МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Харківський національний університет радіоелектроніки

Факультет комп’ютерних наук (або центр післядипломної освіти, або навчально-науковий центр заочної форми навчання)

(повна назва)

Кафедра програмної інженерії

(повна назва)

**КОМПЛЕКСНИЙ КУРСОВИЙ ПРОЄКТ**

**Пояснювальна записка**

рівень вищої освіти перший (бакалаврський)

(тема)

Виконав:

здобувач (ка) 3 курсу, групи ПЗПІ-22-4

В’ячеслав ГЕРЦЕВ

(Власне ім’я, ПРІЗВИЩЕ)

Спеціальність 121 – Інженерія програмного\_ \_\_

забезпечення

(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Програмна інженерія

(повна назва освітньої програми)

Керівник доцент кафедри ПІ Наталя ВАЛЕНДА

(посада, Власне ім’я, ПРІЗВИЩЕ)

|  |
| --- |
| Члени комісії (Власне ім’я, ПРІЗВИЩЕ, підпис) |
|  |
|  |
|  |

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерних наук (або центр післядипломної освіти, або навчально-науковий центр заочної форми навчання)

Кафедра програмної інженерії

Рівень вищої освіти перший (бакалаврський)

Спеціальність 121 – Інженерія програмного забезпечення

Тип програми Освітньо-професійна

Освітня програма Програмна Інженерія

(шифр і назва)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Курс | 3 | Група | ПЗПІ-22-4 | Семестр | 6 |

**ЗАВДАННЯ**

***на курсовий проект(роботу) студента***

здобувачеві Герцеву В’ячеславу Андрійовичу

(прізвище, ім’я, по батькові)

1. Тема роботи Ігровий програмний застосунок в жанрі Roguelike

2. Термін здачі студентом закінченої роботи “ 29 ” 05 2025 р.\_

3. Вихідні дані до проєкту розробка гри в середовищі Unity з використанням мови програмування C#; застосування елементів UI/UX дизайну; імплементація ігрової логіки, керування гравцем, об’єктами та ворогами; реалізація веб сторінки з описом гри та функцією завантаження.

4. Перелік питань, що потрібно опрацювати в роботі

Розробка ігрової архітектури з використанням UML-діаграм, реалізація інтерфейсу користувача та забезпечення юзабіліті, реалізація візуальної частини гри в Unity, підготовка графічних і демонстраційних матеріалів.

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назва етапів роботи | Термін виконання етапів роботи | Примітка |
| 1 | Аналіз предметної галузі | 23.03 – 25.03 | виконано |
| 2 | Розробка постановки задачі | 26.03 – 27.03 | виконано |
| 3 | Проектування ПЗ | 01.04 – 04.03 | виконано |
| 4 | Програмна реалізація | 14.03 – 02.06 | виконано |
| 5 | Аналіз результатів | 02.06 – 04.06 | виконано |
| 6 | Підготовка пояснювальної записки. | 05.06 – 18.06 | виконано |
| 7 | Перевірка на наявність ознак академічного плагіату | 19.06 – 20.06 | виконано |
| 8 | Захист роботи | 22.06 | виконано |

Дата видачі завдання “ 21 ” 03 2025р.

Здобувач \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис)

Керівник роботи доцент кафедри ПІ Наталя ВАЛЕНДА

(підпис) (посада, Власне ім’я, ПРІЗВИЩЕ)

**РЕФЕРАТ / ABSTRACT**

Пояснювальна записка містить: 48 с., 22 рис., 10 джерел.

МОБІЛЬНА ГРА, ІНТЕРФЕЙС КОРИСТУВАЧА, ГЕЙМДИЗАЙН, РОГАЛИК, ТЕТРІС, UNITY, C#, МОБІЛЬНИЙ ФРОНТЕНД

Об'єктом розробки є мобільна гра «Black Prince», що поєднує елементи рогалика, головоломки та аркади, з акцентом на інноваційну ігрову механіку взаємодії з «магічними цеглинами».

Метою розробки є проєктування та реалізація клієнтської частини мобільного застосунку – гри «Black Prince», яка забезпечує інтуїтивно зрозумілий і привабливий інтерфейс, а також візуальну частину геймплею, що включає управління цеглинами, бойові дії, анімації, заставки та рівні.

Методи розробки клієнтської частини включають використання ігрового рушія Unity з мовою програмування C#, а також інструментів для створення UI/UX (Unity UI Toolkit). Візуальні елементи створювалися із застосуванням Figma, Adobe Illustrator та Blender. Під час розробки враховано принципи адаптивного дизайну, зручності для коротких сесій, а також інтуїтивної навігації.

У результаті проєкту створено візуальну частину мобільної гри Black Prince, що включає меню, екрани переходів, графіку рівнів, ефекти взаємодії з ігровими об'єктами та систему збереження прогресу, орієнтовану на кінцевого користувача мобільного пристрою.

MOBILE GAME, USER INTERFACE, GAME DESIGN, ROGUELIKE, TETRIS, UNITY, C#, MOBILE FRONTEND

The object of development is a mobile game called «Black Prince», which combines elements of roguelike, puzzle, and arcade gameplay, focused on innovative mechanics using "magic bricks.

The goal of the development is to design and implement the client side of the mobile application «Black Prince», ensuring an intuitive and engaging interface and a visually rich game experience. The visual elements include brick control, combat interactions, level graphics, animations, and navigation screens.

The methods of frontend development include the Unity game engine with C# programming language, supported by UI/UX tools such as Unity UI Toolkit, Figma, Adobe Illustrator, and Blender. The interface is built according to mobile-first principles, prioritizing accessibility, short-session interaction, and visual clarity.

As a result, a fully functional visual layer of the mobile game Black Prince has been developed, including dynamic interface components, transition scenes, animated effects, and progress-saving system tailored to mobile device users.

**ЗМІСТ**

Вступ.................................................................................................................................8

1. Аналіз предметної області……………………..........................................................9
   1. Аналіз предметної області………………………………………………..9
   2. Виявлення проблем та актуалізація рішень ……………………………13
   3. Постановка задачі…………………………………………………………15
2. Формування вимог до ігрового програмного модулю...........................................17
   1. Концепція гри та ігрові механіки………………………………………..17
   2. Функціональні та нефункціональні вимоги…………………………….19
   3. Вимоги до інтерфейсу, платформи та інструментів розробки………...20
3. Архітектура та проектування ігрового програмного забезпечення …………....22
   1. UML проектування ПЗ …………………………………………………...22
   2. Проектування архітектури ПЗ …………………………………………..28
   3. Проектування бази даних ……………………………………………….31
   4. Створення UI/UX………………………………………………………...32
4. Опис прийнятих програмних рішень……………………………………………..36
5. Впровадження ігрового програмного забезпечення ……………………..…….43

Висновки........................................................................................................................46

Перелік джерел посилання..........................................................................................48

**ПЕРЕЛІК СКОРОЧЕНЬ**

UI – User Interface

UX – User Experience

HUD – Head-Up Display

FPS – Frames Per Second

API – Application Programming Interface

SDK – Software Development Kit

IDE – Integrated Development Environment

2D – Two-Dimensional

3D – Three-Dimensional

AI – Artificial Intelligence

UML – Unified Modeling Language

IT – Information Technology

RPG – Role-Playing Game

JSON – JavaScript Object Notation

ADB – Android Debug Bridge

IL2CPP – Intermediate Language to C++

**ВСТУП**

Сфера цифрових розваг постійно розширюється, і серед численних жанрів відеоігор особливої популярності набувають ігри з простими, але захопливими механіками, доступні для мобільних платформ. Гравці все частіше шукають досвід, який поєднує короткі сеанси гри із глибоким зануренням у геймплей, орієнтуючись на зручність, візуальну привабливість і швидкий прогрес. На цьому тлі створення власного ігрового продукту дає змогу не лише продемонструвати практичні навички розробки, а й створити конкурентоспроможний продукт із потенціалом до масштабування.

Гра Black Prince розробляється як 2D-аркада з елементами фізики та бойової тактики, у якій гравець керує персонажем, що протистоїть ворогам, використовуючи обмежену кількість снарядів (цеглин). Ключовим елементом ігрового процесу є кидання об’єктів у ворогів із урахуванням траєкторії, сили та ефектів зіткнення. Успіх гри залежить від точності, таймінгу та вміння швидко реагувати на зміну ситуації.

Актуальність розробки полягає у зростанні попиту на мобільні ігри, які поєднують короткі ігрові сесії з інтуїтивним управлінням і якісним візуальним оформленням. Проєкт Black Prince розрахований на широку аудиторію користувачів мобільних пристроїв – від школярів до дорослих гравців, яким цікаві динамічні фізичні задачі у формі розважального сюжету. Крім того, важливою метою є відпрацювання практичних навичок UI/UX-дизайну, проєктування архітектури клієнтського програмного модуля, а також реалізації логіки гри засобами рушія Unity.

Метою курсової роботи є створення фронтенд-частини гри Black Prince у вигляді прототипу, придатного для розширення та подальшого вдосконалення. В межах цього проєкту передбачається реалізувати ігрові механіки, інтерфейс користувача, архітектуру модулів гри та документувати ці аспекти за допомогою UML-діаграм.

Основними завданнями, які ставляться у межах роботи, є:

* аналіз предметної області мобільних ігор;
* розробка архітектури клієнтського застосунку;
* проєктування модулів за допомогою UML-діаграм;
* реалізація базової логіки гри та управління персонажем;
* тестування взаємодії модулів та перевірка їх стабільності.

Результати роботи можуть бути використані для побудови комерційних або навчальних проєктів у сфері мобільних ігор, а також як база для інтеграції складніших механік, мультиплеєру чи системи досягнень у подальших ітераціях гри.

**1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ**

1.1 Аналіз предметної галузі

Сфера мобільного геймінгу є однією з найдинамічніших у сучасній ІТ-індустрії. За останні роки спостерігається стрімке зростання попиту на мобільні ігри, що обумовлено широкою доступністю смартфонів, розвитком мобільного інтернету та популярністю коротких ігрових сесій у повсякденному житті користувачів. Відповідно до звітів аналітичних агентств (Newzoo, Statista), близько 50% глобального доходу гейм-індустрії припадає саме на мобільний сегмент [1] (див рис. 1.1).

Изображение выглядит как текст, снимок экрана, круг, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.1 – Діаграма доходу за платформами на 2024 рік [1]

Мобільні ігри класифікуються за різними жанрами – головоломки, аркади, екшени, RPG, симулятори тощо. Серед них стабільно високим попитом користуються гібридні жанри, які поєднують кілька типів геймплею в одному проєкті [2]. Саме в цьому напрямі розвивається гра Black Prince, яка поєднує риси рогалика, головоломки й аркади з динамічною бойовою системою.

Гравці все частіше звертають увагу не лише на ігрову механіку, а й на зручність інтерфейсу, якість візуального оформлення та загальний користувацький досвід (UX). В умовах жорсткої конкуренції на ринку, саме фронтенд-частина (графіка, UI, анімації, інтерактивність) значною мірою визначає, чи затримається користувач у грі після перших 30 секунд взаємодії [3].

Багато популярних мобільних ігор, таких як Candy Crush Saga, Tetris Blitz, Archero чи Dead Cells (Mobile), демонструють різні підходи до оформлення інтерфейсу та адаптації геймплею до формату коротких сесій. Наприклад, Archero пропонує гравцеві мінімалістичний інтерфейс із максимальним фокусом на управління одним пальцем (див. рис. 1.2), тоді як Dead Cells має більш складний HUD, але чудово оптимізований для сенсорного управління (див. рис. 1.3).

У таких іграх простежуються спільні ознаки успішного фронтенд-рішення:

* чітка ієрархія елементів інтерфейсу;
* інтуїтивна навігація;
* адаптивність під різні розміри екранів;
* консистентна візуальна мова (кольори, шрифти, стилізація);
* швидкий доступ до основних дій (почати гру, продовжити, налаштування тощо).

Изображение выглядит как снимок экрана, зеленый

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.2 – Гра Archero (виконано самостійно)



Рисунок 1.3 – Гра Dead Cells (виконано самостійно)

Окремо варто згадати гру The Binding of Isaac, яка стала еталонним прикладом успішної реалізації жанру рогалик із глибоким бойовим геймплеєм, процедурною генерацією рівнів і великою кількістю унікальних предметів, ворогів та механік. Незважаючи на те, що вона спочатку була створена як desktop-проєкт, пізніше з’явилися адаптовані версії для мобільних платформ і Nintendo Switch. Особливість Isaac – в емоційно насиченому візуальному стилі, який водночас поєднує мінімалізм UI і насиченість ігрового простору (див. рис. 1.4).

На основі аналізу цієї гри видно:

* як важливо дотримуватись збалансованості між складністю управління та глибиною механіки;
* наскільки важливу роль відіграє інформативність візуального представлення елементів гри (життя, предмети, атаки, вороги тощо);
* як стилізована графіка може підсилювати атмосферу гри і формувати її унікальну ідентичність [2].



Рисунок 1.4 – Геймплей гри The Binding of Isaac (виконано самостійно)

Аналіз ринку показує, що попит на ігри з унікальним геймплеєм та якісним інтерфейсом зростає. Більшість таких продуктів або надто складні для мобільної аудиторії, або надмірно спрощені [4]. Це створює нішеву можливість для розробки гри, яка:

* не вимагає довготривалого навчання;
* має візуально привабливий, але простий інтерфейс;
* надає гравцю свободу дій та тактичні варіанти;
* підтримує короткі та середні сесії гри.

Таким чином, предметна область проєкту Black Prince – це мобільні ігри гібридного жанру, орієнтовані на казуальних гравців, які шукають новий ігровий досвід у зручному та візуально привабливому форматі.

Вивчення предметної області дало змогу сформулювати вимоги до візуальної частини гри, що полягають у:

* створенні інтуїтивного користувацького інтерфейсу;
* адаптації елементів управління під сенсорні екрани;
* побудові візуального стилю, що відповідає тематиці гри;
* забезпеченні чіткого фідбеку від взаємодій гравця з елементами гри;
* оптимізації графіки для різних пристроїв.

У подальших підрозділах буде детально розглянуто проблеми, які виникають під час реалізації таких вимог, та поставлено конкретну задачу для розробки інтерфейсу гри Black Prince.

1.2 – Виявлення проблем та актуалізація рішень

У процесі аналізу предметної області мобільного геймінгу виявлено низку проблем, які суттєво впливають на якість користувацького досвіду у таких іграх. Ці проблеми безпосередньо стосуються візуальної та інтерфейсної частини мобільного застосунку, що є основною зоною відповідальності фронтенд-розробника у проєкті Black Prince. Їх вирішення є необхідною передумовою створення конкурентоспроможного продукту, який відповідатиме сучасним вимогам ринку.

Однією з найпоширеніших проблем мобільних ігор є перевантаження інтерфейсу – коли на екрані одночасно розміщено надто багато елементів: кнопки, лічильники, індикатори, підказки, реклами. Це ускладнює орієнтування, створює візуальний шум і знижує комфорт користувача, особливо на пристроях із невеликою діагоналлю екрана [5]. У грі Black Prince, яка поєднує динамічну бойову систему, головоломки та будівельні механіки, збереження чистоти ієрархії елементів є критично важливим завданням. Щоб вирішити це, доцільно застосувати модульний підхід, за якого елементи інтерфейсу з’являються лише в контексті їх актуальності, а також розташовуються в логічно розділених зонах з урахуванням пріоритетів взаємодії [6].

Другою важливою проблемою є складність адаптації інтерфейсу під різноманітні типи мобільних пристроїв. В умовах ринку, де користувачі мають пристрої з найрізноманітнішими розмірами та співвідношенням сторін, розробник повинен враховувати не лише технічну сумісність, а й візуальну якість відображення. Рішенням у даному випадку є використання адаптивної сітки у Unity UI, що масштабується відповідно до розміру екрана, з фіксованими відступами та масштабами, а також тестування на низці фізичних і віртуальних пристроїв [7].

Ще однією типовою проблемою є відсутність або недостатність візуального фідбеку на дії користувача. У динамічних іграх гравець має отримувати миттєве підтвердження кожної своєї дії – чи то натискання кнопки, кидання цеглини, чи нанесення удару ворогу. Це створює ефект контролю та занурення. У Black Prince реалізація візуального фідбеку передбачає використання анімацій, спалахів, зміни кольору, звукових сигналів, а також вібрації – тобто мультимодального підходу до зворотного зв’язку [8].

Також існує проблема візуальної неузгодженості елементів інтерфейсу, коли графіка гри виглядає як набір фрагментів, створених у різних стилях. Це не лише шкодить естетиці, а й порушує візуальну цілісність, яка є надзвичайно важливою для створення впізнаваного стилю. У грі Black Prince передбачається використання єдиної кольорової палітри, шрифтів, графічних рішень та обводок, що підтримуватимуть загальну атмосферу похмурої середньовічної стилістики з магічними елементами [9].

Крім того, слід враховувати специфіку мобільної аудиторії, яка зазвичай взаємодіє з іграми в режимі коротких сесій. Багато гравців запускають ігри у транспорті, на перервах або у вільні 5–10 хвилин. Якщо гра не дає змоги швидко зберігатися або вимагає тривалого проходження, її буде відкинуто на користь конкурентів. Щоб уникнути цього, у Black Prince має бути реалізована система коротких рівнів, автозбереження прогресу, функціонал паузи без втрати даних, а також збереження у хмарі, наприклад через OneDrive [10].

Ще один аспект, що впливає на фронтенд, – це продуктивність. Надмірна кількість візуальних ефектів, анімацій або неефективно побудованих UI-компонентів може спричинити падіння FPS, що зробить гру некерованою на слабших пристроях. Для Black Prince планується застосування оптимізаційних технік, таких як спрайт-атласи, пулінг об’єктів, контролювання кількості активних елементів на сцені, а також графічні налаштування для різних рівнів продуктивності.

Узагальнюючи, варто зазначити, що проблеми, виявлені на етапі аналізу, мають системний характер і безпосередньо впливають на успішність продукту. Саме тому актуалізація рішень відбувається через впровадження адаптивного дизайну, модульного підходу до UI, єдиного стилю оформлення, мультимодального фідбеку та оптимізації графічної продуктивності. Такий підхід дозволяє не лише покращити користувацький досвід, а й зробити продукт привабливим для широкої аудиторії мобільних гравців.

1.3 Постановка задачі

На основі аналізу предметної області та виявлених проблем сформульовано задачу курсового проєкту, яка полягає у створенні якісної клієнтської частини мобільної гри Black Prince. Гра має поєднувати в собі механіки головоломки, рогалика та аркади, тому візуальна складова повинна забезпечувати як естетичну привабливість, так і повну функціональну відповідність до вимог ігрового процесу.

Об’єкт розробки – мобільна гра Black Prince, створена за допомогою рушія Unity.

Предмет розробки – інтерфейс користувача, екранна навігація, інтерактивні елементи, анімації, стилізація графіки та логіка відображення візуального контенту у грі.

Мета розробки – створити інтуїтивно зрозумілий, візуально цілісний, адаптивний інтерфейс мобільної гри Black Prince, який забезпечить комфортну взаємодію користувача з грою, враховуючи особливості мобільних платформ та коротких ігрових сесій.

Для досягнення поставленої мети необхідно вирішити такі завдання:

* розробити дизайн основного ігрового інтерфейсу з урахуванням модульної структури та мінімалістичного підходу;
* забезпечити адаптацію UI під різні розміри екранів мобільних пристроїв;
* створити візуально узгоджену стилістику у рамках обраної тематики (середньовіччя з магічними елементами);
* реалізувати ефекти зворотного зв’язку на дії гравця (натискання, переміщення, атаки);
* впровадити систему переходів між екранами (головне меню, екран паузи, вікно програшу/перемоги, налаштування тощо);
* оптимізувати використання графічних ресурсів для забезпечення стабільної роботи гри на пристроях із середньою та низькою продуктивністю;
* забезпечити можливість швидкого збереження та відновлення прогресу;
* протестувати функціонування візуальних елементів на різних платформах і внести корективи за результатами тестування.

Очікуваним результатом проєкту є створення повноцінного візуального прототипу гри Black Prince, реалізованого у вигляді інтерактивного інтерфейсу на базі Unity, який можна інтегрувати до основного ігрового ядра. Врахування вищезазначених завдань дозволить забезпечити зручність взаємодії користувача з грою, візуальну цілісність, естетичність та відповідність сучасним вимогам до мобільних застосунків.

**2 ФОРМУВАННЯ ВИМОГ ДО ІГРОВОГО ПРОГРАМНОГО МОДУЛЯ**

2.1 Концепція гри та ігрові механіки

Мобільна гра Black Prince – це проєкт у жанрі аркадного рогалика з елементами головоломки, побудований навколо унікальної механіки взаємодії з «магічними цеглинами». Гравець виступає в ролі загадкового принца, який володіє силою матеріалізовувати цеглини, використовуючи їх для боротьби, побудови та вирішення просторових задач. Головна ідея гри полягає в поєднанні простого сенсорного керування з багаторівневою тактикою і варіативністю проходження.

Black Prince орієнтована на мобільну платформу, зокрема Android-пристрої. Формат гри враховує різні сценарії ігрових сесій. Короткі сесії (5 – 15 хвилин) ідеальні для гравців, які грають у перервах або під час поїздок. Сеанс може складатися з одного або кількох рівнів чи завдань, які завершуються швидко, без втрати відчуття прогресу. Середні сесії (15 – 30 хвилин) дозволяють пройти кілька послідовних рівнів, вирішити складні головоломки або взяти участь у бойових етапах. Це найоптимальніший формат для глибшого занурення в гру без надмірного навантаження. Довгі сесії (30 хвилин і більше) призначені для гравців, які хочуть повністю зануритися у світ гри – наприклад, на вихідних або у вечірній час. У таких сесіях можливе проходження кількох великих рівнів, боротьба з босами або прогрес через сюжетні розгалуження.

Black Prince побудована за моделлю довгої сесії, поділеної на короткі та середні структурні одиниці, що дозволяє підтримувати інтерес до гри та не навантажувати користувача зайвою складністю. Рівні проєктуються так, щоб кожен мав завершену логіку та приносив гравцеві задоволення навіть у межах кількох хвилин взаємодії.

Основна геймплейна механіка побудована навколо використання «магічних цеглин», які можуть бути:

* інструментом атаки – гравець може кидати цеглини у ворогів, зіштовхувати їх у прірву або активувати пастки;
* будівельним матеріалом – гравець може формувати мости, платформи, тимчасові укриття;
* ключем до головоломок – цеглини можна використовувати для натискання важелів, блокування лазерів або формування специфічних форм для активації механізмів.

Гра не має класичної сюжетної структури з діалогами або рівнями на час, однак передбачає поступове ускладнення геймплею: з'являються нові типи ворогів, механізми, обмеження простору, додаткові завдання. Кожен рівень – це окрема головоломка або бойове випробування, що дозволяє гравцеві обрати власну стратегію: атакувати, обійти, заблокувати або комбінувати.

Інтерфейс гри складається з кількох ключових екранів:

* головне меню – доступ до гри, налаштувань, інформації про розробника;
* налаштування – звук, мова, управління, скидання прогресу;
* екран гри – зона взаємодії з цеглинами, індикатори здоров’я, спроби, кнопка паузи;
* вікно поразки або перемоги – підсумкова статистика, кнопки «повторити» або «наступний рівень»;
* пауза – кнопки «продовжити», «вийти», «налаштування».

Управління реалізовано повністю сенсорним способом:

* свайпами – гравець пересуває персонажа;
* тапає – кидає цеглини або взаємодіє з об’єктами;
* утримування – заряджає кидок або змінює напрям дії.

Візуально гра виконана в стилістиці середньовічного темного фентезі з елементами містики. Колористика базується на темних відтінках сірого, з фіолетовими акцентами для ефектів і активних елементів. Шрифт – стилізований під рукописний, проте достатньо контрастний для зручного читання на екранах смартфонів. Усі графічні елементи (іконки, спрайти, анімації) витримані в єдиному стилі, що забезпечує візуальну цілісність гри.

Таким чином, Black Prince – це гра, яка поєднує логіку, дію та творчість в одному компактному форматі, доступному для широкої мобільної аудиторії. Гнучкий ігровий процес, інтуїтивне управління та привабливий дизайн забезпечують базу для створення якісного програмного модуля з перспективою масштабування.

2.2 Функціональні та нефункціональні вимоги

Для забезпечення повноцінного функціонування мобільної гри Black Prince, яка поєднує динамічний геймплей, головоломки та взаємодію з тактичним середовищем, необхідно визначити чіткий перелік функціональних та нефункціональних вимог. Врахування цих вимог на етапі проєктування забезпечить стабільність роботи гри, зручність користувача, адаптивність до різних пристроїв і готовність до розширення функціоналу в майбутньому.

Функціональні вимоги:

* відображення ігрового поля та об'єктів у реальному часі;
* керування цеглинами (перетягування, кидання);
* система зіткнень з ворогами та навколишнім середовищем;
* індикатори здоров’я, лічильники спроб, індикатор прогресу рівня;
* екран перемоги з кнопками «повторити», «наступний рівень»;
* меню налаштувань: звук, управління, скидання прогресу;
* збереження прогресу в локальну пам’ять або хмару;
* реакція на події: звук, вібрація, анімація.

Реалізація цих функцій створить інтуїтивно зрозумілу та логічну структуру гри, яка не потребує додаткових інструкцій для початку гри.

Нефункціональні вимоги:

* плавність анімацій (не менше 30 FPS на середньому пристрої);
* підтримка роздільної здатності 16:9, 18:9, 21:9;
* низьке енергоспоживання (не більше 15% заряду за 30 хвилин гри);
* швидкий запуск гри (до 5 секунд);
* можливість гри в офлайн-режимі (окрім хмарного збереження);
* UI має бути інтуїтивно зрозумілим без додаткових інструкцій.

Усі ці вимоги створені з урахуванням особливостей мобільного ринку – гри повинні бути легкими для засвоєння, швидко запускатися, мати візуально чисту структуру та не перевантажувати пристрій.

2.3 Вимоги до інтерфейсу, платформи та інструментів розробки

Якість візуального оформлення мобільної гри є одним із ключових чинників, що впливають на перше враження користувача, рівень занурення в ігровий процес і бажання продовжувати гру. У проєкті Black Prince велика увага приділяється реалізації інтерфейсу користувача, його адаптивності, візуальній цілісності та відповідності очікуванням мобільної аудиторії.

Інтерфейс гри має бути мінімалістичним, логічно структурованим і візуально узгодженим. Елементи UI розміщуються за модульною сіткою, що масштабується під екран пристрою. Усі кнопки, іконки та індикатори повинні бути добре помітними, не перекривати ігрове поле та зберігати свою функціональність на екранах із різним співвідношенням сторін. Колірна палітра гри витримується у темних фонових відтінках із фіолетовими акцентами, які асоціюються з магічною атмосферою. Шрифти обираються стилізовані, але достатньо контрастні й масштабовані, щоб забезпечити зручність читання навіть на екранах невеликої діагоналі.

Особливу увагу приділено реалізації зворотного зв’язку. Кожна дія користувача має супроводжуватися візуальними ефектами – натискання кнопок супроводжується зміною кольору, анімацією або легким підсвічуванням. Кидання цеглин у ворогів супроводжується ефектами частинок і звуковими сигналами. Також може бути використана вібрація як додатковий канал сприйняття дії. Усі елементи управління розташовуються в межах зони доступу великого пальця – тобто гра має підтримувати зручну гру однією рукою.

Платформою для реалізації гри обрано Android з підтримкою версій 8.0 і новіших. Це дає змогу охопити більшість сучасних мобільних пристроїв. У подальшому проєкт може бути портований на iOS без суттєвих змін у фронтенд-архітектурі. Рушієм розробки є Unity, що надає повний набір інструментів для 2D-графіки, анімацій, фізики та UI-компонентів. Уся логіка гри реалізовується мовою програмування C# із використанням системи Canvas UI або Unity UI Toolkit для побудови масштабованих інтерфейсів.

Для проєктування макетів та погодження візуальної частини використовується Figma – як інструмент попереднього макетування, створення прототипів і тестування UX-рішень. Іконки та статичні графічні елементи створюються у Adobe Illustrator у векторному форматі для масштабованості. За потреби для анімацій або створення простих 3D-елементів використовується Blender. Для реалізації добровільної підтримки розробника планується інтеграція платіжного SDK – LiqPay, з розміщенням відповідної кнопки у меню гри.

Розробка інтерфейсної частини гри здійснюється за ітеративним принципом: спочатку створюються макети екранів, потім вони погоджуються з художником і геймдизайнером, після чого реалізуються в Unity, тестуються на різних пристроях і коригуються. Для роботи в команді застосовується Scrum-підхід з короткими спринтами та щоденними синхронізаціями.

Розробка орієнтується на стабільність, масштабованість та простоту в розширенні: інтерфейс легко адаптується під додавання нових екранів (магазин, інвентар, рівні тощо), а структура коду дозволяє повторне використання компонентів.

Таким чином, вимоги до інтерфейсу, платформи та засобів розробки гри Black Prince сформульовані таким чином, щоб забезпечити сучасний, привабливий, стабільний і зручний для користувача візуальний досвід, який відповідає очікуванням мобільної аудиторії та стандартам індустрії.

**3 АРХІТЕКТУРА ТА ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

3.1 UML проєктування ПЗ

У контексті розробки мобільної гри Black Prince UML-діаграми використовувалися для опису ігрових сценаріїв, внутрішньої структури об'єктів, послідовності взаємодій та життєвих циклів елементів гри.

На етапі проєктування одним із ключових кроків стало створення діаграми варіантів використання (Use Case Diagram), яка відображає взаємодію користувача з ігровими функціями (див. рис. 3.1).

Изображение выглядит как текст, диаграмма, снимок экрана, круг

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.1 – Use-case діаграма

Основний актор – гравець, який взаємодіє з системою через такі сценарії, як запуск гри, початок нового рівня, керування персонажем, кидання магічних цеглин, перемога або поразка, а також перехід до меню налаштувань. Така діаграма допомагає уявити логіку гри з точки зору користувача та визначити, які модулі відповідають за реалізацію кожного варіанту використання.

Для опису структури ієрархії об'єктів гри було створено діаграму класів (див. рис. 3.2). Вона дозволяє наочно уявити ключові сутності, їхні атрибути, методи, спадкування та зв’язки між ними.Центральним класом є Game, який виступає як точка входу до гри та координує взаємодію між усіма компонентами. Він містить посилання на такі модулі, як Level, Player, Enemy, CollisionManager, Foreground, а також управляє ініціалізацією, оновленням стану та завершенням гри. Таким чином, Game реалізує патерн фасаду, делегуючи конкретну логіку допоміжним класам.

Изображение выглядит как текст, диаграмма, План, Параллельный

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.2 – Діаграма класів

Клас Level відповідає за структуру конкретного ігрового рівня. Він містить дані про розміщення об'єктів, доступні ресурси, типи ворогів, стартову позицію гравця та кінцеву точку. Рівень створюється на основі шаблонів або вручну, що дозволяє масштабувати гру й додавати нові рівні без змін основного коду.

Клас Player – основна ігрова одиниця, яку контролює користувач. Він містить атрибути, що описують стан персонажа (координати, здоров’я, швидкість, активні здібності), а також методи для виконання ігрових дій: переміщення, кидання цеглин, взаємодія з об'єктами, реакція на атаки ворогів. У контексті взаємодії з класом Projectile, гравець може створювати снаряди з магічних цеглин.

Клас Enemy є узагальненням усіх типів ворогів. Залежно від конкретної реалізації, вороги можуть атакувати гравця, переслідувати його або патрулювати територію. Усі підкласи ворогів успадковують базову поведінку (move, attack, takeDamage), але можуть перевизначати її залежно від типу (наприклад, ближній/дальній бій, рух по траєкторії, телепортація тощо).

Клас Item – допоміжна сутність, що включає різноманітні предмети, які гравець може зібрати або використати. Це можуть бути підсилення, ключі, бонуси, а також пасивні об'єкти (наприклад, важелі для активації дверей). Логіка підбирання предметів і їх вплив на гравця реалізована через методи interactWith(Player) або applyEffect().

Клас Projectile моделює снаряди, зокрема магічні цеглини, якими гравець може атакувати ворогів або взаємодіяти з навколишнім середовищем. Об'єкти цього класу мають параметри траєкторії, швидкості, шкоди, а також логіку обробки зіткнень.

Клас Character виступає як батьківський клас для Player і Enemy. У ньому зосереджено спільні характеристики: здоров’я, координати, методи takeDamage, die, move. Це дозволяє застосовувати принцип поліморфізму, обробляючи всі активні об'єкти сцени як екземпляри одного базового типу.

Клас CollisionManager відповідає за виявлення та обробку зіткнень між усіма активними об’єктами: гравцем, ворогами, снарядами, предметами. Це дозволяє централізувати логіку колізій та обробляти їх у рамках одного циклу гри.

Клас Foreground використовується для графічного оформлення переднього плану, тобто елементів, які накладаються поверх геймплею (HUD, ефекти, декоративні елементи). Він не має прямої участі в геймплеї, але визначає візуальну складову сцени.

Усі зв’язки між класами моделюються відповідно до принципів об’єктно-орієнтованого проєктування: агрегування (Game – Level), композиція (Level – Enemy, Item, Foreground), спадкування (Player і Enemy – Character). Така структура забезпечує гнучкість, повторне використання коду, масштабованість і легку підтримку проєкту в майбутньому.

Особливу увагу було приділено діаграмі станів персонажа (див. рис. 3.3).

Изображение выглядит как текст, диаграмма, План, Технический чертеж

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.3 – Діаграма станів

На діаграмі зображено модель переходів між основними станами ігрового персонажа – класу Player. Вона демонструє, як залежно від взаємодії користувача та внутрішніх подій змінюється поведінка об’єкта протягом ігрового процесу.

Гравець починає у стані Idle (очікування), і залежно від натискання певних кнопок переходить у відповідні стани:

* натискання кнопок руху (Left/Right) переводить гравця у стан Move;
* натискання кнопки стрибка – у стан Jump, після якого можливий перехід у Fall при втраті контакту з землею;
* натискання кнопки атаки – у стан Attack, що повертається до Idle після завершення атаки;
* контакт із ворогом – у стан Take damage, де гравець втрачає здоров’я.

У разі, якщо після завдання шкоди health дорівнює 0, відбувається перехід до кінцевого стану – Endgame logic, що може означати програш, перезапуск або вихід до меню. Усі переходи супроводжуються відповідними умовами або діями

Щоб деталізувати взаємодію між ігровими об'єктами в процесі бою, було побудовано діаграму послідовностей (sequence diagram), яка моделює сценарій «атаки гравця на ворога за допомогою магічної цеглини». Ця діаграма показує порядок викликів методів між об'єктами Player, GameSystem, Brick (GameObject), Enemy AI, що виникають при бойовій взаємодії (див. рис. 3.4).

Процес починається з дії гравця – вибору цеглини через сенсорне управління. Гравець перетягує цеглину у напрямку ворога, ініціюючи подію Select Brick, яка надсилається до об’єкта Brick. Далі викликається метод CalculateTrajectory, що дозволяє системі визначити напрям польоту снаряду. Цей етап супроводжується запитом до ігрової системи (GameSystem) для визначення траєкторії польоту, після чого Brick повертає координати цілі.

Після підтвердження напрямку, гравець відпускає цеглину, і запускається анімація польоту. Brick переходить у стан In flight. Одночасно викликається метод CheckCollision, що взаємодіє з Enemy AI для виявлення можливого зіткнення. Якщо зіткнення відбувається, Brick повідомляє про це систему, яка активує обробку бою (ShowCombatResult) – зменшується здоров’я ворога (ReduceEnemyHealth), а система зберігає або відображає результат бою.

Изображение выглядит как текст, диаграмма, Параллельный, число

Контент, сгенерированный ИИ, может содержать ошибки. Рисунок 3.4 – Діаграма послідовностей

Сценарій завершується поверненням до гравця візуального фідбеку: ворог анімує реакцію на отримання шкоди, а також оновлюються показники в інтерфейсі (наприклад, рахунок, залишки цеглин, кількість ворогів на рівні).

Узагальнюючи, застосування UML-діаграм у процесі проєктування гри Black Prince дозволило не лише задокументувати архітектуру програмного модуля, а й забезпечити узгодженість у розумінні між розробником, геймдизайнером, дизайнером інтерфейсу та іншими членами команди. Це значно підвищило ефективність командної роботи, пришвидшило розробку й полегшило налагодження проєкту в середовищі Unity.

3.2 Проектування архітектури ПЗ

Архітектура програмного забезпечення мобільної гри Black Prince побудована з урахуванням цілей проєкту, особливостей обраної платформи (Android) та ресурсів розробки. Вона покликана забезпечити стабільність, ефективність та масштабованість програмного продукту, дозволяючи при цьому створити якісну гру з відлагодженим інтерфейсом, логікою та ігровим процесом.

Гра реалізується у вигляді єдиного клієнтського застосунку, побудованого на рушії Unity. Вона функціонує повністю офлайн, не потребуючи постійного підключення до інтернету, а всі дані про прогрес зберігаються локально на пристрої користувача. Така архітектура була обрана свідомо – вона значно спрощує розгортання, не вимагає додаткової інфраструктури (наприклад, серверів), та є оптимальною для ігор із одиночним проходженням.

Загалом, архітектура проєкту є монолітною, проте логічно структурованою. Всі компоненти чітко поділені за функціональними ознаками. Центральним елементом архітектури гри Black Prince виступає GameEngine.cs, який координує всі основні процеси в ігровій системі. Він керує станами гри, запуском рівнів, обробкою перемог і поразок, комунікацією між ігровими об’єктами та іншими компонентами. Саме GameEngine ініціює більшість системних дій, включно з оновленням фізики, обробкою введення та візуалізацією.

З GameEngine тісно взаємодіє LevelManager.cs, який відповідає за завантаження і повторне ініціювання рівнів, а також з UIManager.cs, що керує відображенням усіх елементів інтерфейсу: меню, HUD, повідомлень про програш або перемогу.

Ключовим модулем для керування персонажем є CharacterController.cs, який координує рухи та стани гравця. Він тісно пов’язаний з InputHandler.cs – компонентом, що опрацьовує сенсорне введення користувача та передає відповідні команди іншим системам. Зокрема, InputHandler керує тим, як гравець взаємодіє з PlayerMovement.cs для управління переміщенням, і з BlockMovement.cs для запуску цеглин.

BlockMovement.cs відповідає за обробку руху снарядів (цеглин), включно з фізичними законами, траєкторією, а також ініціацією зіткнень. Виявлення самих зіткнень реалізовано через CollisionDetector.cs, який обробляє зіткнення між гравцем, снарядами та ворогами. Дані про колізії передаються назад у GameEngine для логічної обробки: зменшення здоров’я, завершення гри тощо.

EnemyAI.cs реалізує поведінку ворогів: пересування, реагування на появу гравця, атаки. Цей модуль також взаємодіє з CollisionDetector для обробки ушкоджень, отриманих від блоків. Всі об’єкти, які можуть отримувати шкоду (вороги, гравець), реалізують інтерфейс IDamageable, що забезпечує єдину точку обробки пошкоджень без необхідності прив’язки до конкретних класів.

Важливу роль відіграє PhysicsEngine.cs, що відповідає за обробку гравітації, імпульсів та інших фізичних ефектів. RenderEngine.cs відповідає за графічне відображення сцени, а AudioEngine.cs – за звуковий супровід: фонову музику, ефекти пострілів, зіткнень, повідомлень тощо. Взаємодія між цими трьома модулями йде через виклики з GameEngine або відповідні події.

EventSystem.cs реалізує внутрішню подієву модель гри. Наприклад, при загибелі гравця генерується подія onPlayerDeath, яка слухається UIManager для відображення екрана завершення гри, а також AudioEngine – для програвання відповідного звуку. На основі цих компонентів було сформовано компонентну діаграму (див. рис. 3.5)

Усі модулі спроєктовані з чіткими інтерфейсами. Наприклад, IControllable – для об’єктів, які можуть бути керованими гравцем, IDamageable – для тих, що отримують шкоду, IRenderable – для графічних компонентів. Це забезпечує гнучкість системи та полегшує її розширення. Додати нового ворога, нову механіку або інтерфейсну панель можна, не змінюючи основної логіки гри, а лише реалізувавши відповідний інтерфейс і зареєструвавши об’єкт у потрібному менеджері.

Уся система функціонує в межах сцени Unity, використовуючи класичну архітектуру на основі MonoBehaviour, з використанням делегатів, викликів методів та подій. Це робить систему легкою для підтримки, масштабування й адаптації під нові сценарії або рівні. Також для кращого розуміння розміщення компонентів та їх зв’язків на рисунку 3.6 зображено діаграму розгортування.

Изображение выглядит как диаграмма, текст, План, Технический чертеж

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.5 – Компонентна діаграма

Изображение выглядит как текст, диаграмма, План, Технический чертеж

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.6 – Діаграма розгортування

Важливу роль відіграють і нефункціональні вимоги. Для досягнення стабільної роботи гри необхідно забезпечити продуктивність не нижче 30 кадрів за секунду навіть на середньобюджетних пристроях. Оптимізація графіки, контроль кількості активних об’єктів, використання пулінгу – усе це враховано на рівні архітектурного проєктування. Безпека реалізується через обмежений доступ до файлової системи: збереження здійснюється лише в дозволених системних директоріях. Для резервування прогресу використовуються як PlayerPrefs, так і дублювання у JSON-файли.

Окремо варто відзначити аналіз ризиків. Основні з них – це можлива втрата продуктивності при великій кількості об'єктів, збої при зміні сцен, втрата збережень або відмова UI-елементів. Для мінімізації цих ризиків застосовуються техніки попереднього тестування, логування помилок, дублювання збережень та ієрархічна структура об'єктів у сцені.

Підсумовуючи, архітектура гри Black Prince дозволяє досягти оптимального балансу між простотою реалізації, гнучкістю, стабільністю та розширюваністю. Усі основні модулі мають чітко визначену відповідальність, а їх взаємодія добре ізольована та структурована. Це створює передумови для успішного завершення розробки гри, а також для її подальшого розширення – наприклад, додавання нових режимів, рівнів або системи досягнень без радикальних змін у базовій архітектурі.

3.3 Проектування бази даних

У процесі розробки ігрового проєкту Black Prince було проаналізовано доцільність впровадження повноцінної системи керування базами даних (СКБД). Враховуючи архітектуру гри, її жанр, сценарії використання та формат розгортання, було прийнято рішення не використовувати окрему реляційну або нереляційну базу даних для зберігання даних.

Гра реалізується як однокористувацький мобільний додаток, орієнтований на проходження рівнів, взаємодію з оточенням та суперниками в межах заздалегідь підготовлених сцен. Така структура не потребує обробки великих обсягів динамічних даних, які потребували б централізованого або складного зберігання.

Основні гравцеві дані, як-от: номер останнього пройденого рівня, кількість спроб або перемог, обрані параметри персонажа (на кшталт кольору плаща чи меча), загальна кількість балів – зберігаються локально за допомогою механізмів серіалізації, що надає рушій Unity. Для цієї мети використовуються такі інструменти:

* PlayerPrefs – для простих налаштувань та значень типу int, float, string;
* JSON-файли – для збереження стану гри або даних про прогрес;
* ScriptableObject – для статичних даних (наприклад, конфігурації рівнів або характеристик ворогів).

Таким чином, усе зберігання реалізовано на клієнтському боці, що спрощує архітектуру, зменшує залежності та виключає потребу в налаштуванні додаткових сервісів. Такий підхід дозволяє досягнути оптимального балансу між функціональністю та продуктивністю в межах мобільного середовища.

Варто зазначити, що у випадку розширення гри до багатокористувацького режиму, запровадження системи акаунтів, рейтингової таблиці або хмарного збереження прогресу – структура проєкту може бути доповнена відповідною СКБД, зокрема реляційною базою типу SQLite або хмарними сервісами, такими як Firebase Realtime Database чи Google Play Saved Games API.

Таким чином, у поточному стані проєкту непотрібно проєктувати окрему базу даних, що дозволяє сфокусувати розробку на геймплейній логіці, візуальній складовій та взаємодії з користувачем без додаткових витрат на інфраструктуру зберігання.

3.4 Створення UI/UX.

Розробка гри Black Prince передбачала не лише впровадження основної геймплейної механіки, а й створення зручного, естетично привабливого та інтуїтивного інтерфейсу користувача (UI), що забезпечує комфортний користувацький досвід (UX). Головна мета дизайну UI/UX полягала в тому, щоб надати гравцю чітке розуміння керування, миттєвий зворотний зв’язок на його дії та підтримати загальний художній стиль проєкту – середньовічну бойову атмосферу, наповнену духом героїзму й протистояння.

Уся візуальна частина гри орієнтується на створення атмосфери темного фентезі: кольорова палітра побудована навколо відтінків сталі, крові, темного каменю та золота; кнопки стилізовані під метал або дерево; шрифти – декоративні, але збережено їх читабельність. На фоні головного меню розміщено анімовану сцену з героєм на тлі заходу сонця, що підтримує епічну наративну рамку гри. Звукове оформлення – окрема частина UI/UX, яке включає в себе фонову музику в стилі оркестрового середньовічного маршу, звукові ефекти ударів, втрати здоров’я, перемоги та поразки.

Інтерфейс гри умовно розділений на кілька ключових частин. Головне меню містить кнопки «Грати», «Налаштування», та «Вихід», на рисунку 3.7 схематично показано його.

Изображение выглядит как текст, снимок экрана, Прямоугольник, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.7 – Схематичне позначення головного меню

Після початку гри з’являється HUD – ігровий екран з мінімалістичним набором елементів: індикатори здоров’я гравця та ворога, лічильник очок, кнопка атаки цеглою та спеціальних дій. Приклад геймплею зображено на рисунку 3.8.

Изображение выглядит как Компьютерная игра, снимок экрана, Программное обеспечение для видеоигр, мультфильм

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.8 – Приклад геймплею

Після завершення бою відкривається екран результатів з анімацією перемоги чи поразки, відповідними ефектами та пропозиціями: «Грати ще», «Меню», «Наступний рівень».

Оскільки проєкт орієнтований на мобільні пристрої, особливу увагу приділено зручності керування пальцями, адаптації до різних розмірів екранів та розміщенню кнопок у досяжних для великого пальця зонах. Наприклад, кнопка атаки завжди розташовується у правому нижньому куті, що дозволяє використовувати її великим пальцем, не закриваючи огляд. Усі важливі елементи винесено в кути екрану, щоб не заважати самому бою.

UI реалізовано з використанням Unity UI-системи, включаючи Canvas, EventSystem, TextMeshPro для тексту, а також анімовані переходи та ефекти через систему Animator. Усі елементи збережені у вигляді Prefab-об’єктів для зручності повторного використання. Менеджментом інтерфейсу займається окремий скрипт UIManager.cs, який керує усіма станами – від завантаження до бойової фази та меню поразки або перемоги. Він взаємодіє з іншими компонентами гри через подієву модель: зокрема, слухає події від GameManager, такі як onGameStart, onPlayerDeath, onVictory, onPause.

Особливу роль у формуванні UX відіграє простота навігації, швидкість відгуку та передбачуваність поведінки. Наприклад, усі кнопки забезпечують візуальний і звуковий фідбек при натисканні, а перемикання між сценами супроводжується анімаціями, що створюють плавний перехід і не викликають дратування чи дезорієнтації. Користувач завжди знає, у якому він стані, що є його наступною дією та які наслідки вона матиме.

Таким чином, створення UI/UX у грі Black Prince є невід’ємною частиною загальної архітектури й дизайну проєкту. Правильно побудований інтерфейс не тільки полегшує взаємодію з грою, але й поглиблює занурення в атмосферу, сприяє задоволенню від геймплею та формує позитивний досвід, який гравець забажає повторити.

**4 ОПИС ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ**

У процесі реалізації гри Black Prince було розроблено низку функціональних модулів, кожен з яких відповідає за окремий аспект ігрового процесу: керування, бойову механіку, візуалізацію та логіку. Нижче наведено основні архітектурні рішення, підкріплені прикладами коду та зображеннями реалізації інтерфейсу та логіки гри.

Головне меню є стартовим екраном гри, з якого гравець може розпочати гру, перейти до налаштувань або вийти з програми. Воно реалізоване на основі Canvas із використанням стандартних елементів Unity UI: Button, TextMeshPro, Image. Зовнішній вигляд головного меню представлено на рисунку 4.1.

Изображение выглядит как облако, снимок экрана, небо, текст

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.1 – Головне меню гри Black Prince (виконано самостійно)

Екран налаштувань дозволяє гравцю змінювати основні параметри: гучність музики та звуків, чутливість управління, а також перемикати мову інтерфейсу. Для цього застосовано повзунки Slider, перемикачі Toggle та спадні списки Dropdown. Візуальний вигляд меню налаштувань показано на рисунку 4.2.

Изображение выглядит как текст, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.2 – Меню налаштувань (виконано самостійно)

Під час гри гравець бачить мінімалістичний HUD. Усі елементи адаптивні та масштабуються до різних розширень екрана (див. рис. 4.3).

Изображение выглядит как снимок экрана, Компьютерная игра, Программное обеспечение для видеоигр, мультфильм

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.3 – HUD під час гри (виконано самостійно)

Цеглини є головною бойовою механікою. Клас BlockMovement.cs відповідає за запуск об’єкта з певною силою, обробку траєкторії та колізії. У грі Black Prince система колізій відіграє центральну роль у реалізації фізичної взаємодії між об’єктами – гравцем, ворогами, блоками навколишнього середовища та цеглинами (снарядами). Її реалізація заснована на фізичному рушії Unity (Physics2D), використовуючи компоненти Collider2D, Rigidbody2D та події OnCollisionEnter2D або OnTriggerEnter2D. Щоб визначити, чи може гравець стрибати, використовується raycast або невеликий BoxCollider2D під персонажем (див. рис. 4.4) та для поверхні (див. рис. 4.5).

Изображение выглядит как снимок экрана, мультфильм, Вымышленный персонаж, Программное обеспечение для видеоигр

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.4 – Колізія головного героя (виконано самостійно)

Изображение выглядит как снимок экрана, Прямоугольник, линия, стена

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.5 – Колізія поверхні (виконано самостійно)

Це дозволяє точно виявити, коли гравець стоїть на поверхні, уникаючи стрибків у повітрі.

**private void OnTriggerExit2D(Collider2D collision)**

**{**

**if (collision.gameObject.tag == "ground" || collision.gameObject.tag == "block")**

**{**

**grounded = false;**

**}**

**}**

Коли гравець торкається об’єкта з тегом "ground" або "block", вмикається змінна grounded = true, що дозволяє стрибок

**if (collision.gameObject.tag == "ground")**

**{**

**grounded = true;**

**}**

Також для кращої обробки зіткнення цегли з ворогами та поверхнею вона має свій невеликий BoxCollider2D (див. рис. 4.6)

Изображение выглядит как Прямоугольник, снимок экрана, строительство, строительный материал

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.6 – Колізія цеглин (виконано самостійно)

На рисунку 4.7 представлено момент польоту цеглини під час бою.

Изображение выглядит как снимок экрана, Компьютерная игра, Программное обеспечение для видеоигр, мультфильм

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.7 – Цеглина в польоті у напрямку ворога (виконано самостійно)

Після попадання цеглиною по ворогу, він зникає а цеглина рухаеться далі, до зіткнення з статичною поверхнею. На рисунку 4.8 наведено знищення скелета з мечем та прикріплення кубику до поверхні, для подальшого будування шляху до проходження рівня.

Изображение выглядит как снимок экрана, Компьютерная игра, Программное обеспечение для видеоигр, мультфильм

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.8 – Цеглина в польоті у напрямку ворога (виконано самостійно)

Після завершення гри з’являється відповідне вікно з виграшем (див. рис. 4.9) або програшем (див. рис. 4.10) відповідно до результату гри з кількістю зароблених монет та досвіду.

Изображение выглядит как текст, снимок экрана, графический дизайн, Графика

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.9 – Вікно виграшу (виконано самостійно)

Изображение выглядит как снимок экрана, Красочность, прямоугольный

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.10 – Вікно програшу (виконано самостійно)

Обидва вікна мають анімовану появу (спливання), звук триумфу або поразки, та графічні акценти. Програмно це реалізовано через зіткнення з ворогом або виходом.

**else if(collision.gameObject.tag == "Enemy")**

**{**

**Loose\_window.SetActive(true);**

**}**

**else if (collision.gameObject.tag == "Exit")**

**{**

**thx.SetActive(true);**

**}**

Клас UIManager.cs обробляє зміну екранів гри: головне меню, інтерфейс гри, фінальні вікна перемоги/поразки. Основний елемент побудований на Canvas Unity з підтримкою TextMeshPro.

Ці елементи утворюють повноцінну фізичну модель всередині гри, що поєднує простоту у керуванні з варіативною поведінкою в бою. Такий підхід дозволяє досягти балансу між аркадною динамікою і відчуттям «живого» світу.

**5 ВПРОВАДЖЕННЯ ІГРОВОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

Розробка гри Black Prince на базі Unity передбачала не лише реалізацію основних функціональних механік, але й створення повноцінного візуального досвіду, адаптованого до мобільних платформ. У цьому розділі детально описано підходи до впровадження гри, зокрема процес підготовки до публікації, тестування інтерфейсу та оптимізація UI/UX.

Гра Black Prince з самого початку орієнтована на мобільні пристрої, зокрема Android, як основну цільову платформу. Це пояснюється високою популярністю Android-пристроїв серед користувачів, легкістю розгортання на тестових девайсах та широкими можливостями Unity.

Для побудови Android-збірки було здійснено налаштування у вікні Build Settings, де обрано платформу Android, встановлено відповідну версію API (не нижче 30), активовано IL2CPP як бекенд скриптів, а також увімкнено ARMv7 і ARM64 архітектури для сумісності з більшістю пристроїв.

Особлива увага приділялась створенню та налаштуванню keystore-файлу, що необхідний для цифрового підпису гри перед розгортанням на Google Play. Після формування підписаної .apk збірки гру було завантажено на пристрої через ADB для первинного тестування.

Візуальна частина гри розроблялася з урахуванням необхідності масштабування під різні розміри та співвідношення екранів. Для цього в Unity використано компонент Canvas Scaler з режимом Scale with Screen Size, що дозволяє автоматично адаптувати інтерфейс до поточного розміру дисплея. Крім того, застосовано Safe Area padding для пристроїв з вирізами або панелями.

Усі основні UI-елементи – кнопки, текстові блоки, панелі, HUD – мають адаптивні anchors, pivot та відносні RectTransform, що забезпечує їх правильне розташування незалежно від роздільної здатності.

Було реалізовано кілька ключових сцен, що відповідають за інтерфейс:

* головне меню з кнопками "Грати", "Налаштування", "Вийти";
* меню налаштувань, де гравець може обрати рівень гучності, мову, а також переглянути управління;
* екран виграшу з анімацією перемоги, кількістю зібраних балів і кнопкою "Продовжити";
* екран поразки з повідомленням про невдачу та кнопкою "Повторити спробу".

Для побудови UI було використано TextMeshPro, що забезпечує високу якість шрифтів і гнучке налаштування масштабування тексту. Усі стилі збережено у вигляді окремих стилізованих UI prefab, що дозволяє легко застосовувати їх до різних сцен.

Користувацький досвід значною мірою залежить від миттєвості та якості візуальних реакцій на дії гравця. У грі реалізовано:

* спалах екрану при пошкодженні або смерті персонажа;
* анімацію появи ворожих об'єктів з використанням Animator;
* візуальні ефекти при попаданні цеглиною в супротивника – частинки, коливання спрайтів, колірні зміни;
* перемикання сцен за допомогою fade-ефектів, реалізованих через CanvasGroup і корутини.

Ці елементи не лише покращують естетику, але й допомагають гравцеві краще орієнтуватися у грі.

Фінальний прототип було протестовано на кількох фізичних пристроях з різними характеристиками.

Було перевірено:

* читабельність інтерфейсу;
* відгук кнопок;
* плавність анімацій;
* швидкість завантаження сцен.

Виявлені помилки:

* текст на кнопках іноді обрізався – вирішено через AutoSize;
* кнопки частково перекривалися HUD – перерозподілено layout.

Перед створенням фінальної збірки було оптимізовано:

* розміри спрайтів (через Sprite Packer та компресію);
* кількість draw call (через об'єднання canvas-елементів);
* розміри анімаційних кліпів та їх частоту оновлення;
* мова ресурсів – використано таблиці локалізації (наприклад, англ. та укр. JSON-файли).

Таким чином, процес впровадження гри Black Prince охоплював не лише технічне збирання проєкту, а й комплексну підготовку візуальної частини до адаптивного та масштабованого використання. Завдяки Unity та продуманому дизайну, інтерфейс гри став важливою складовою геймплею, забезпечуючи зручність, інтуїтивність і естетичну привабливість.

**ВИСНОВКИ**

У ході розробки мобільної гри Black Prince було здійснено повний цикл створення ігрового програмного забезпечення – від аналізу предметної області, формування вимог та архітектурного проєктування до реалізації, тестування та створення супровідної документації. Кожен з учасників комплексного курсового проєкту взяв на себе окрему функціональну роль, що забезпечило ефективний поділ праці та якісну реалізацію всіх складових гри.

Я відповідав за візуальну складову гри – проєктування та реалізацію UI/UX в Unity, створення інтерактивних екранів (головного меню, екранів виграшу, програшу, налаштувань тощо), адаптацію інтерфейсу під різні мобільні екрани, налаштування анімацій, управління подіями UI через UIManager, EventSystem, а також за візуальні ефекти через RenderEngine. Крім цього, було протестовано зручність інтерфейсу на кількох пристроях.

Станіслав Логвін реалізував ігрову логіку в середовищі Unity: модулі керування гравцем (PlayerMovement, CharacterController), взаємодію з ігровими об’єктами (Brick, BlockMovement), поведінку супротивників (EnemyAI), обробку колізій (CollisionDetector) та загальний контроль над ігровим процесом (GameEngine, LevelManager). Забезпечив сценарії боїв, логіку перезапуску рівнів та обробку станів гри (старт, пауза, завершення тощо).

Ігор Балицький розробив вебсторінку, присвячену грі, що містить її опис, візуальні матеріали, демонстраційне відео та можливість завантаження APK-файлу. Забезпечив адаптивність вебінтерфейсу, сумісність з мобільними браузерами та доступність посилань на ресурси гри. Це дало змогу забезпечити презентабельність проєкту, а також легке поширення серед потенційних користувачів або інвесторів.

У рамках роботи було виконано:

* глибокий аналіз потреб мобільної цільової аудиторії та сценаріїв використання;
* формування функціональних та нефункціональних вимог;
* створення концепції гри та геймплейних механік;
* проєктування програмного забезпечення за допомогою UML-діаграм: Use Case, Sequence, Component, Deployment;
* реалізація UI/UX на основі сучасних принципів юзабіліті;
* розробка основної ігрової механіки, об’єктів, поведінки супротивників;
* тестування та створення стабільної збірки для Android;
* розробка вебресурсу для презентації проєкту.

Unity став ключовим інструментом у реалізації, забезпечуючи злагоджену взаємодію візуальної частини та логіки гри. Висока гнучкість архітектури дає можливість масштабувати застосунок, розширюючи його новими функціями, ігровими режимами або підтримкою мережевої гри.

Таким чином, поставлену задачу реалізовано повністю: створено ігрову програму з акцентом на зручність, естетичність та адаптацію під мобільні пристрої. Результат розробки може бути використаний як базовий прототип для майбутніх мобільних ігор з подібною архітектурою.

**ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ**

1.Newzoo. Global Games Market Report 2023 [Електронний ресурс]. – URL: <https://newzoo.com/resources/trend-reports/newzoos-global-games-market-report-2024-free-version> (дата звернення: 23.03.2022)

2.Литвин В. М. Основи програмування комп’ютерних ігор. – Львів: Бескид-Біт, 2020. – 312 с.

3.Степаненко І. Г. Основи дизайну інтерфейсів. – Харків: ХНУРЕ, 2022. – 184 с.

4.Deterding S., Dixon D., Khaled R., Nacke L. From Game Design Elements to Gamefulness: Defining Gamification // In: *Proceedings of the 15th International Academic MindTrek Conference*, 2011. – P. 9–15.

5.Norman D. The Design of Everyday Things. – MIT Press, 2013. – 368 p.

6.UX Collective. Mobile game UX design best practices [Електронний ресурс]. – URL: <https://uxdesign.cc/boost-ux-with-mobile-ux-design-principles-and-best-practices-907e4f9fdd5d> (дата звернення: 24.03.2022)

7.Unity Technologies. Unity Manual [Електронний ресурс]. – URL: <https://docs.unity3d.com/Manual/index.html> (дата звернення: 24.03.2022)

8.Galitz W. O. The Essential Guide to User Interface Design. – Wiley, 2017. – 512 p.

9.Глушков В. М. Основи комп’ютерної графіки. – К.: Видавництво Ліра-К, 2017. – 248 с.

10.Statista. Mobile games revenue worldwide 2023 [Електронний ресурс]. – URL: <https://www.statista.com/statistics/748044/mobile-games-revenue-worldwide/> (дата звернення: 25.03.2022)