Класи еквівалентності вхідних даних

| Вхідні дані | Правильні класи | Неправильні класи |
|---|---|---|
| Логін | 1 Букви та числа | 2 Не було введено нічого<br>3 Було введено не вірний логін |
| Пароль | 4 Будь-які символи | 5 Дані не введено<br>6 Введено не вірний пароль |

Результати тестування класів представлено у таблицях 2.31 – 2.32.

Таблиця 2.31 – Тести правильних класів еквівалентності

| № тесту | Покритий клас | Вхідні умови | Результати тестування |
|---|---|---|---|
| 1 | 1 | UserLogin1 | Тест пройдено |
| | 4 | Userpassword | |

Таблиця 2.32 – Тести неправильних класів еквівалентності

| № тесту | Покритий клас | Вхідні умови | Результати тестування |
|---|---|---|---|
| 1 | 2 | Логін: | Введіть логін. |
| 2 | 3 | Login@^ | Введено неправильний логін. |
| 3 | 5 | Пароль: | Введіть пароль. |
| 4 | 6 | Incorrect password | Введено неправильний пароль. |

Тестування прецеденту «Налаштування профілю» методом варіантів використання.

Сценарій 1.

Передумова:

– Гру запущено;
– Користувач авторизувався.

Основний потік подій:

1. Натиснути кнопку налаштувань;
2. Замінити обрані значення (Ім’я, фото профілю, гучність);
3. Натиснути на кнопку підтвердження змін.

Очікувані результати:

1. Налаштування змінено;
2. Вивід повідомлення про успішне внесення змін.

Сценарій 2.

Передумова:

– Гру запущено;
– Користувач авторизувався.

Основний потік подій:

1. Натиснути кнопку налаштувань;
2. Замінити обрані значення (Ім’я, фото профілю, гучність);

3. Закрити панель без підтвердження змін.

Очікувані результати:

1. Вивід повідомлення про невдале внесення змін.

Сценарій 3.

Передумова:

– Гру запущено;

– Користувач авторизувався.

Основний потік подій:

1. Натиснути кнопку налаштувань;

2. Замінити обрані значення на некоректні (Ім’я, фото профілю, гучність);

3. Натиснути на кнопку підтвердження змін.

Очікувані результати:

1. Налаштування не змінено;

2. Вивід повідомлення про не вірні значення.

Після проєктування мобільної гри було сформовано аналіз вимог, розроблено інформаційну модель, приклади користувацького інтерфейсу та діаграму варіантів використання. До кожного варіанту використання було складено специфікації. Також до декількох прецедентів створено діаграми діяльності.

# 3 РЕЗУЛЬТАТИ РОЗРОБКИ МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА «CRAZYGAMES»

У результаті виконання кваліфікаційної роботи було створено мобільну гру для видавництва «CrazyGames». Першим етапом розробки став аналіз практичної задачі що необхідно розв’язати. Далі було проведено пошук аналогічних рішень, проведено їх аналіз з метою визначити переваги та недоліки продуктів. На основі проведеного аналізу було обґрунтовано доцільність розроби власної мобільної гри.

За результатами аналізу предметної області було сформовано постанову задачі на розробку власної гри, визначено основні функції та виявлено користувачів. Було визначено оптимальну архітектуру для проєкту.

На етапі створення ескізного проєкту розроблено діаграми діяльності для основних прецедентів, концептуальну модель даних та виконано проєктування користувацького інтерфейсу.

При технічному проєктуванні створено статичну модель у вигляді діаграми класів. Побудовано динамічну модель мобільної гри.

Під час робочого проєктування затверджено інструменти та засобу розробки. А саме мова програмування C#, рушій Unity, середовище розробки Visual Studio Code та систему керування базою даних MongoDB. Після цього відбулося кодування відповідних програмних модулів.

Вкінці було створено програму та обрано методику випробувань. Згідно з цим проведено тестування мобільної гри з метою виявлення та усунення помилок. Усі випробування надведено у додатку В.

Останнім етапом було створено детальну інструкцію мобільної гри, її наведено у додатку Г.

Технічне завдання представлено у додатку, опис мобільної гри у додатку Б, програма те методику випробувань у додатку В, інструкція для користувача у додатку Г, а код програми наведено у додатку Д.

# 4 ОХОРОНА ПРАЦІ

Охорона праці є важливим аспектом будь-якої діяльності, що забезпечує безпеку та здоров'я працівників на робочому місці. Для забезпечення ефективного управління охороною праці необхідно враховувати кілька ключових складових

## 4.1 Огляд основних законодавчих та нормативних документів про охорону праці в Україні

В Україні охорона праці є важливою складовою забезпечення безпеки на робочому місці, і її регулюють численні законодавчі та нормативні акти. Основним документом, що визначає загальні принципи та вимоги в цій сфері, є Закон України "Про охорону праці". Він встановлює обов'язки роботодавців та працівників щодо забезпечення безпечних умов праці, передбачає проведення відповідного інструктажу, навчання, а також вказує на необхідність створення відповідних умов для запобігання травмам та професійним захворюванням.

Крім того, важливе значення мають положення Кодексу законів про працю України, який регулює трудові відносини та включає розділи, присвячені охороні праці, безпеці на робочому місці, а також умовам праці на виробництві. Цей документ визначає порядок організації охорони праці, відповідальність роботодавців за порушення встановлених норм, а також процедуру вирішення трудових спорів.

Серед нормативних документів особливе місце займають Державні санітарні норми та правила охорони праці для окремих видів діяльності. Ці правила регламентують специфічні вимоги до умов праці на підприємствах різних галузей, враховуючи їхні особливості та потенційні ризики для працівників. Вони визначають допустимі норми щодо впливу шкідливих факторів, таких як шум, вібрація, хімічні речовини, а також правила безпеки під час роботи з обладнанням та машинами.

Усі ці документи разом створюють цілісну систему правового регулювання охорони праці в Україні, яка спрямована на збереження здоров'я та життя працівників, створення безпечних умов праці та забезпечення ефективного контролю за дотриманням встановлених норм.

## 4.2 Опис структури управління питаннями охорони праці на підприємстві

Ефективне управління питаннями охорони праці на підприємстві передбачає чітку організацію та функціонування спеціальних підрозділів та відповідальних осіб. Основна структура включає наступні елементи:

Служба охорони праці: Служба охорони праці є ключовим підрозділом, відповідальним за забезпечення безпеки на робочому місці. Вона зазвичай підпорядковується безпосередньо керівнику підприємства або його заступнику, що забезпечує високий рівень управлінської підтримки та відповідальності. Основні завдання служби охорони праці включають:

Проведення інструктажів для працівників з питань охорони праці, включаючи первинний, повторний та позаплановий інструктаж.

Перевірка дотримання вимог охорони праці на підприємстві, в тому числі перевірка відповідності умов праці та використання засобів індивідуального захисту.

Аналіз умов праці з метою виявлення потенційних ризиків та небезпек, а також розробка рекомендацій для їх усунення або зменшення.

Розробка заходів для покращення умов праці, які включають впровадження нових стандартів безпеки, модернізацію обладнання, а також організацію медичних оглядів працівників.

Відповідальні особи за охорону праці: Крім служби охорони праці, на підприємстві можуть бути призначені відповідальні особи, які займаються питаннями охорони праці на рівні окремих відділів або структурних підрозділів. Це можуть бути:

Інженер з охорони праці, який здійснює контроль за дотриманням технічних

норм і стандартів, проводить навчання та інструктажі, а також контролює роботу системи охорони праці.

Медичний працівник або медсестра, відповідальний за проведення медичних оглядів, моніторинг стану здоров’я працівників і надання першої медичної допомоги при нещасних випадках.

Спеціаліст з безпеки праці на виробничих ділянках, який контролює виконання інструкцій з охорони праці на конкретних робочих місцях.

Комісії з охорони праці: На підприємствах також можуть створюватися спеціальні комісії з охорони праці, до складу яких входять представники адміністрації, профспілок і трудового колективу. Основні функції таких комісій:

Розгляд і вирішення питань, що стосуються охорони праці.

Аналіз нещасних випадків та пропозиції щодо їх запобігання.

Оцінка ефективності заходів з охорони праці та підготовка рекомендацій для покращення умов праці.

Документування та звітність: Важливою складовою структури управління є систематичне документування всіх процесів та результатів діяльності з охорони праці. Це включає ведення обліку інструктажів, перевірок, медичних оглядів, а також складання звітів про стан охорони праці та виконання встановлених норм.

Організація такої структури дозволяє забезпечити системний підхід до управління охороною праці, що сприяє підвищенню безпеки та зменшенню ризиків на підприємстві, а також забезпечує виконання законодавчих вимог у цій сфері.

## 4.3Методи забезпечення техніки безпеки та охорони праці для конкретних посад

Для ефективного забезпечення техніки безпеки та охорони праці на підприємстві необхідно встановити чіткі вимоги для кожної посади або конкретного завдання. Це включає в себе наступні ключові аспекти:

Проведення інструктажів:

Інструктаж для нових працівників: Для всіх нових співробітників необхідно

провести вступний інструктаж, що включає ознайомлення з основними вимогами охорони праці, правилами безпеки, а також специфікою робочого місця. Це дозволяє новим працівникам швидше адаптуватися до робочого процесу та зрозуміти, як уникнути потенційних ризиків.

Регулярне підвищення кваліфікації: Периодичне проведення повторних інструктажів і тренінгів для працівників допомагає оновити знання про нові методи безпечної роботи та зміни в нормативних вимогах. Це також включає спеціальні тренінги з безпеки для виконання конкретних завдань або використання нового обладнання.

Забезпечення засобами індивідуального захисту (ЗІЗ):

Вибір та надання ЗІЗ: Залежно від специфіки посади або завдання, працівники повинні бути забезпечені відповідними засобами індивідуального захисту, такими як спецодяг, захисні окуляри, рукавиці, маски, захисні шоломи, навушники тощо. Важливо, щоб ЗІЗ відповідали стандартам якості та були у належному стані.

Правильне використання та обслуговування ЗІЗ: Працівники повинні бути навчені правильно використовувати та доглядати за засобами індивідуального захисту. Регулярна перевірка та заміна зношених або пошкоджених ЗІЗ є важливою частиною забезпечення безпеки.

Моніторинг стану робочого обладнання:

Регулярне технічне обслуговування: Важливо забезпечити регулярне технічне обслуговування та перевірку стану робочого обладнання, щоб запобігти можливим аваріям та поломкам. Це включає плановий огляд, заміну зношених частин, калібрування та ремонт обладнання.

Контроль за безпекою обладнання: Постійний моніторинг і контроль за станом обладнання дозволяє вчасно виявити та усунути потенційні небезпеки, що може знизити ризик аварій та травм на робочому місці.

Забезпечення техніки безпеки та охорони праці на кожній посаді та для конкретних завдань є ключовим для створення безпечного робочого середовища. Це не лише допомагає захистити здоров'я та життя працівників, але й підвищує

ефективність роботи та знижує ймовірність виникнення виробничих інцидентів.

## 4.4 Аналіз шкідливих та небезпечних факторів для конкретних посад або завдань

Аналіз шкідливих та небезпечних факторів є критично важливим етапом у забезпеченні безпеки праці на підприємстві. Цей процес включає ідентифікацію та оцінку ризиків, які можуть виникнути внаслідок виконання конкретних посадових обов'язків або завдань. Основні етапи аналізу включають:

Ідентифікація шкідливих факторів:

– Фізичні фактори: Це включає вплив шуму, вібрацій, надмірної температури, освітлення, а також небезпеки, що виникають при роботі з важким обладнанням або в умовах підвищеної вологості. Фізичні фактори можуть негативно впливати на здоров'я працівників, викликати втому, порушення слуху або зниження зору.

– Хімічні фактори: Вони включають вплив токсичних, корозійних або подразнюючих хімічних речовин. Це може бути контакт з хімікатами, пари або пил, які можуть викликати отруєння, алергічні реакції, або проблеми з дихальними шляхами.

– Біологічні фактори: Це включає вплив патогенних мікроорганізмів, бактерій, вірусів, а також контакт з небезпечними біологічними матеріалами, такими як кров або сеча. Біологічні фактори можуть спричинити інфекційні захворювання або алергічні реакції.

– Психологічні фактори: Стрес, надмірне навантаження, монотонна робота або конфлікти на робочому місці також можуть бути шкідливими для психічного здоров'я працівників, викликаючи втому, тривогу, депресію або інші проблеми.

Ідентифікація небезпечних факторів:

– Механічні небезпеки: Це включає небезпеки, пов'язані з роботою з механічним обладнанням, таким як рухомі частини машин, гострі кромки, чи

механічні помилки, що можуть призвести до травм.

– Електричні небезпеки: Це пов'язано з ризиками електричного удару, короткого замикання, або небезпеки, що виникають при роботі з електрообладнанням.

– Будівельні небезпеки: Небезпеки, що пов'язані з роботою на висоті, з важкими матеріалами, або в умовах, де є ризик обвалення конструкцій чи інших нещасних випадків.

Оцінка ризиків:

– Оцінка ймовірності і серйозності наслідків: Кожен з ідентифікованих факторів оцінюється з точки зору ймовірності його виникнення і серйозності можливих наслідків для здоров'я і безпеки працівників.

– Аналіз існуючих заходів безпеки: Перевірка, чи є вже діючі заходи для зменшення впливу шкідливих та небезпечних факторів, та їх ефективність.

Розробка заходів для зменшення впливу:

– Впровадження контролюючих заходів: Запровадження нових процедур, інструкцій, і технічних засобів для зменшення або усунення шкідливих і небезпечних факторів. Це може включати покращення вентиляції, встановлення захисних огороджень, або використання засобів індивідуального захисту.

– Навчання та підвищення кваліфікації: Проведення навчання для працівників з питань безпеки, включаючи правильне використання обладнання, процедури екстреного реагування та інші заходи для зменшення ризиків.

Аналіз шкідливих та небезпечних факторів дозволяє визначити та впровадити ефективні заходи для забезпечення безпеки праці, що сприяє зменшенню ризиків травм і захворювань, підвищенню рівня захисту працівників та загальної безпеки на підприємстві.

## 4.5 Розробка заходів для зменшення дії шкідливих факторів

Після проведення аналізу ризиків та виявлення шкідливих і небезпечних факторів на робочому місці, необхідно розробити та впровадити конкретні заходи,

які допоможуть мінімізувати їх негативний вплив на працівників. Ось основні категорії заходів, які можуть бути реалізовані:

Зміна умов праці:

– Вентиляція: Встановлення або покращення системи вентиляції для забезпечення належного повітрообміну, що дозволяє зменшити концентрацію шкідливих газів, пилу та інших забруднюючих речовин в атмосфері робочого приміщення. Це може включати установку витяжних систем, очищувачів повітря або регулярне провітрювання.

– Освітлення: Забезпечення достатнього рівня освітлення на робочих місцях для зменшення навантаження на зір і уникнення зорового перевантаження. Це може включати встановлення відповідних освітлювальних приладів, регулювання рівня освітленості та забезпечення хорошого розподілу світла.

– Температура: Регулювання температурного режиму в приміщеннях, де виконуються роботи, щоб забезпечити комфортні умови праці. Це може включати встановлення систем кондиціонування або обігріву, а також забезпечення адекватного рівня вологості.

Зменшення часу впливу шкідливих факторів:

– Скорочення робочого дня: У разі, якщо умови праці є надзвичайно шкідливими, можна розглянути можливість скорочення тривалості робочого дня для зменшення часу, протягом якого працівники піддаються впливу шкідливих факторів.

– Ротація працівників: Запровадження системи ротації для працівників, що дозволяє змінювати їх робочі місця або завдання, щоб уникнути тривалого впливу одних і тих же шкідливих факторів. Це дозволяє зменшити загальне навантаження та знизити ризики професійних захворювань.

Використання захисних бар'єрів та екранування:

– Захисні бар'єри: Установка фізичних бар'єрів або огороджень, які можуть захистити працівників від небезпечних частин обладнання, гарячих поверхонь, хімічних речовин або інших потенційних небезпек.

– Екранування: Використання захисних екранів або панелей для

обмеження прямого контакту працівників з шкідливими факторами, такими як радіація, пил або рідини. Це може включати встановлення прозорих захисних перегородок або екранів, які знижують ризики впливу.

Організація додаткових перерв або відпочинку:

– Додаткові перерви: Організація регулярних перерв у роботі для працівників, які працюють в умовах підвищеного ризику, щоб дозволити їм відновити сили і зменшити вплив шкідливих факторів. Це може включати короткі перерви на відпочинок або зміни виду діяльності.

– Відпочинок: Забезпечення комфортних зон для відпочинку, де працівники можуть зняти стрес і відпочити від шкідливих умов праці. Це включає облаштування кімнат для відпочинку, обладнаних необхідними зручностями.

Розробка і впровадження цих заходів допоможе створити більш безпечне і комфортне робоче середовище, зменшити ризики травм і захворювань, а також підвищити загальну ефективність і продуктивність праці.

# ВИСНОВКИ

У результаті виконання цієї кваліфікаційної роботи було розв’язано задачу інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов, а саме розроблено мобільну гру для видавництва «CrazyGames».

Для досягнення мети було розв’язано наступні задачі:

– проведено аналіз практичної задачі інженерії програмного забезпечення;

– проведено аналіз та опис предметної області та аналіз готових рішень мобільної гри зі схожою тематикою;

– проведено обґрунтування доцільності розробки мобільної гри для видавництва «CrazyGames»;

– сформульовано вимоги розроблюваної мобільної гри та створено технічне завдання на розробку мобільної гри для видавництва «CrazyGames»;

– обрано оптимальний архітектурний стиль, згідно з яким створено архітектуру мобільної гри;

– створено ескізний, технічний та робочий проєкти мобільної гри для видавництва «CrazyGames»;

– написано документацію до мобільної гри.

Результатом розробки стала мобільна гра яка задовольняє потреби видавництва «CrazyGames» та буде опублікована.

СПИСОК ВИКОРИНСТИХ ДЖЕРЕЛ

1. Мобільна гра «Soul Knight» URL: https://play.google.com/store/apps/details?id=com.ChillyRoom.DungeonShooter&hl=ru (дата звернення 18.05.2025).

2. Мобільна гра «Survivor.io» URL: https://play.google.com/store/apps/details?id=com.dxx.firenow&hl=ru (дата звернення 18.05.2025).

3. Комп’ютерна гра «Enter the Gungeon» URL: https://store.steampowered.com/app/311690/Enter_the_Gungeon/ (дата звернення 20.05.2025).

4. Про охорону праці» від 14.10.1992 № 2694 URL: https://zakon.rada.gov.ua/laws/show/2694-12 (дата звернення 21.05.2025).

# ДОДАРОК А – ТЕХНІЧНЕ ЗАВДАННЯ МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА «CRAZYGAMES»

Вступ

Повна назва розроблюваного проєкту: «Розробка мобільної гри для видавництва «CrazyGames».

Галузь застосування: Публікація на різних мобільних платформах з дистрибуції ігор.

## 1. Підстава для розробки

Підставою для розробки гри стало завдання на кваліфікаційну роботу на здобуття ступеня вищої освіти «бакалавр».

## 2. Призначення розробки

### 2.1 Функціональне призначення

Функціональним призначенням мобільної гри є забезпечення користувачів приємним ігровим досвідом, задовольняє їх потреби. Також не менш важливим функціональним призначенням є подальша монетизація гри задля отримання прибутку.

### 2.2 Експлуатаційне призначення

Експлуатаційним призначенням гри є надання користувачам можливості приємно провести час, отримати емоції та спілкуватися з іншими користувачами.

## 3. Вимоги до програмного забезпечення

### 3.1 Вимоги до складу виконуваних функцій

Користувач

Користувач повинен мати доступ до функцій гри.

- Збереження прогресу (Гравець може зберігати прогрес у грі та продовжувати проходження з того місця, де зупинився);
- Доступ до повної версії гри (Користувач отримує доступ до всіх рівнів, персонажів і можливостей гри);
- Редагування профілю (Можливість змінювати ім'я, аватар);
- Система досягнень (Гравець може переглядати свої досягнення та нагороди);
- Взаємодія з іншими гравцями (Гра підтримує онлайн-режим за допомогою локального серверу, користувач може змагатися з друзями або грати разом);
- Вибір ігрового режиму (Гравець повинен мати змогу обирати потрібний йому ігровий режим);
- Взаємодія з ворогами (Гравець повинен мати змогу завдавати шкоди ворогам в процесі проходження гри);
- Взаємодія з предметами (Гравець повинен мати змогу піднімати, викидати, руйнувати та використовувати певні предмети на ігровому рівні);
- Купівля платного контенту (Доступ до магазину, де можна придбати додаткові предмети, персонажів або бонуси за реальні гроші);
- Оновлення та події (Отримання сповіщень про нові оновлення, ігрові події та акції).

Адміністратор (розробник гри)

Адміністратор – це особа або група осіб, які мають доступ до внутрішніх налаштувань гри та керують її функціоналом.

Функції для адміністратора:

- Управління користувачами (Адміністратор може блокувати або розблоковувати акаунти, переглядати звіти про порушення правил гри);
- Оновлення контенту (Додавання нових рівнів, персонажів, предметів, балансування ігрового процесу);
- Аналіз статистики (Перегляд активності користувачів, їхніх досягнень, популярності рівнів і загальної ефективності гри);
- Керування внутрішньоігровими покупками (Налаштування цін, знижок, акцій та перевірка коректної роботи платіжних систем);
- Моніторинг продуктивності гри (Контроль за відсутністю технічних збоїв, виправлення помилок та забезпечення стабільної роботи).

## 3.2 Вимоги до організації вхідних даних

Вхідні дані

Вхідні дані -це інформація, яку система отримує від користувача або адміністратора для забезпечення ігрового процесу, взаємодії з сервером та внутрішньоігрових операцій.

Для користувача:

- Збереження прогресу (ID користувача, Поточний рівень, Досягнення, Статистика, Збережені налаштування);
- Редагування профілю (Ім’я користувача, Аватар, Налаштування звуку та управління);
- Запит на отримання списку досягнень (ID користувача);
- Взаємодія з іншими гравцями (Запити на додавання друзів, Запити на спільну гру, Статус мережевого з’єднання);

- Купівля платного контенту (ID користувача, ID товару, Сума покупки, Платіжна інформація);
- Отримання оновлень та повідомлень (ID користувача, Час останнього входу, Налаштування сповіщень).
-

Для адміністратора:

- Управління користувачами (ID користувача, Статус акаунта, Скарги, Дії модератора);
- Додавання та редагування контенту (Назва рівня, Опис, Графічні ресурси, Складність, Нагороди);
- Аналіз статистики (Кількість активних гравців, Час у грі, Популярність рівнів, Кількість покупок);
- Налаштування внутрішньоігрових покупок (Назва товару, Опис, Ціна, Доступність, Акційні пропозиції);
- Моніторинг продуктивності (Логи помилок, Час завантаження рівнів, Відгуки користувачів).

Вихідні дані

Вихідні дані -це інформація, яку система формує та надає користувачам або адміністраторам на основі отриманих вхідних даних.

Для користувача:

- Ігровий інтерфейс та статус (Поточний рівень, Доступні персонажі, Очки досвіду, Список завдань)
- Система досягнень (Перелік досягнень, Отримані нагороди, Прогрес до наступного рівня)
- Статус купленого контенту (Доступність куплених предметів, Залишок ігрової валюти)
- Оновлення та події (Оновлення контенту, Акції, Тимчасові події)

Для адміністратора:

- Список користувачів (ID користувача ,Ім’я користувача, Статус акаунта, Дата останнього входу)
- Статистика гри (Кількість активних гравців, Популярність режимів, Найчастіші покупки)
- Звіти про помилки (Тип помилки, Лог-файли, Дата та час збою)
- Фінансові звіти (Загальний дохід, Найпопулярніші покупки,)

Примітка

В процесі роботи над проєктом перелік вхідних та вихідних даних може змінюватися залежно від потреб, особливостей реалізації функцій та необхідності додаткових параметрів. Представлений список є базовим і може бути доповнений або скорочений на етапах тестування та випуску оновлень.

## 3.3 Вимоги до надійності

### 3.3.1 Вимоги до забезпечення надійності функціонування мобільної гри

Основними вимогами для забезпечення функціонування гри є використання справного мобільного телефону та стабільне з’єднання з інтернетом.

### 3.3.2 Час відновлення після відмови

Час відновлення після відмови у деяких випадках напряму залежить від справності технічного засобу користувача та його з’єднання з інтернетом. Проте у випадках фатальних збоїв серверу час відновлення не повинен перевищувати 10 хвилин.

## 3.4 Вимоги до умов експлуатації

### 3.4.1 Кліматичні умови експлуатації

Умови експлуатації гри відповідають умовам пристроїв на яких завантажена та використовується гра.

### 3.4.2 Вимоги до чисельності та кваліфікованості користувачів

Необхідний рівень підготовки користувачів: мінімальні навички володіння мобільним телефоном.

Необхідний рівень підготовки адміністратору: Повна кваліфікація у сфері адміністрування ігор, знання принципів роботи баз даних та мережевих технологій для налаштування, моніторингу статистики.

### 3.5 Вимоги до інформаційної та програмної сумісності

Мінімальні вимоги інформаційної і програмної сумісності серверу

- ОС: Windows 10 Linux або Ubuntu 20.10;
- СУБД MySQL 8.0.

Мінімальні вимоги до інформаційної та програмної сумісності пристрою користувача

- ОС: Android 10 або IOS 14;
- Обсяг потрібної пам’яті 300Мб.

## 4. Вимоги до програмної документації

Програмна документація повинна містити:

- технічне завдання;
- керівництво користувача;
- програма та методика випробувань;
- опис програмного забезпечення;
- код програми.

## 5. Техніко-економічні показники

Техніко-економічні показники не розраховувались

## 6. Стадії та етапи розробки

Стадії та етапи розробки ПЗ наведено у таблиці А.1.

Таблиця А.1 – Стадії та етапи розробки ПЗ

| № | Стадії та етапи | Термін виконання |
|---|---|---|
| 1 | Аналіз предметної області | 07.04.2025 |
| 2 | Аналіз вимог до ПЗ | 14.04.2025 |
| 3 | Розробка архітектури ПЗ | 21.04.2025 |
| 4 | Розробка проєкту ПЗ | 05.05.2025 |
| 5 | Реалізація та тестування ПЗ | 12.05.2025 |

## 7. Порядок контролю і приймання

Контроль за розробкою кожної частини гри на кожному етапі повинен виконуватись з урахуванням попередньо визначених у технічному завданні вимог. Кожна стадія розробки повинна бути виконана та представлена у визначені строки та узгоджена із замовником.

Прийом виконується згідно програми і методики випробувань і документується за допомогою протоколу проведення випробувань. У випадку виявлення недоліків під час прийому складається відповідний акт про знаходження помилки, який підписується представниками обох сторін. Розробник повинен не більш ніж за два тижні представити виплавлену версію гри та оповістити замовника про повторне проходження процедури перевірки.

# ДОДАТОК Б – ОПИС МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА «CRAZYGAMES»

## 1. Загальні відомості

Найменування: Мобільна гра для видавництва «CrazyGames».

Для роботи мобільної гри потрібні:

– Мобільний телефон з операційною системою Android 10 або вище;

– Мобільний телефон з операційною системою iOS 14 або вище.

Розроблюване програмне забезпечення було створене з використанням рушія Unity та мови програмування C#.

## 2. Функціональне призначення розробки

Функціональним призначенням мобільної гри є надання користувачам якісного ігрового досвіду та відпочинку. Гра також сприятиме комунікації між користувачами за рахунок багатокористувацьких режимів. У подальшому цей програмний продукт може стати джерелом доходу для розробника та видавництва, так яка у грі присутня реклама та внутрішньоігрові покупки.

Мобільна гра повинна містити такий функціонал:

1. Авторизація у грі.

Вхідні дані:

– Логін;

– Пароль.

Вихідні дані:

– Результат авторизації.

2. Реєстрація у грі.

Вхідні дані:

– Логін;

– Пароль.

Вихідні дані:

– Результат реєстрації.

3. Налаштування профілю користувача.

Вхідні дані:

– Ім’я;

– Гучність;

– Фото профілю.

Вихідні дані:

– Результат налаштування профілю.

4. Вибір ігрового режиму.

Вхідні дані:

– Ідентифікатор режиму;

– Складність.

Вихідні дані:

– Початок гри у обраному режимі.

5. Додавання друзів користувача.

Вхідні дані:

– Ідентифікатор користувача.

Вихідні дані:

– Результат додавання друзів у список.

6. Обробка звітів адміністратором (створення, редагування, видалення).

Вхідні дані:

– Ідентифікатор звіту;

– Текст звіту;

– Тип звіту.

Вихідні дані:

– Результат обробки звітів.

3. Опис логічної структури

Мобільна гра розроблена за шаблоном MVC, він в свою чергу містить 3 основні модулі.

Модуль Model відповідає за роботу з ігровою логікою (він містить класи роботи з базою даних, генерації рівнів, виклику реклами, обробки покупок, створення ворогів та предметів).

Модуль View відповідає за роботу з користувацьким інтерфейсом (містить класи переходу між сценами, відображення елементів інтерфейсу та форм оплати).

Модуль Controller відповідає за обробку дій гравця та взаємодію його з ігровим середовищем (містить класи керування гравцем, взаємодії гравця за предметами та ворогами).

4. Технічні засоби

Для стабільного функціонування гри необхідні такі засоби з мінімальними параметрами:

- CPU: MediaTek Helio G85 - 2.0 Мгц;
- GPU: ARM Mali-G52 MP2;
- RAM: 4 ГБ;
- SD: 200MB вільного місця.

5 Виклик та завантаження

Завантаження мобільної гри відбувається на сервісах «PlayMarket» та «AppStore». Виклик відбувається безпосередньо на пристрої користувача шляхом натискання відповідної іконки.

6 Вхідні дані

Вхідні дані - це інформація, яку система отримує від користувача або адміністратора для забезпечення ігрового процесу, взаємодії з сервером та внутрішньоігрових операцій.

Для користувача:

– Збереження прогресу (ID користувача, Поточний рівень, Досягнення, Статистика, Збережені налаштування);

– Редагування профілю (Ім’я користувача, Аватар, Налаштування звуку та управління);

– Запит на отримання списку досягнень (ID користувача);

– Взаємодія з іншими гравцями (Запити на додавання друзів, Запити на спільну гру, Статус мережевого з’єднання);

– Купівля платного контенту (ID користувача, ID товару, Сума покупки, Платіжна інформація);

– Вибір ігрового режиму (Гравець повинен мати змогу обирати потрібний йому ігровий режим);

– Взаємодія з ворогами (Гравець повинен мати змогу завдавати шкоди ворогам в процесі проходження гри);

– Взаємодія з предметами (Гравець повинен мати змогу піднімати, викидати, руйнувати та використовувати певні предмети на ігровому рівні);

– Отримання оновлень та повідомлень (ID користувача, Час останнього входу, Налаштування сповіщень).

Для адміністратора:

– Управління користувачами (ID користувача, Статус акаунта, Скарги, Дії модератора);

– Додавання та редагування контенту (Назва рівня, Опис, Графічні ресурси, Складність, Нагороди);

– Аналіз статистики (Кількість активних гравців, Час у грі, Популярність рівнів, Кількість покупок);

– Налаштування внутрішньоігрових покупок (Назва товару, Опис, Ціна, Доступність, Акційні пропозиції).

7. Вихідні дані

Вихідними даними є інформація отримана у результаті виконання операцій під час гри.

Для користувача:

– Ігровий інтерфейс та статус (Поточний рівень, Доступні персонажі, Очки досвіду, Список завдань);

– Система досягнень (Перелік досягнень, Отримані нагороди, Прогрес до наступного рівня);

– Статус купленого контенту (Доступність куплених предметів, Залишок ігрової валюти);

– Оновлення та події (Оновлення контенту, Акції, Тимчасові події).

Для адміністратора:

– Список користувачів (ID користувача ,Ім’я користувача, Статус акаунта, Дата останнього входу);

– Статистика гри (Кількість активних гравців, Популярність режимів, Найчастіші покупки);

– Звіти про помилки (Тип помилки, Лог-файли, Дата та час збою);

– Фінансові звіти (Загальний дохід, Найпопулярніші покупки).

# ДОДАТОК В – ПРОГРАМА ТА МЕТОДИКА ВИПРОБУВАНЬ МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА «CRAZYGAMES»

## 1. Об’єкт випробувань

Об’єктом випробувань є мобільна гра для видавництва «CrazyGames».

Сфера використання:

Розваги та забезпечення гравців якісним ігровим досвідом. Сприяння комунікації при кооперативному проходженні.

## 2. Мета випробування

Метою є перевірка відповідності мобільної гри зазначеним вимогам, котрі містить технічне завдання наведене у додатку А.

## 3. Вимоги до мобільної гри

Під час випробування потрібно впевнитись у коректній роботі всіх описаних компонентів мобільної гри та їх відповідності функціональним вимогам, що визначені у технічному завданні.

## 4. Вимоги до програмної документації

Програмна документація повинна включати в себе:

– Технічне завдання;
– Опис програми;
– Програму та методику випробувань;
– Інструкцію користувача;
– Код програми.

## 5. Засоби та порядок випробувань

Програмні засоби випробувань

До програмних засобів випробувань для випробувань належить операційна система Android або iOS.

Апаратні засоби випробувань

Мобільний телефон з компонентами комплектації вище за:

– CPU: MediaTek Helio G85 - 2.0 Мгц;
– GPU: ARM Mali-G52 MP2;
– RAM: 4 ГБ;
– SD: 200MB вільного місця.

Порядок проведення випробувань

Проведення випробувань повинно відповідати наступному порядку:

– Виконуються інструкції до кожної функції;
– Робиться аналіз оптимальних результатів.

При виявленні помилок складається їх перелік і узгоджується термін їх виправлення.

6. Методика випробувань

Випробування проводиться за методикою «чорної скриньки». Через великий об’єм випробуваних функцій представимо тільки декілька ключових функцій мобільної гри. Програму випробувань наведено у таблиці В.1.

Таблиця В.1 – Програма випробувань.

| № | Дія | Очікуваний результат |
|---|---|---|
| 1 | Авторизація з коректними даними. | Користувач опиняється у головному меню гри. |
| 2 | Налаштування профілю з коректними даними. | Зміни збережено та користувач опиняється у головному меню. |
| 3 | Вибір режиму гри. | Користувач починає гру у вибраному режимі. |
| 4 | Додавання друзів з коректними даними. | Користувача з відповідним ідентифікатором додано до списку друзів. |

Кінець таблиці В.1.

| № | Дія | Очікуваний результат |
|---|---|---|
| 5 | Покупка внутрішньоігрового контенту з коректними даними. | Користувач підтвердив оплату та обраний внутрішньігровий товар додано у його інвентар. |
| 6 | Перегляд реклами за винагороду | Користувач завершує перегляд та отримує винагороду |

# ДОДАТОК Г – ІНСТРУКЦІЯ КОРИСТУВАЧА МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА «CRAZYGAMES»

## Функціональне призначення

Функціональним призначенням мобільної гри є надання користувачам якісного ігрового досвіду та можливості для повноцінного відпочинку, розваги й емоційного розвантаження у вільний час. Вона створена з метою залучення гравців до захопливого ігрового процесу, що поєднує інтуїтивно зрозумілий інтерфейс, привабливий візуальний стиль та зручне керування. Крім цього, гра сприятиме активній взаємодії та комунікації між користувачами завдяки наявності багатокористувацьких режимів. У подальшому розвитку цей програмний продукт має потенціал стати не лише засобом розваги, а й стабільним джерелом доходу як для індивідуального розробника, так і для студії чи видавництва. Це досягається завдяки інтеграції реклами, внутрішньоігрових покупок.

## Експлуатаційне призначення

Експлуатаційне призначення мобільної гри полягає в забезпеченні користувачам зручного, доступного та стабільного доступу до інтерактивного ігрового процесу на мобільних пристроях у будь-який час і в будь-якому місці. Гра призначена для регулярного використання в умовах обмеженого часу або під час відпочинку, забезпечуючи швидкий запуск, оптимізовану продуктивність та комфортну взаємодію з інтерфейсом.

Мобільна гра виконує наступні функції:

1. Авторизація у грі.

Вхідні дані:

– Логін;

– Пароль.

Вихідні дані:

– Результат авторизації.

2. Реєстрація у грі.

Вхідні дані:

– Логін;

– Пароль.

Вихідні дані:

– Результат реєстрації.

3. Налаштування профілю користувача.

Вхідні дані:

– Ім’я;

– Гучність;

– Фото профілю.

Вихідні дані:

– Результат налаштування профілю.

4. Вибір ігрового режиму.

Вхідні дані:

– Ідентифікатор режиму;

– Складність.

Вихідні дані:

– Початок гри у обраному режимі.

5. Додавання друзів користувача.

Вхідні дані:

– Ідентифікатор користувача.

Вихідні дані:

– Результат додавання друзів у список.

6. Обробка звітів адміністратором (створення, редагування, видалення).

Вхідні дані:

– Ідентифікатор звіту;
– Текст звіту;
– Тип звіту.

Вихідні дані:

– Результат обробки звітів.

Умови виконання програми

Для стабільного функціонування гри необхідні такі засоби з мінімальними параметрами:

– CPU: MediaTek Helio G85 - 2.0 Мгц;
– GPU: ARM Mali-G52 MP2;
– RAM: 4 ГБ;
– SD: 200MB вільного місця.

Для завантаження та користування мобільною грою користувач повинен володіти базовими навичками пошуку інформації, навичками володіння смартфоном та навичками роботи з інтерфейсом. Користувач повинен бути здатен самостійно знайти продукт на сервісах «PlayMarket» та «AppStore», знати як встановлювати та видаляти мобільні ігри і виконувати базові налаштування.

Ці вимоги дозволяють впевнитися в тому що гравець зможе користуватися грою та отримати бажаний результат.

Виконання програми

1. Авторизація та реєстрація

У разі відсутності облікового запису необхідно заповнити реєстраційну форму, вказавши деякі дані, такі як логін користувача та пароль, після чого підтвердити створення акаунта. Якщо обліковий запис вже існує, користувач має виконати авторизацію, ввівши свої облікові дані для входу до гри.

2. Вибір режиму

Після входу до системи користувач повинен перейти до головного меню та обрати бажаний режим гри (одиночний або багатокористувацький). Залежно від вибору, гравець або розпочне гру наодинці, або підключиться до сесії з іншими користувачами для спільної гри в режимі реального часу.

3. Отримання винагороди

Для отримання додаткових бонусів користувач повинен переглянути коротке рекламне відео, яке запускається за запитом у відповідному розділі гри. Після повного перегляду реклами йому автоматично нараховуються вказані винагороди.

4. Налаштування профілю

Користувач повинен перейти до розділу налаштувань профілю, де він може змінити особисту інформацію, таку як ім’я користувача, аватар, та тип керування.

5. Прийняти запрошення у друзі

Користувач повинен перейти до розділу запитів у друзі, де відображаються вхідні запрошення від інших гравців, переглянути список запрошень та натиснути кнопку підтвердження, щоб додати обраного користувача до списку друзів та отримати можливість спільної гри з ним у багатокористувацьких режимах.

6. Надіслати запрошення у друзі

Користувач повинен відкрити розділ пошуку друзів, унікальний ідентифікатор в поле пошуку, після чого переглянути результати та надіслати запит на додавання у друзі вибраному гравцеві для подальшої взаємодії у грі.

7. Здійснення покупки

Користувач повинен перейти до внутрішньоігрового магазину, обрати бажаний товар або пакет (наприклад, ігрову валюту чи косметичні елементи), після чого підтвердити покупку, обравши зручний спосіб оплати та дотримуючись інструкцій платіжної системи. Після успішної оплати придбаний контент автоматично додається до облікового запису користувача.

# ДОДАТОК Д – КОД МОБІЛЬНОЇ ГРИ ДЛЯ ВИДАВНИЦТВА «CRAZYGAMES»

Код наведено нижче у лістингах 1 – 6.

## Лістинг 1 – ConfigurateRoom.cs

```
using System;
using System.Collections;
using System.Collections.Generic;
using Unity.VisualScripting;
using UnityEngine;
using UnityEngine.Tilemaps;

public class ConfigurateRoom : MonoBehaviour
{
    [Header("Replacing Tiles Settings")]
    [SerializeField] private List<ReplacingData> repData = new List<ReplacingData>();
    [SerializeField] private Tilemap tilemap;

    [Header("Wall Settins")]
    [SerializeField] public GameObject[] Walls;

    [Serializable]
    public struct ReplacingData
    {
        public string TileName;
        public GameObject Prefab;
        public List<Vector3Int> position;
    }

    private void Start()
    {
        ReplaceTile();
    }

    public void OffWall(string key)
    {

        //Debug.Log(gameObject.name + " " + key);
        switch (key)
```

```
            {

                case "UP":
                    Walls[1].gameObject.SetActive(false);
                    break;

                case "BOTTOM":
                    Walls[0].gameObject.SetActive(false);
                    break;

                case "LEFT":
                    Walls[2].gameObject.SetActive(false);
                    break;

                case "RIGHT":
                    Walls[3].gameObject.SetActive(false);
                    break;
            }
        }

        public void ReplaceTile()
        {

            BoundsInt bounds = tilemap.cellBounds;
            for (int i = 0; i < repData.Count; i++)
            {
                for (int x = bounds.xMin; x < bounds.xMax; x++)
                {
                    for (int y = bounds.yMin; y < bounds.yMax; y++)
                    {
                        Vector3Int pos = new Vector3Int(x, y, 0);
                        TileBase tile = tilemap.GetTile(pos);

                        if (tile is Tile tileObj && tileObj.sprite.name ==
repData[i].TileName)
                        {
                            repData[i].position.Add(pos);
                        }
                    }
                }
            }

            for (int i = 0; i < repData.Count; i++)
```

```
            {
                foreach (Vector3Int pos in repData[i].position)
                {
                    Vector3 worldPos = tilemap.GetCellCenterWorld(pos);
                    Instantiate(repData[i].Prefab,                    worldPos,
Quaternion.identity);
                    tilemap.SetTile(pos, null); // Удаляем тайл
                }
            }
        }
    }
```

Лістинг 2 – DungonGeneroator.cs

```
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using Unity.VisualScripting;
using UnityEngine;

public class DungonGeneroator : MonoBehaviour
{
    [SerializeField] private GameObject[] _rooms;
    [SerializeField] private int _roomsCount;

    [SerializeField]  private  Dictionary<string  ,Vector3>  _spawnDir  =  new
Dictionary<string, Vector3>(){
            {"UP", new Vector3(0,40,0)},
            {"BOTTOM", new Vector3(0,-40,0)},
            {"LEFT", new Vector3(40,0,0)},
            {"RIGHT",new Vector3(-40,0,0)}
    };
    [SerializeField]   private   List<GameObject>   SpawnedRooms   =   new
List<GameObject>();

    private void Start()
    {
        GenerateDungeon();
    }

    private void GenerateDungeon()
    {
```

```
            GameObject Room = Instantiate(_rooms[Random.Range(0, _rooms.Length)],
new Vector3(0, 0, 0), Quaternion.identity);
            SpawnedRooms.Add(Room);

            Dictionary<string,Vector3> dirCopy;

            while (SpawnedRooms.Count < _roomsCount)
            {
                int randomRoom = Random.Range(0, _rooms.Length);
                dirCopy = new Dictionary<string, Vector3>(_spawnDir);

                GameObject      mainRoom      =      SpawnedRooms[Random.Range(0,
SpawnedRooms.Count)];

                bool isField;
                while (true)
                {
                    if(dirCopy.Count == 0){break;}
                    isField = false;
                    string                        randomDir                        =
dirCopy.Keys.ElementAt(Random.Range(0,dirCopy.Count));

                    for (int i = 0; i < SpawnedRooms.Count; i++)
                    {
                        if (mainRoom.transform.position + dirCopy[randomDir] ==
SpawnedRooms[i].transform.position)
                        {
                            dirCopy.Remove(randomDir);
                            isField = true;
                            break;
                        }
                    }
                    if (isField) { continue; }
                        Room         =         Instantiate(_rooms[Random.Range(0,
_rooms.Length)],       mainRoom.transform.position       +       dirCopy[randomDir],
Quaternion.identity);
                        CoridorDir(Room, mainRoom,randomDir);
                        SpawnedRooms.Add(Room);
                    break;

                }
            }
        }
```

```
        private void CoridorDir(GameObject RoomObj,GameObject mainRoomObj, string
randomDir){
            RoomObj.GetComponent<ConfigurateRoom>().OffWall(randomDir);
            switch(randomDir){
                case "LEFT":

mainRoomObj.GetComponent<ConfigurateRoom>().OffWall("RIGHT");
                break;

                case "RIGHT":
                    mainRoomObj.GetComponent<ConfigurateRoom>().OffWall("LEFT");
                break;

                case "UP":

mainRoomObj.GetComponent<ConfigurateRoom>().OffWall("BOTTOM");
                break;

                case "BOTTOM":
                    mainRoomObj.GetComponent<ConfigurateRoom>().OffWall("UP");
                break;
            }
        }
    }
```

Лістинг 3 – Peaks.cs

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

public class Peaks : MonoBehaviour
{
    private Animator anim;
    private bool playerEnter;
    [SerializeField] private string targetTag;
    [SerializeField] private int damage;

    private void Start()
    {
        anim = GetComponent<Animator>();
    }
```

```
    private void OnTriggerEnter2D(Collider2D collision)
    {
        if(collision.gameObject.tag == targetTag){
            anim.SetBool("Active", true);
            playerEnter = true;
        }
    }
    private void OnTriggerExit2D(Collider2D collision)
    {
        if(collision.gameObject.tag == targetTag){
            playerEnter = false;
        }
    }

    public void Hit(){
        if(playerEnter){
            Player.Instance.TakeDamage(damage);
            anim.SetBool("Active", false);
        }
        else{
            anim.SetBool("Active", false);
        }
    }
}
```

Лістинг 4 – Player.cs

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

public class Player : MonoBehaviour
{
    public static Player Instance { get; private set; }

    [Header("Stats")]
    [SerializeField] private float hp;
    [SerializeField] private float mp;
    [SerializeField] private float sp;

    [Header("UI")]
    [SerializeField] private GameObject uiDeathPanel;
    [SerializeField] private Image uiHP;
```

```
    void Awake()
    {
        if (!Instance){
            Instance = this;
            DontDestroyOnLoad(this);
        }
        else{
            Destroy(gameObject);
        }
    }
    private void Start()
    {
        uiHP.fillAmount = hp/100;
    }

    public void Update(){
        if (Input.GetMouseButtonDown(0)){
            TakeDamage(10);
        }
    }

    public void TakeDamage(int damage){
        if((hp - damage) > 0 ){
            hp -= damage;
            uiHP.fillAmount = hp/100;
            Debug.Log(hp/100);
        }
        else{
            hp = 0;
            uiHP.fillAmount = hp/100;
            Death();
        }
    }

    private void Death(){
        uiDeathPanel.SetActive(true);
    }
}
```

Лістинг 5 – PlayerMovement.cs

```
using UnityEngine;

public class PlayerMovement : MonoBehaviour
{
    [Header("Movement Settings")]
    [SerializeField] private float moveSpeed = 5f;

    [Header("References")]
    [SerializeField] private Rigidbody2D rb;

    private Vector2 movement;

    private void Start()
    {
        rb = GetComponent<Rigidbody2D>();
    }

    private void Update()
    {
        movement.x = Input.GetAxisRaw("Horizontal");
        movement.y = Input.GetAxisRaw("Vertical");

        if (movement.x > 0)
            GetComponent<SpriteRenderer>().flipX = false;
        else if (movement.x < 0)
            GetComponent<SpriteRenderer>().flipX = true;
    }

    private void FixedUpdate()
    {
        rb.MovePosition(rb.position + movement.normalized * moveSpeed *
Time.fixedDeltaTime);
    }
}
```

Лістинг 6 – EnemySpawner.cs

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

```
public class EnemySpawner : MonoBehaviour
{
    [SerializeField] private List<GameObject> enemys = new List<GameObject>();
    [SerializeField] private GameObject Doors;
    [SerializeField, Range(1,5)] private int countOfWaves;
    [SerializeField, Range(1,8)] private int countOfEnemys;
    [SerializeField] private Vector2[] spawnBounds;
    private bool WavesAreEnd;
    private int curentWaves;

    void Start()
    {
        for (int i = 0; i < spawnBounds.Length; i++)
        {
            spawnBounds[i] += (Vector2)transform.position;
        }
    }

    private void OnTriggerStay2D(Collider2D other) {
        if(other.gameObject.tag == "Player"){
            SpawnEnemys();
            DoorsController();
        }
    }

    private void DoorsController(){
        if(!WavesAreEnd){
            Doors.SetActive(true);
        }
        else{
            Doors.SetActive(false);
        }
    }
    private void SpawnEnemys(){

        List<GameObject> cwEnemys = new List<GameObject>();

        if(GameObject.FindGameObjectsWithTag("Enemy").Length == 0){
            WavesAreEnd = true;
        }
        else{
            WavesAreEnd = false;
        }
```

```
            if(WavesAreEnd && curentWaves < countOfWaves){
                for(int i = 0;i<countOfEnemys; i++){
                    int rangom = Random.Range(0,enemys.Count);
                    cwEnemys.Add(enemys[rangom]);
                }

                for(int i = 0;i <cwEnemys.Count;i++){
                    Vector2 position = new
Vector2(Random.Range(spawnBounds[0].x,spawnBounds[1].x),Random.Range(spawnBounds[0
].y,spawnBounds[1].y));

                    Instantiate(cwEnemys[i],position,Quaternion.identity,
gameObject.transform);
                }
                curentWaves++;
                Debug.Log(curentWaves);
            }
        }
    }
```