МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет «Острозька академія»

Навчально науковий інститут ІТ та бізнесу

Кафедра інформаційних технологій та аналітики даних

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня магістра
на тему: «**Управління проєктом розробки мобільної гри на рушії Unreal Engine»**

Виконав: студент 2 курсу, групи МУП-2
другого (магістерського) рівня вищої освіти
спеціальності 122 Комп'ютерні науки
ОПП «Управління проєктами»
Синиця Дмитро Олескандрович

Керівник: ***Місай В.В., викладач, фахівець-практик кафедри ІТБ***

Рецензент: *кандидат технічних наук, доцент, доцент кафедри прикладної математики Донецького національного університету імені Василя Стуса*

Загоруйко Любов Василівна

РОБОТА ДОПУЩЕНА ДО ЗАХИСТУ

Завідувач кафедри інформаційних технологій та аналітики даних
(проф., д.е.н. Кривицька О.Р.)

Протокол № 5 від 04 грудня 2025 р.

Острог, 2025

АНОТАЦІЯ

кваліфікаційної роботи

на здобуття освітнього ступеня магістра

Тема: *Управління проєктом розробки мобільної гри на рушії Unreal Engine.*

Автор: Синиця Дмитро Олександрович

Науковий керівник: *Місай В.В., викладач, фахівець-практик кафедри ІТБ*

Захищена «……»…..... 2025 року.

Пояснювальна записка до кваліфікаційної роботи: *с.84, рис. 7, джерела: 39.*

Ключові слова: *управління проєктом; мобільна гра; розробка гри; Unreal Engine; ігрові механіки; 3D-моделі; користувацький інтерфейс; тестування та оптимізація; командна робота; планування проєкту.*

Короткий зміст праці:

Магістерська робота присвячена дослідженню та практичній реалізації процесу управління проєктом мобільної гри на рушії Unreal Engine. У роботі проаналізовано сучасні підходи до організації та планування розробки ігор, використання інструментів управління проєктами та командної взаємодії. Розглянуто можливості Unreal Engine 5, Blender, Adobe Photoshop та Adobe Illustrator для створення ігрових механік, моделей персонажів і середовища, користувацького інтерфейсу та інтерактивних елементів.

У практичній частині роботи розроблено прототип мобільної гри, реалізовано основні механіки взаємодії та навігації, створено концепт-дизайни та 3D-моделі, а також проведено тестування продуктивності та стабільності. Окрему увагу приділено контролю виконання завдань, плануванню етапів розробки та оптимізації процесу управління проєктом. Результати роботи можуть бути використані для подальшої розробки мобільних ігор та вдосконалення практик управління проєктами у сфері ігрової індустрії.

ANNOTATION

of a qualification paper

for a master's degree

Theme: *Project management of a mobile game development project using Unreal Engine.*

Author: *Dmytro Sinitsa*

Scientific supervisor: *Misai V.V., lecturer, specialist-practitioner at the ITB department*

Defended «……»… .....of 2025.

Explanatory note to the qualification work: *p.80, pic.7, sources: 39.*

Keywords: *project management; mobile game; game development; Unreal Engine; game mechanics; 3D models; user interface; testing and optimization; teamwork; project planning.*

Summary of the paper:

The master's thesis is devoted to the research and practical implementation of the process of managing a mobile game project using the Unreal Engine engine. The work analyzes modern approaches to organizing and planning game development, using project management tools, and team interaction. The capabilities of Unreal Engine 5, Blender, Adobe Photoshop, and Adobe Illustrator for creating game mechanics, character and environment models, user interface, and interactive elements are considered. In the practical part of the work, a prototype of a mobile game was developed, the main interaction and navigation mechanics were implemented, concept designs and 3D models were created, and performance and stability testing was conducted. Special attention is paid to task performance control, planning development stages, and optimizing the project management process. The results of the work can be used for further development of mobile games and improving project management practices in the gaming industry.

ЗМІСТ

ВСТУП

Розробка мобільних ігор сьогодні є однією з найшвидше зростаючих сфер цифрової індустрії, що поєднує творчий дизайн, сучасні технології та потреби користувачів у високоякісному інтерактивному контенті. Високі вимоги до графіки, ігрового процесу та інтерактивності зумовлюють необхідність комплексного підходу до планування та організації розробки. Кожен проєкт мобільної гри потребує ефективного управління, що включає координацію роботи команди, контроль виконання завдань, тестування функціональних можливостей та оптимізацію продуктивності на різних мобільних пристроях.

Особливу роль у цьому процесі відіграє застосування сучасних інструментів розробки, таких як Unreal Engine 5, який дозволяє реалізувати високоякісну графіку, фізичну взаємодію об’єктів та інтерактивні сценарії. Для створення графічного контенту та 3D-моделей використовуються програми Blender, Adobe Photoshop та Illustrator, що дозволяє досягти високого рівня деталізації персонажів, середовища та користувацького інтерфейсу.

Поєднання технічної реалізації гри та організації процесу управління проєктом дозволяє забезпечити не лише якість кінцевого продукту, а й своєчасне виконання всіх етапів розробки, ефективне розподілення завдань між членами команди та мінімізацію ризиків. Крім того, правильна організація управління проєктом сприяє оптимізації робочих процесів, підвищенню продуктивності команди, поліпшенню користувацького досвіду та створенню конкурентоспроможного продукту на ринку мобільних ігор.

Мобільні ігри сьогодні займають провідне місце в індустрії цифрових розваг і формують нові стандарти взаємодії користувача з контентом. Зростання попиту на високоякісні інтерактивні продукти, що поєднують реалістичну графіку, динамічний геймплей та зручний користувацький інтерфейс, вимагає від розробників комплексного підходу до організації процесу створення ігор.

Актуальність - обумовлена необхідністю поєднання технічних та організаційних аспектів розробки. Ефективне управління проєктом дозволяє не

лише забезпечити своєчасне виконання завдань, а й оптимізувати роботу команди, координувати взаємодію між різними спеціалістами, мінімізувати потенційні ризики та підвищити якість кінцевого продукту на всіх етапах розробки.

Особливо важливим є впровадження сучасних інструментів та методологій управління, таких як Agile, Scrum та Kanban, які дозволяють гнучко реагувати на зміни в проєкті, ефективно планувати спринти та оцінювати прогрес виконання завдань. Крім того, використання професійних програмних засобів для створення графічного контенту та реалізації ігрових механік, зокрема Unreal Engine 5, Blender, Adobe Photoshop та Illustrator, дає змогу розробляти високоякісні 3D-моделі персонажів і середовища, інтерактивні елементи та користувацький інтерфейс, що відповідає сучасним вимогам гравців.

Поєднання організаційних та технічних підходів сприяє підвищенню продуктивності команди, більш ефективному розподілу ресурсів та створенню мобільних ігор, які відзначаються високим рівнем інтерактивності, стабільністю роботи та привабливим геймплеєм, що відповідає сучасним стандартам індустрії.

Метою магістерської роботи є дослідження та практична реалізація процесу управління проєктом розробки мобільної гри на рушії Unreal Engine 5, що передбачає комплексне поєднання технічних та організаційних аспектів розробки. Основна увага приділяється не лише створенню ігрового контенту та інтерактивних механік, а й ефективному плануванню, організації командної роботи, контролю виконання завдань і оцінці результатів на всіх етапах проєкту.

Досягнення мети передбачає вирішення таких ключових завдань:

1. **Аналіз сучасних підходів до розробки мобільних ігор** – дослідження актуальних тенденцій у жанрах, механіках, дизайні інтерфейсу та інтерактивності.

2. **Дослідження методологій управління проєктами** – вивчення сучасних інструментів та практик управління командою, планування та контролю виконання завдань (Agile, Scrum, Kanban).

3. **Проєктування ігрових механік та користувацького інтерфейсу** – створення концепту та прототипу гри, розробка інтерактивних систем, навігації персонажа, діалогової системи та інвентарю з використанням Blueprints і C++ API.

4. **Створення графічного та 3D-контенту** – моделювання персонажів, об’єктів та ігрового середовища, розробка текстур, анімацій та елементів інтерфейсу за допомогою Blender, Adobe Photoshop та Illustrator.

5. **Реалізація та інтеграція компонентів у рушії Unreal Engine 5** – програмування ігрових механік, інтерактивних об’єктів та систем взаємодії, налаштування рівнів, анімацій і користувацького інтерфейсу.

6. **Тестування та оптимізація проєкту** – перевірка стабільності роботи гри, продуктивності на різних мобільних пристроях, виявлення помилок та вдосконалення ігрового процесу.

7. **Оцінка ефективності управління проєктом** – аналіз організації командної роботи, контролю виконання завдань та управління ризиками, підготовка рекомендацій щодо оптимізації процесу розробки.

Завдання дослідження:

Проаналізувати сучасні підходи до розробки мобільних ігор – дослідити актуальні тенденції у жанрах, ігрових механіках, дизайні інтерфейсу та інтерактивності.

Вивчити методології управління проєктами в ігровій індустрії – оцінити застосування Agile, Scrum, Kanban та інших інструментів організації командної роботи та контролю виконання завдань.

Розробити концепт та технічне завдання мобільної гри – визначити структуру проєкту, ключові механіки та функціональні вимоги.

Створити графічний та 3D-контент для гри – моделювання персонажів, об’єктів та середовища, розробка текстур, анімацій та елементів користувацького інтерфейсу за допомогою Blender, Adobe Photoshop та Illustrator.

Реалізувати ігрові механіки та інтерактивні системи у Unreal Engine 5 – налаштування поведінки персонажів, інтерактивних об’єктів, діалогів, інвентарю та навігаційних систем.

Протестувати та оптимізувати гру – перевірка стабільності, продуктивності та якості взаємодії користувача на різних мобільних пристроях.

Оцінити ефективність управління проєктом – аналіз організації роботи команди, контролю виконання завдань, управління ризиками та підготовка рекомендацій щодо підвищення ефективності розробки.

Об’єктом дослідження у даній магістерській роботі є процес управління проєктом розробки мобільної гри на рушії **Unreal Engine 5**, який охоплює комплексну організацію роботи команди розробників, планування завдань, координацію та контроль усіх етапів створення ігрових механік, графічного контенту, інтерактивних систем, а також тестування, налагодження та оптимізацію кінцевого продукту. Цей процес включає не лише технічну реалізацію ігрового контенту, а й управління ресурсами, термінами виконання завдань, взаємодію між різними підрозділами команди та забезпечення ефективної комунікації між усіма учасниками проєкту.

Вивчення об’єкта дослідження спрямоване на виявлення ефективних методів управління проєктами мобільних ігор, які дозволяють забезпечити високу якість ігрового контенту, своєчасне виконання всіх завдань, оптимальне використання людських і матеріальних ресурсів та мінімізацію ризиків під час розробки. Особлива увага приділяється інтеграції сучасних методів планування, контролю та комунікації в команді, застосуванню професійних інструментів і технологій для моделювання, анімації, текстурування та програмування, що забезпечує стабільну роботу гри на різних мобільних платформах, відповідність сучасним вимогам користувачів, а також конкурентоспроможність кінцевого продукту на ринку.

Предметом дослідження є організаційні та технологічні процеси управління розробкою мобільної гри на рушії Unreal Engine 5, що включають планування етапів проєкту, розподіл завдань та ролей у команді, контроль

виконання завдань і управління ресурсами. Особливу увагу приділено взаємодії між розробниками, дизайнерами та тестувальниками, а також інтеграції графічного контенту, анімацій, ігрових механік та користувацького інтерфейсу у межах одного проєкту.

Предмет дослідження охоплює методи організації командної роботи, застосування сучасних методологій управління проєктами (Agile, Scrum, Kanban), використання інструментів для контролю прогресу та комунікації, а також технології створення та оптимізації контенту (Blender, Adobe Photoshop, Illustrator). Особливий акцент робиться на забезпеченні ефективної взаємодії між усіма компонентами розробки для досягнення високої якості гри, стабільної роботи на мобільних пристроях і відповідності вимогам користувачів та ринку.

Предмет дослідження дозволяє вивчати вплив управлінських рішень на продуктивність команди, ефективність використання ресурсів, дотримання термінів виконання завдань та загальну якість кінцевого продукту.

Методи дослідження

Для досягнення мети магістерської роботи я використав комплекс методів, які дозволяють всебічно дослідити процес управління розробкою мобільної гри на рушії Unreal Engine 5. Вони охоплюють моделювання, аналітичний та проєктний підходи, експериментальну перевірку ігрових механік, оцінку користувацького досвіду, порівняльний аналіз та статистичну обробку результатів. Завдяки цьому можна оцінити ефективність організації роботи команди, інтеграції контенту та управління ресурсами на всіх етапах розробки.

1. **Метод моделювання** – використовується для створення прототипів рівнів, систем взаємодії об’єктів, діалогів та інвентарю, а також для перевірки логіки роботи ігрових механік перед їх остаточною реалізацією.

2. **Аналітичний метод** – передбачає вивчення сучасних підходів до розробки мобільних ігор, аналіз існуючих методологій управління проєктами (Agile, Scrum, Kanban), дослідження практик планування, організації та контролю командної роботи.

3. **Експериментальний метод** – полягає у практичній реалізації мобільної гри, інтеграції графічного контенту, моделей персонажів та середовища, а також тестуванні ігрових механік і продуктивності гри на різних мобільних пристроях.

4. **Метод проєктування** – застосовується для створення концепту гри, розробки технічного завдання, ігрових механік, інтерактивних систем та користувацького інтерфейсу, що дозволяє формалізувати процес розробки та визначити послідовність реалізації завдань.

5. **Анкетування та опитування користувачів** – використовується для оцінки користувацького досвіду та сприйняття розробленого продукту, що дає змогу визначити сильні та слабкі сторони реалізованих механік та інтерфейсу.

6. **Метод порівняльного аналізу** – дозволяє оцінити ефективність різних методів управління та інструментів розробки, порівняти результати тестування продуктивності, стабільності та зручності користувацького інтерфейсу.

7. **Статистичний та якісний аналіз** – передбачає обробку результатів тестування, оцінку продуктивності гри, стабільності роботи та взаємодії користувача з ігровим середовищем для формування рекомендацій щодо оптимізації та покращення проєкту.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ МОБІЛЬНОЇ ІГРОВОЇ РОЗРОБКИ ТА СУЧАСНІ МЕТОДИКИ УПРАВЛІННЯ.

1.1. Сучасні тренди та жанри мобільних ігор

Сьогодні розробка мобільних ігор є однією з найдинамічніших сфер цифрової індустрії, що швидко розвивається завдяки постійному зростанню кількості користувачів мобільних пристроїв та підвищенню вимог до ігрового досвіду. Основні тенденції розвитку мобільних ігор пов’язані з інтерактивністю, соціальною взаємодією, високою якістю графіки, глибокими сюжетними лініями та адаптацією під різні платформи.

Одним із головних трендів є зростання популярності кросплатформених ігор, які дозволяють гравцям взаємодіяти незалежно від типу пристрою. Це стимулює розробників до створення ігор з уніфікованими механіками та інтерфейсами, що забезпечують комфортний досвід на смартфонах, планшетах та іноді навіть на ПК [1].

Ще одним важливим напрямом є соціальна інтеграція та багатокористувацький режим, який включає кооперативні завдання, змагання та обмін ресурсами між гравцями. Такі механіки підвищують залученість користувачів і стимулюють довготривалу взаємодію з продуктом.

З точки зору жанрової класифікації, мобільні ігри можна поділити на кілька основних груп:

1. Казуальні ігри – прості за механікою та доступні для широкого кола користувачів. Вони зазвичай мають короткі ігрові сесії, зрозумілі правила та привабливу графіку.
2. Стратегічні ігри – передбачають планування ресурсів, управління об’єктами або армією, а також розробку тактик для досягнення цілей. Цей жанр користується популярністю серед гравців, які цінують глибину ігрового процесу.

3. Рольові ігри (RPG) – фокусуються на розвитку персонажів, сюжетній лінії та взаємодії з навколишнім світом. Часто передбачають систему квестів та прокачування навичок.

4. Пригодницькі та квестові ігри – орієнтовані на проходження сюжетних завдань, вирішення головоломок та дослідження віртуального світу. Вони поєднують інтерактивність із логічними викликами для гравця.

5. Симулятори та спортивні ігри – дозволяють моделювати реальні процеси або спортивні змагання, пропонуючи високий рівень деталізації та реалістичності.

6. Екшн та шутери – швидкісні ігри з акцентом на рефлекси, реакцію та координацію дій гравця, часто включають мультиплеєрні режими.

Сучасні тренди мобільних ігор також включають використання технологій доповненої та віртуальної реальності, інтеграцію зі стрімінговими платформами та застосування систем штучного інтелекту для створення адаптивного ігрового досвіду. Крім того, велике значення набуває монетизація через мікротранзакції та внутрішньоігрові покупки, що визначає економічну модель багатьох сучасних проєктів.

Врахування цих тенденцій та жанрових особливостей є важливим при розробці мобільних ігор, оскільки дозволяє створювати конкурентоспроможні продукти, які відповідають сучасним вимогам ринку та очікуванням користувачів [2].

Ще одним важливим трендом є адаптація ігор під різні мобільні пристрої та платформи, що дозволяє забезпечити уніфікований ігровий досвід на смартфонах з різними розмірами екранів, планшетах і навіть на ПК. Це стимулює використання кросплатформових рушіїв, таких як Unreal Engine 5, і забезпечує збереження функціональності та продуктивності гри на різних пристроях.

Серед сучасних технологічних новацій виділяють впровадження доповненої (AR) та віртуальної реальності (VR), що дозволяє гравцям занурюватися у повністю інтерактивне середовище, підвищуючи залученість і емоційний ефект від ігрового процесу. Також активно застосовуються системи

штучного інтелекту, які адаптують поведінку NPC, підлаштовують складність завдань під рівень гравця і створюють динамічний та персоналізований ігровий досвід.

Важливим аспектом сучасних мобільних ігор є економічна модель проектів**,** що включає мікротранзакції, внутрішньоігрові покупки та рекламні механіки. Це дозволяє не лише монетизувати продукт, а й стимулювати гравців до активної взаємодії та розвитку персонажів чи ігрових об’єктів [3].

Також варто зазначити зростаючу роль аналітики даних у мобільній ігровій індустрії. Розробники використовують інструменти збору та обробки даних про поведінку користувачів, що дозволяє оцінювати ефективність ігрових механік, оптимізувати баланс, передбачати відтік гравців та покращувати монетизаційні стратегії.

Урахування цих тенденцій та інноваційних підходів дозволяє створювати сучасні мобільні ігри, які є конкурентоспроможними на глобальному ринку, відповідають очікуванням користувачів та забезпечують високу якість ігрового досвіду [4].

1.2. Особливості розробки квестових ігор для мобільних платформ

Квестові ігри на мобільних платформах мають низку специфічних особливостей, які значною мірою визначають підхід до їх розробки та реалізації. По-перше, це оптимізація гри під обмежені ресурси пристроїв, таких як процесор, оперативна пам’ять та енергоспоживання. Мобільні пристрої значно поступаються по продуктивності ПК та консолям, тому розробник має ретельно продумувати баланс між високою якістю графіки, плавністю анімацій та ефектами, щоб забезпечити стабільну роботу гри на різних моделях смартфонів та планшетів.

По-друге, інтерфейс користувача та система управління повинні бути максимально інтуїтивними та адаптованими для сенсорного екрану. Гравець повинен легко орієнтуватися в меню, взаємодіяти з об’єктами середовища та

виконувати завдання без зайвих складнощів. Особлива увага приділяється дизайну кнопок, жестів та системі підказок, що допомагає гравцеві орієнтуватися у складних ситуаціях і підтримує плавний ігровий процес [5].

По-третє, сюжетна складова та інтерактивність займають ключове місце у квестових іграх. Сюжет повинен бути логічно структурованим, цікавим і доступним для розуміння під час коротких ігрових сесій, характерних для мобільних користувачів. Важливим елементом є включення логічних задач, головоломок та інтерактивних об’єктів, які стимулюють мислення, заохочують до експериментів і дослідження ігрового світу, а також підвищують залученість та мотивацію гравця [6].

При розробці мобільних квестових ігор слід враховувати адаптацію контенту під різні розміри екранів та роздільну здатність, що забезпечує однаково комфортний досвід на різних пристроях. Не менш важливим є інтегрування звуку, музичного супроводу, анімацій та візуальних ефектів, які формують атмосферу гри та підсилюють емоційне занурення користувача у сюжет.

Ще одним важливим аспектом є баланс між складністю завдань та доступністю гри для широкого кола користувачів. Завдання повинні бути достатньо цікавими та різноманітними, щоб утримувати увагу гравця, але водночас не перевантажувати його надмірною складністю, яка може викликати фрустрацію [7].

Також слід зазначити роль тестування та збору зворотного зв’язку від користувачів у процесі розробки. Аналіз реакцій гравців дозволяє оптимізувати рівні, підказки, складність завдань, а також покращити навігацію та управління, що в кінцевому результаті підвищує якість ігрового досвіду.

Всі ці аспекти разом формують унікальний ігровий досвід**,** який робить мобільні квестові ігри цікавими, інтерактивними та привабливими для широкого кола користувачів, забезпечує високий рівень залучення і дозволяє гравцям отримувати задоволення від проходження квестів та дослідження віртуального світу [8].

1.3. Основи проєктування ігрових механік

Проєктування ігрових механік є ключовим етапом розробки будь-якої мобільної гри, оскільки саме механіки визначають взаємодію гравця з ігровим світом, рівень залученості та загальний користувацький досвід. Геймплей формується на основі правил, завдань, обмежень та реакцій системи на дії користувача, а ефективне проєктування механік дозволяє створювати інтерактивні, цікаві та зрозумілі ігри.

Першим кроком у розробці є визначення цілей та ролі гравця, а також того, які дії він повинен виконувати для досягнення цих цілей. Важливо, щоб дії були логічно пов’язані з сюжетом та мотивували користувача досліджувати світ, вирішувати задачі та взаємодіяти з іншими персонажами чи об’єктами [9].

Другим аспектом є побудова системи правил та обмежень, які регулюють поведінку гравця і навколишнього середовища. Це включає управління ресурсами, часові обмеження, складність завдань та умови прогресу. Правильно спроєктовані правила забезпечують баланс між складністю і доступністю гри, підвищують інтерес гравців та стимулюють повторне проходження.

Не менш важливим є проектування взаємодії з ігровим середовищем та об’єктами, що включає як механіки дослідження, так і системи інтерактивності. В квестових іграх особлива увага приділяється пошуку предметів, виконанню завдань, використанню інвентарю, а також впливу дій гравця на розвиток сюжету та реакцію персонажів.

Також необхідно враховувати психологічні аспекти геймплею, такі як винагорода, прогресія, елементи несподіванки та змагальні або соціальні стимули. Це допомагає підтримувати інтерес гравця, формує мотивацію до активної взаємодії з грою та підвищує її затягуючий ефект [10].

У сучасній мобільній розробці велике значення має адаптивність механік під платформу. Мобільні пристрої мають обмежену продуктивність та різні формати екранів, тому механіки повинні бути оптимізованими для сенсорного управління, коротких ігрових сесій та зручної навігації.

На практиці проєктування ігрових механік передбачає створення прототипів та тестування на ранніх етапах розробки. Це дозволяє оцінити ефективність механік, виявити слабкі місця та внести необхідні зміни до фінальної версії гри. Інструменти, такі як Unreal Engine 5 та його система Blueprints, дають змогу швидко реалізовувати та модифікувати механіки без потреби глибокого програмування, що значно прискорює процес розробки та покращує якість кінцевого продукту [11].

В цілому, основи проєктування ігрових механік поєднують у собі технічну, логічну та психологічну складові, і правильне їх врахування є запорукою створення цікавого, інтерактивного та конкурентоспроможного мобільного продукту.

1.4. Теоретичні основи інтерактивності та динаміки ігрового процесу

Інтерактивність та динаміка складають основу будь-якої мобільної гри, оскільки визначають характер взаємодії гравця з віртуальним середовищем і впливають на загальний досвід користувача. Інтерактивність передбачає здатність системи реагувати на дії гравця, пропонувати альтернативні шляхи розвитку подій, адаптувати складність завдань та формувати відчуття контролю. Вона включає моделі взаємодії з об’єктами середовища, механіки активації подій, систему підказок та керування інвентарем або ресурсами [12].

Ключовим аспектом інтерактивності є зворотний зв’язок, який гравець отримує у відповідь на свої дії. Це можуть бути візуальні або аудіо сигнали, зміни стану персонажів чи об’єктів, повідомлення про виконання завдань чи прогрес у квесті. Правильна організація зворотного зв’язку дозволяє гравцю зрозуміти наслідки своїх дій, підвищує залученість та мотивує продовжувати гру.

Динаміка ігрового процесу охоплює такі елементи, як розвиток персонажа, зміну складності завдань, прогресію рівнів, систему винагород і покарань, а також адаптацію сценаріїв під стиль та швидкість гри конкретного користувача

[13]. Вона забезпечує відчуття розвитку та досягнень, підтримує інтерес гравця протягом усієї гри і сприяє повторним сеансам взаємодії.

Важливим елементом є баланс між складністю та доступністю. Завдання повинні бути достатньо складними, щоб стимулювати мислення і логіку, але водночас не перевантажувати гравця. Під час мобільних сесій, які часто короткі, надмірна складність може призвести до втрати інтересу. Тому інтерактивність і динаміка повинні бути оптимізовані під тривалість та формат мобільних ігрових сесій.

Також теоретично обґрунтовується вплив інтерактивності на мотивацію та поведінку користувача**.** Використання систем прогресії, досягнень, накопичення ресурсів або розвитку персонажа створює внутрішню мотивацію гравця. Включення елементів несподіванки, різноманітних викликів та адаптивних сценаріїв дозволяє підтримувати психологічну зацікавленість та підсилює емоційне занурення [14].

У мобільних квестових іграх інтерактивність поєднується з сюжетом та логікою завдань, формуючи єдиний цілісний досвід. Ретельне планування взаємодії гравця з об’єктами, системою підказок, головоломками та NPC дозволяє створювати захопливі сценарії, які стимулюють дослідження світу гри та підвищують задоволення від проходження квестів.

Таким чином, теоретичні основи інтерактивності та динаміки ігрового процесу визначають правила проєктування механік, формують логіку взаємодії гравця з грою та забезпечують високий рівень залученості і користувацького досвіду у мобільних іграх [15].

1.5. Методології управління IT-проєктами: Agile, Scrum, Kanban

Ефективне управління проєктом мобільної гри передбачає використання сучасних методологій, які забезпечують гнучкість, контроль виконання завдань та координацію роботи команди. Найпоширенішими підходами є Agile, Scrum та

Kanban, кожен з яких має свої особливості та переваги у контексті розробки мобільних ігор.

Agile - це гнучка методологія, яка дозволяє швидко адаптувати процес розробки до змінних вимог та умов проєкту. Основна ідея Agile полягає в ітеративній розробці та регулярному отриманні зворотного зв'язку, що дає змогу своєчасно коригувати механіки, сюжетні елементи та графічний контент гри. Вона сприяє більш ефективній комунікації між розробниками, дизайнерами та тестувальниками, забезпечує прозорість процесу та дозволяє швидко реагувати на нові ідеї або технічні виклики [16].

Scrum - практична реалізація Agile, що включає структуру спринтів, ролі команди (Product Owner, Scrum Master, Development Team) та регулярні зустрічі (Daily Scrum, Sprint Planning, Sprint Review, Retrospective). У розробці мобільної гри Scrum дозволяє розбивати проєкт на короткі ітерації, визначати пріоритети завдань, оцінювати продуктивність команди та контролювати прогрес виконання. Це особливо корисно при створенні квестових ігор, де постійно змінюються сценарії, механіки та рівні.

Kanban - метод візуалізації завдань за допомогою дошок і карток, який допомагає відстежувати стан роботи над різними елементами проєкту. Kanban дозволяє швидко визначати вузькі місця, балансувати навантаження та підвищувати продуктивність команди. У мобільній розробці Kanban часто використовується для контролю виконання графічних завдань, тестування механік та інтеграції контенту в рушій Unreal Engine [17].

Використання цих методологій у поєднанні дозволяє **підвищити ефективність управління проєктом**, забезпечити своєчасне виконання завдань, поліпшити комунікацію в команді та мінімізувати ризики, що виникають під час розробки мобільних ігор. Вибір конкретного підходу залежить від розміру команди, складності проєкту та специфіки механік, що реалізуються у грі.

1.6. Інструменти контролю виконання та командної взаємодії

Для успішної розробки мобільної гри важливим є застосування інструментів, що дозволяють контролювати виконання завдань та організовувати взаємодію в команді. До таких інструментів відносяться системи управління проєктами (Jira, Trello, Asana), які дають змогу планувати завдання, відстежувати прогрес, встановлювати пріоритети, розподіляти ролі та вести документацію про виконані етапи [18].

Особливе значення має **координація роботи команди у рамках рушія Unreal Engine 5**. UE5 надає вбудовані засоби для командної взаємодії та контролю версій, такі як **Source Control**, що дозволяє кільком розробникам одночасно працювати над одним проєктом, зберігати історію змін, уникати конфліктів у файлах та координувати інтеграцію контенту. Це особливо важливо при роботі над складними квестовими іграми, де одночасно створюються 3D-моделі, анімації, графіка та ігрові механіки.

Крім того, UE5 містить інструменти для **візуального контролю ігрових механік та рівнів**, такі як Level Sequencer, Blueprint Debugger і Live Preview. Вони дозволяють перевіряти роботу механік у реальному часі, оцінювати взаємодію елементів середовища та виявляти помилки на ранніх етапах розробки [19].

Також для командної взаємодії використовуються **системи спільного документування та комунікації**, такі як Confluence, Slack або Microsoft Teams, що забезпечують обмін інформацією, узгодження завдань, обговорення змін та швидке вирішення проблем.

Використання таких інструментів дозволяє забезпечити **своєчасне виконання завдань, прозорість процесів, ефективну комунікацію всередині команди** та стабільну інтеграцію всіх елементів мобільної гри. Це підвищує якість кінцевого продукту, зменшує ризики виникнення технічних помилок та забезпечує оптимальну організацію процесу розробки [20].

1.7. Використання Unreal Engine 5 у мобільній розробці

Сучасна розробка мобільних ігор вимагає застосування потужних інструментів, які дозволяють поєднувати високу графічну якість, складні ігрові механіки та інтерактивні системи. Unreal Engine 5 забезпечує такі можливості, пропонуючи розробникам широкий спектр засобів для створення 3D-моделей, анімацій, фізичних систем, освітлення та звукового супроводу, а також ефективні методи оптимізації продуктивності під обмежені ресурси мобільних пристроїв. Теоретичне розуміння архітектури рушія дозволяє ефективно планувати структуру проєкту, передбачати взаємодію між компонентами та оптимально розподіляти ресурси команди.

Однією з ключових переваг UE5 є робота з **Blueprints**, що дає змогу створювати логіку ігрових механік за допомогою візуального програмування. Це прискорює процес прототипування, полегшує інтеграцію нових функцій і зменшує залежність від глибокого програмування. Паралельно застосування **C++ API** забезпечує гнучкість та масштабованість для реалізації складних систем, які потребують точного контролю над процесами гри, даними та інтеграцією сторонніх сервісів [21].

Для досягнення високої візуальної якості мобільних ігор рушій пропонує технології **Nanite** та **Lumen**, що дозволяють деталізувати геометрію об’єктів і реалізувати реалістичне динамічне освітлення без значного впливу на продуктивність пристроїв. Це створює ефект занурення та покращує користувацький досвід. Додатково UE5 надає інструменти для створення ефектів частинок (**Niagara**), анімацій персонажів (**Control Rig**) та кінематографічних сцен (**Sequencer**), що дозволяє поєднувати інтерактивність із високою художньою деталізацією [22].

Рушій також інтегрується з системами контролю версій та платформами для командної взаємодії, що забезпечує прозорий розподіл ролей і завдань серед учасників проєкту. Це дозволяє відстежувати виконання кожного завдання в реальному часі, контролювати зміни в коді та ресурсах, а також оперативно

виявляти та усувати помилки на будь-якому етапі розробки. Використання таких систем сприяє підвищенню дисципліни та відповідальності в команді, дозволяє уникати дублювання роботи та забезпечує ефективну координацію між дизайнерами, програмістами, аніматорами та тестувальниками. Крім того, інтеграція з такими платформами допомагає планувати ресурси, контролювати дотримання термінів, здійснювати аудит якості та документувати всі етапи роботи, що значно знижує ризики провалу проєкту та підвищує ймовірність створення мобільної гри високої якості, що відповідає сучасним вимогам користувачів та стандартам ринку [23].

Використання Unreal Engine 5 у мобільній розробці не лише сприяє створенню технічно та графічно якісного продукту, а й забезпечує комплексний підхід до інтеграції ігрових механік, анімацій та інтерактивних систем у цілісний ігровий процес, що відповідає сучасним стандартам ринку та очікуванням користувачів [24].

1.8. Аналіз і порівняння ігрових рушіїв та обґрунтування вибору Unreal Engine 5

Сучасний ринок ігрових рушіїв пропонує різноманітні інструменти для розробки мобільних ігор, серед яких найбільш популярними є **Unreal Engine 5**, **Unity**, **Godot**, а також менш поширені платформи, такі як **CryEngine** та **Defold**. Кожен із цих рушіїв має свої сильні та слабкі сторони, що впливає на вибір платформи для конкретного проєкту.

Unity є універсальним рушієм з широким співтовариством і великою кількістю готових ресурсів, що забезпечує швидкий старт розробки. Він добре підходить для 2D- та 3D-ігор, має підтримку мобільних платформ і просту інтеграцію з зовнішніми сервісами. Проте складні 3D-сцени з високою деталізацією можуть вимагати додаткової оптимізації та обмежень щодо графічної якості.

Godot є відкритим рушієм з легким доступом до коду та зручними інструментами для 2D-ігор, але його функціонал для складних 3D-проектів поки що обмежений, а кількість готових ресурсів і документації значно менша [25].

CryEngine та інші професійні рушії відомі своєю високою графічною якістю, проте вони часто складні в освоєнні, мають високу вимогу до апаратного забезпечення та обмежену підтримку мобільних платформ.

Вибір Unreal Engine 5 обґрунтований його унікальними можливостями для створення високоякісних мобільних ігор із деталізованою графікою та складними ігровими механіками. Серед ключових переваг UE5:

- потужна система Blueprints для швидкого прототипування ігрових механік;

- підтримка **C++**, що забезпечує гнучкість та масштабованість;

- передові технології Nanite та Lumen, які дозволяють створювати деталізовані сцени та реалістичне освітлення без значного навантаження на мобільні пристрої;

- інтеграція з інструментами командної роботи та системами контролю версій, що забезпечує ефективне управління проєктом;

- широка підтримка платформи та активна спільнота розробників, що дозволяє швидко отримувати допомогу та ресурси [26].

Аналіз існуючих ігрових рушіїв та порівняння їхніх можливостей свідчить, що Unreal Engine 5 є найбільш доцільним вибором для розробки мобільної квестової гри. Це обумовлено поєднанням високих графічних можливостей, що забезпечують деталізовані сцени та реалістичне освітлення, з гнучкістю програмування як через Blueprints, так і через C++, що дозволяє створювати складні ігрові механіки та інтерактивні системи. Крім того, UE5 надає зручні інструменти для інтеграції різноманітного контенту та організації взаємодії між компонентами гри. Підтримка командної роботи, систем контролю версій та ефективних методів управління проєктами дозволяє оптимально розподіляти завдання, відстежувати прогрес розробки та контролювати якість на кожному етапі [27]. Саме ці переваги роблять UE5 найкращим рішенням для створення

мобільних квестових ігор із високим рівнем інтерактивності, привабливою графікою та стабільною продуктивністю на різних платформах [28].

Висновки до Розділу 1

У першому розділі проведено всебічний аналіз сучасних тенденцій розвитку мобільних ігор, ізокрема зосереджено увагу на популярних жанрах, ключових механіках та інтерактивності, що визначають якість користувацького досвіду. Було детально розглянуто особливості створення квестових ігор для мобільних платформ, включно з оптимізацією під обмежені ресурси пристроїв, адаптацією ігрового контенту під різні розміри екранів та забезпеченням зручності сенсорного управління. Встановлено, що інтерактивність та динаміка ігрового процесу є визначальними чинниками залучення користувачів, а балансування складності завдань, система підказок та мотиваційні елементи значно підвищують зацікавленість гравця та покращують ігровий досвід [29].

Досліджено сучасні методології управління IT-проєктами, зокрема Agile, Scrum та Kanban, та їх застосування у процесі мобільної розробки. Було визначено, що ці підходи дозволяють ефективно організувати роботу команди, контролювати виконання завдань на всіх етапах проєкту, мінімізувати ризики та забезпечувати гнучкість у реагуванні на зміни вимог користувачів. Значну увагу приділено інструментам контролю та координації командної взаємодії, що включають системи управління проєктами, засоби контролю версій та комунікаційні платформи, які сприяють прозорому розподілу ролей та відстеженню прогресу розробки.

Окремо розглянуто можливості Unreal Engine 5, що надають розробникам інструменти для інтеграції графіки, анімацій та ігрових механік, застосування Blueprints для швидкого прототипування та C++ API для реалізації складних систем. Технології Nanite та Lumen забезпечують високу візуальну якість мобільних ігор при мінімальному впливі на продуктивність, що є критично важливим для сучасних мобільних платформ [30].

Таким чином, розглянуті теоретичні основи формують міцну базу для практичної частини магістерської роботи, оскільки дозволяють глибше зрозуміти сучасні тренди мобільної ігрової розробки, методи ефективного управління проєктами та оптимального використання професійних інструментів. Це сприяє створенню мобільних ігор високої якості, що відповідають очікуванням користувачів, сучасним стандартам ринку та забезпечують комерційну привабливість продукту [31].

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ МОБІЛЬНОЇ ГРИ В UNREAL ENGINE

2.1. Концептуальна розробка та технічне завдання

Концептуальна розробка мобільної гри є першим і одним з найважливіших етапів проєкту, який визначає загальну ідею, жанр, сюжетну лінію, ігрові механіки та ключові функціональні елементи. На цьому етапі відбувається формування технічного завдання (ТЗ), що включає опис основних компонентів гри, вимоги до графіки, анімацій, інтерактивності, рівнів складності, системи винагород та монетизації.

У ТЗ зазвичай визначаються:

- цільова аудиторія гри;
- жанрова спрямованість (у даному випадку - квестова гра);
- основні ігрові механіки та логіка взаємодії гравця з об’єктами;
- дизайн персонажів та середовища;
- технічні вимоги до мобільних платформ, сумісність та оптимізація;
- інтеграція систем управління, UI/UX та навігації;
- засоби контролю прогресу, збору статистики та монетизації.

Концептуальна розробка передбачає також створення концепт-дизайнів та прототипів, що дозволяють наочно представити майбутню гру та перевірити ідеї перед повномасштабною реалізацією. Цей процес допомагає команді розробників зрозуміти структуру гри, визначити пріоритети та забезпечити чітке бачення кінцевого продукту [32, с. 95].

Технічне завдання відіграє ключову роль у подальшому управлінні проєктом, оскільки воно слугує основою для планування етапів розробки, розподілу ролей у команді та контролю виконання завдань. Чітко сформульоване ТЗ дозволяє мінімізувати ризики, уникати непорозумінь і забезпечує ефективну

координацію між дизайнерами, програмістами та аніматорами на всіх стадіях створення гри.

Розпочинаючи роботу над проєктом «Color Rush», першочерговим завданням стало створення детального плану розробки та аналіз технічних вимог. Цей етап виявився критично важливим для подальшого успішного виконання всіх поставлених завдань. На основі початкового технічного завдання було проведено глибокий аналіз кожного компоненту майбутньої гри, що дозволило виявити потенційні складності та оптимізувати підходи до їх вирішення. Методологія планування базувалася на принципах гнучкої розробки програмного забезпечення, де кожен етап був детально структурований з урахуванням специфічних вимог ігрової індустрії та особливостей мобільних платформ [33, с. 116].

Процес аналізу технічних вимог розпочався з декомпозиції основного завдання на менші, керовані компоненти. Було ідентифіковано п'ять основних підсистем: систему ігрового поля та плиток, логіку управління точками та їх життєвим циклом, механізми взаємодії користувача з ігровими елементами, підсистему інтерфейсу користувача та систему управління станами гри. Кожна з цих підсистем була детально проаналізована з точки зору технічної складності, часових витрат на реалізацію, потенційних ризиків та взаємозалежностей з іншими компонентами системи.

Особливо ретельної уваги потребувала оцінка технічних обмежень цільових мобільних платформ. Аналіз показав, що Android та iOS мають різні підходи до управління пам'яттю, обробки сенсорних подій та оптимізації графічного рендерингу [34, с. 224]. Ці відмінності були враховані при плануванні архітектури системи, що дозволило уникнути критичних проблем на пізніх етапах розробки. Додатково було проведено дослідження продуктивності Unreal Engine 5 на різних конфігураціях мобільних пристроїв, що допомогло визначити оптимальні налаштування для забезпечення стабільної роботи гри.

Особлива увага була приділена архітектурному плануванню системи. Враховуючи специфіку Unreal Engine 5 та його Blueprint-орієнтований підхід,

було вирішено створити модульну архітектуру, де кожен компонент гри функціонує як незалежний модуль із чітко визначеними інтерфейсами взаємодії. Такий підхід забезпечує не лише легкість розробки, але й можливість майбутнього розширення функціональності без кардинальної перебудови існуючого коду [35]. Архітектурне рішення базувалося на патерні Model-View-Controller, адаптованому для специфіки ігрового движка, де Model представляє ігрову логіку та дані, View відповідає за візуалізацію та UI, а Controller управляє взаємодією між користувачем та системою.

Детальне архітектурне планування включало визначення принципів комунікації між модулями через систему подій та делегатів Unreal Engine, що забезпечує слабкий зв'язок між компонентами та високу гнучкість системи. Було запроектовано ієрархію Blueprint класів з базовими абстрактними класами для спільної функціональності та спеціалізованими нащадками для конкретних ігрових елементів. Ця структура дозволяє легко додавати нові типи ігрових об'єктів та модифікувати існуючі без впливу на інші частини системи.

Архітектурний план також включав стратегію управління ресурсами та оптимізації продуктивності. Було запроектовано систему об'єктних пулів для часто створюваних та знищуваних об'єктів, таких як точки на ігровому полі, що значно зменшує навантаження на збирач сміття та покращує загальну продуктивність системи. Додатково було заплановано використання асинхронного завантаження ресурсів для мінімізації затримок під час ігрового процесу [36, с. 52].

Використовуючи платформу Miro, було створено комплексну діаграму архітектури проєкту, яка включала схему взаємодії між основними компонентами системи, детальний flow-chart ігрового процесу від початку до завершення гри, а також план організації файлової структури проєкту. Ця візуальна документація стала незамінним інструментом протягом усього періоду розробки, дозволяючи швидко орієнтуватися у складності системи та приймати обґрунтовані технічні рішення. Діаграми були створені з використанням стандартизованих нотацій UML та специфічних символів для Blueprint системи

Unreal Engine, що забезпечило їх зрозумілість для різних учасників процесу розробки.

Візуальне планування включало створення детальних діаграм послідовності для критичних ігрових сценаріїв, таких як процес появи точки, обробка натискання користувача, оновлення рахунку та перехід між станами гри. Кожен сценарій був проаналізований з точки зору потенційних точок відмови та способів їх обробки [37]. Було створено діаграми станів для основних ігрових об'єктів, що дозволило чітко визначити всі можливі стани та переходи між ними.

Особлива увага була приділена плануванню користувацького досвіду через створення детальних wireframe-ів всіх екранів гри та інтерактивних прототипів ключових взаємодій. Ці прототипи дозволили заздалегідь виявити потенційні проблеми з юзабіліті та оптимізувати інтерфейс до початку фактичної розробки. Wireframe-и були створені з урахуванням різних розмірів екранів та орієнтацій пристроїв, що забезпечило універсальність дизайну [38, с. 89].

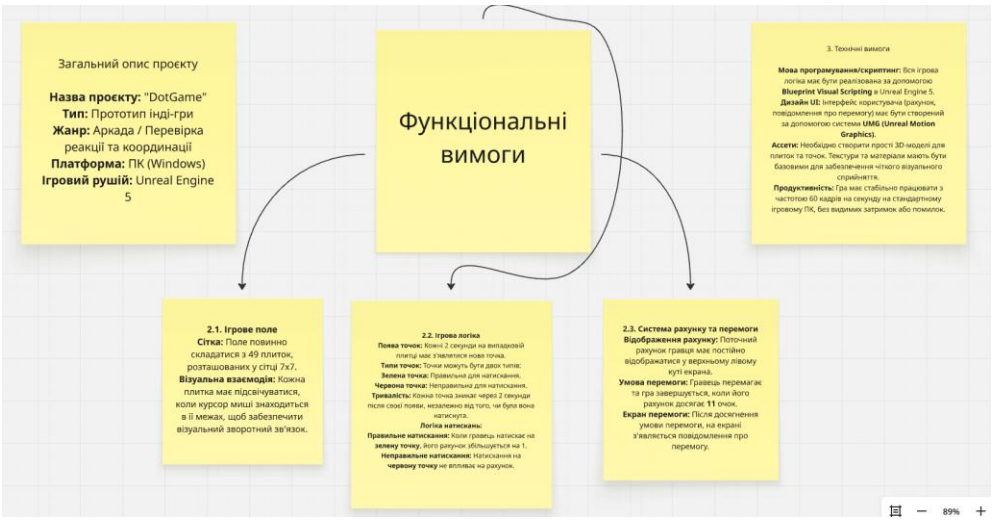

Рис. 1. Створення структури проєкту в дошках Miro

Джерело:[створено автором]

Планування також включало детальну оцінку ризиків проєкту та розробку стратегій їх мітигації. Були ідентифіковані технічні ризики, пов'язані з продуктивністю на мобільних пристроях, потенційними проблемами сумісності

між різними версіями Android та iOS, а також ризики, пов'язані з інтеграцією різних систем Unreal Engine. For кожного ризику було розроблено план попередження та план реагування у випадку його реалізації.

Окремою частиною планування стала розробка стратегії тестування, яка включала план модульного тестування кожного компоненту, інтеграційного тестування взаємодії між системами, тестування продуктивності на різних конфігураціях пристроїв та user acceptance testing для оцінки користувацького досвіду. Було визначено критерії готовності для кожного етапу розробки та метрики якості, які повинні бути досягнуті.

Підготовка робочого середовища включала не лише встановлення необхідного програмного забезпечення, але й його оптимальну конфігурацію для максимальної продуктивності розробки. Було встановлено Unreal Engine 5.3 з усіма необхідними компонентами для мобільної розробки, включаючи Android SDK, NDK, Java Development Kit у відповідних версіях, а також Xcode для iOS розробки. Особлива увага була приділена налаштуванню емуляторів та підключенню фізичних пристроїв для тестування, що дозволило проводити регулярне тестування на реальному обладнанні протягом усього процесу розробки [39, с. 367].

Конфігурація Unreal Engine включала налаштування проєктних шаблонів, оптимізацію компіляційних налаштувань для швидкого ітераційного розвитку, а також встановлення додаткових плагінів для розширення функціональності середовища розробки. Було налаштовано систему автоматичного збереження та резервного копіювання проєкту, що мінімізувало ризик втрати даних під час розробки.

Система контролю версій була налаштована з використанням Git з урахуванням специфіки роботи з бінарними файлами Unreal Engine. Було створено детальну структуру гілок для різних етапів розробки, включаючи основну гілку для стабільного коду, гілки розробки для нових функцій та гілки для експериментальних можливостей. Налаштовано автоматичні правила для

merge request-ів та систему continuous integration для автоматичного тестування змін.

Організація файлової структури проєкту була виконана відповідно до найкращих практик Unreal Engine розробки, з чіткими правилами іменування файлів та папок, логічним групуванням ресурсів за типами та функціональністю. Було створено детальні стандарти кодування для Blueprint-ів, включаючи правила іменування змінних, функцій та класів, а також стандарти документування коду для забезпечення його подальшої підтримки.

Додатково було налаштовано інтегроване середовище для створення та редагування ресурсів, включаючи Blender для 3D моделювання, GIMP для роботи з текстурами та зображеннями, а також Audacity для обробки звукових ефектів. Всі ці інструменти були інтегровані в єдиний workflow з автоматизованими скриптами для експорту ресурсів у форматах, сумісних з Unreal Engine.

Планування включало також розробку детального графіку проєкту з визначенням ключових milestone-ів та deliverable-ів. Було створено Gantt діаграму з урахуванням взаємозалежностей між завданнями та критичним шляхом проєкту. Кожне завдання було оцінено за складністю та часом виконання, з урахуванням буферного часу для непередбачених обставин.

Особлива увага була приділена плануванню документації проєкту, включаючи технічну документацію для розробників, користувацьку документацію для гравців та адміністративну документацію для управління проєктом. Було визначено стандарти оформлення документації та інструменти для її створення та підтримки в актуальному стані.

Завершальним етапом планування стала підготовка критеріїв успіху проєкту та метрик для оцінки його ефективності. Було визначено технічні показники, такі як продуктивність гри на різних пристроях, час завантаження, споживання пам'яті, а також користувацькі метрики, включаючи час навчання нових гравців, рівень утримання користувачів та загальну задоволеність ігровим

досвідом. Ці метрики стали основою для оцінки успішності проєкту та визначення напрямків для майбутніх покращень.

2.2. Створення 3D-контенту та графічних ресурсів (Blender, Photoshop, Illustrator)

Для сучасної мобільної гри важливим аспектом є не лише якісна ігрова логіка, але й візуальна складова, яка забезпечує занурення гравця у віртуальний світ. Створення 3D-контенту та графічних ресурсів є ключовим етапом розробки, адже саме вони формують зовнішній вигляд персонажів, середовища, інтерфейсу та інтерактивних об’єктів.

На етапі розробки необхідно враховувати особливості мобільних платформ: обмежені ресурси, різні розміри та роздільні здатності екранів, а також ефективне використання полігональної сітки і текстур для забезпечення стабільної продуктивності. Крім того, важливо підтримувати єдиний стиль гри та забезпечувати сумісність ресурсів із рушієм Unreal Engine 5.

Використання спеціалізованих інструментів для моделювання, текстурування та створення векторної графіки дозволяє реалізувати поставлені завдання максимально ефективно. Далі розглядається процес створення 3D-моделей, текстур та графічних елементів із застосуванням Blender, Adobe Photoshop та Illustrator, а також їх інтеграція у проєкт для забезпечення цілісності та високої якості візуального оформлення мобільної гри.

У Blender було створено основні 3D-моделі персонажів, об’єктів середовища та інтерактивних елементів гри. Особлива увага приділялась оптимізації полігональної сітки для забезпечення стабільної роботи на мобільних пристроях та правильному налаштуванню UV-розгорток для текстурування. Також були використані базові анімації персонажів та об’єктів із застосуванням ріггінгу та скелетних систем, що дозволило забезпечити плавний рух і інтерактивність у грі.

Adobe Photoshop застосовувався для створення текстур, деталей графічного інтерфейсу, ефектів освітлення та створення sprite-ресурсів. Тут виконувалось коригування кольорової палітри, робота з альфа-каналами для прозорих елементів та підготовка ресурсів до імпорту в Unreal Engine 5.

Adobe Illustrator використовувався для створення векторних елементів інтерфейсу, іконок, логотипів та інших графічних елементів, які потребували масштабованості без втрати якості. Це дозволило підготувати ресурси для адаптації під різні розміри екранів мобільних пристроїв.

Комплексна робота з цими інструментами забезпечила створення узгодженого візуального стилю гри, а також інтеграцію графічних ресурсів у рушій UE5 для подальшої реалізації ігрових механік та інтерфейсу користувача.

2.3. Реалізація ігрових механік та інтерактивних систем

Після завершення підготовки технічного завдання та створення графічного контенту наступним етапом є безпосередня реалізація ігрових механік та інтерактивних систем. Цей етап передбачає програмну реалізацію логіки гри, інтеграцію 3D-моделей та графічних ресурсів, а також забезпечення взаємодії користувача з ігровим світом.

У контексті мобільної розробки особлива увага приділяється оптимізації ресурсів, забезпеченню стабільної продуктивності та адаптації керування під сенсорні екрани. Для реалізації механік використовується рушій Unreal Engine 5, який надає широкі можливості як через візуальне програмування Blueprints, так і через гнучкий C++ API. Це дозволяє швидко прототипувати ігрові системи, створювати взаємодії між об’єктами та налаштовувати поведінку персонажів і елементів середовища.

Особливу роль відіграє інтерактивність, яка забезпечує гравцеві можливість безпосередньо впливати на події у грі та отримувати своєчасний зворотний зв’язок. До основних компонентів реалізації входять: система управління персонажем, логіка появи та обробки об’єктів, динамічне реагування

на дії користувача, управління станами гри та реалізація діалогових і квестових систем.

Реалізація ігрових механік та інтерактивних систем відіграє ключову роль у створенні мобільної гри, оскільки саме від ефективності цих компонентів залежить повноцінний ігровий досвід користувача. Впровадження продуманих механік дозволяє забезпечити логічну послідовність дій, правильну реакцію об’єктів на взаємодію гравця та підтримку інтересу протягом усього проходження гри.

Інтерактивні системи дозволяють гравцеві активно впливати на події у грі, отримувати зворотний зв’язок та відчувати контроль над ігровим світом. Це включає управління персонажем, взаємодію з об’єктами, реагування на події та зміни станів гри, реалізацію квестових завдань та сценаріїв. Кожен елемент взаємодії формує відчуття занурення та робить геймплей більш насиченим і динамічним.

Використання Unreal Engine 5 забезпечує можливість поєднувати візуальне програмування Blueprints та програмування на C++ для створення складних механік і адаптації їх під мобільні платформи. Це дозволяє ефективно інтегрувати графічні ресурси, анімації та звуковий супровід у єдину систему, оптимізувати продуктивність гри та забезпечити стабільну роботу на різних пристроях.

Завдяки ретельній реалізації механік та інтерактивних систем створюється цілісний ігровий процес, який відповідає сучасним стандартам якості, підвищує залученість користувачів та забезпечує комерційну конкурентоспроможність мобільного продукту.

Першим кроком стало створення ігрового поля. Для цього було розроблено основний Blueprint актора, який динамічно генерує сітку 7x7 плиток. Кожна плитка є окремим Blueprint-актором, що має свій матеріал та логіку для підсвічування при наведенні курсору. Цей підхід дозволив легко масштабувати сітку та керувати поведінкою кожної плитки окремо.

Фундаментальним елементом архітектури гри став центральний контролер, реалізований у вигляді GameMode Blueprint. Цей компонент виступає в ролі координатора всіх ігрових процесів, забезпечуючи ініціалізацію ігрового поля при запуску гри, управління станами гри від головного меню до екрану перемоги, координацію комунікації між різними системами, а також обробку збереження та завантаження користувацьких налаштувань та прогресу.

Розробка системи плиток потребувала особливої уваги до деталей, оскільки саме ці елементи забезпечують основну взаємодію з гравцем. Кожна плитка представлена окремим Blueprint класом «PuzzleBlock», що містить базову геометрію у вигляді статичного меша куба з ретельно підібраними пропорціями для оптимального візуального сприйняття. Система матеріалів була розроблена з підтримкою динамічної зміни кольорів та параметрів, що дозволяє створювати плавні візуальні ефекти підсвічування та реакції на дії користувача.

Особливу увагу було приділено системі колізії, яка забезпечує точне розпізнавання натискань миші та їх правильну обробку. Компонент колізії був налаштований таким чином, щоб мінімізувати помилкові спрацьовування та забезпечити максимально точну реакцію на дії гравця. Логіка візуального зворотного зв'язку включає не лише зміну кольору при наведенні курсору, але й тонкі анімаційні ефекти, що роблять взаємодію більш інтуїтивною та приємною.

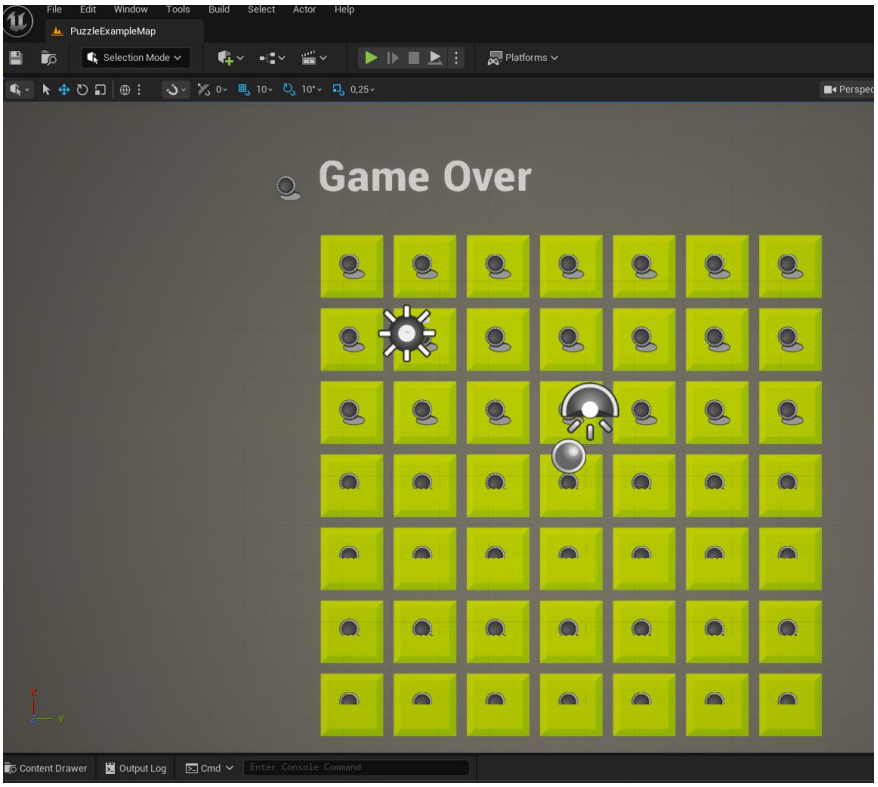

Рис. 2. Створене ігрове поле

Джерело: [створено автором]

Алгоритм динамічного створення ігрового поля представляє собою складну систему, яка генерує сітку розміром сім на сім плиток із математичною точністю позиціонування. Використовуючи вкладені цикли, система створює масив із сорока дев'яти плиток, кожна з яких розміщується у просторі з урахуванням заданих відступів та масштабування. Критично важливим аспектом цієї системи є збереження посилань на всі створені плитки в спеціальному масиві, що забезпечує швидкий доступ до будь-якої плитки для подальших маніпуляцій.

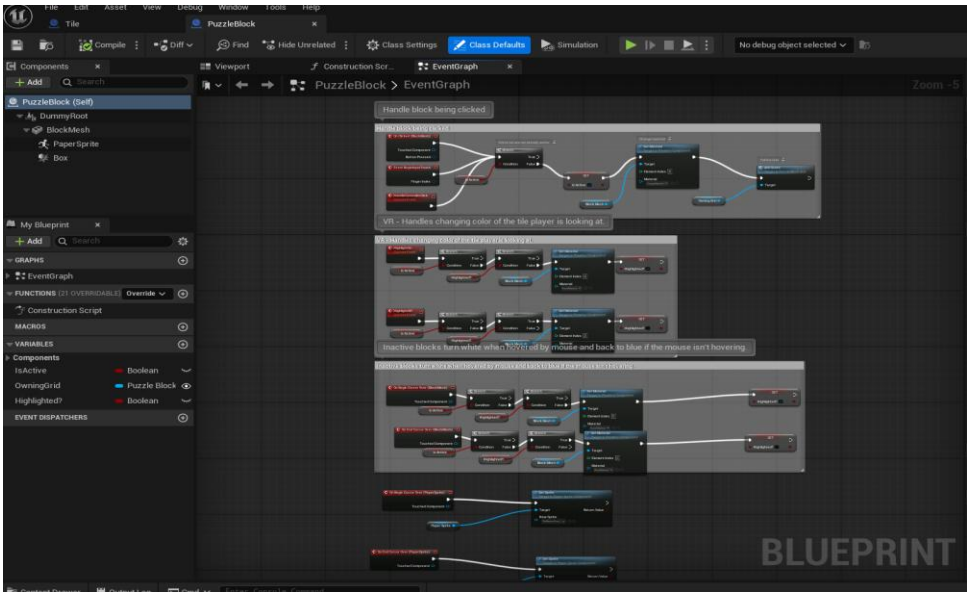

Рис. 3. Блюпринт блоку пазлу

Джерело: [створено автором]

Представлено Blueprint для створення інтерактивної плитки (PuzzleBlock), яка реагує на кліки миші та наведення курсора. Коментарі про VR свідчать про те, що гра має розширені функціональні можливості.

Для оптимізації продуктивності було впроваджено техніку об'єднання статичних мешів, яка значно зменшує навантаження на графічну підсистему без втрати візуальної якості. Ця оптимізація особливо важлива для мобільних пристроїв, де ресурси обмежені, а продуктивність є критичним фактором користувацького досвіду.

Серцем ігрової механіки стала складна система управління точками, яка забезпечує випадкову появу та зникнення ігрових елементів відповідно до заданих правил. Таймерна підсистема, побудована на основі функції Set Timer by Function Name, створює регулярні події з інтервалом у дві секунди, забезпечуючи постійний ритм гри та підтримуючи напругу протягом всього ігрового процесу.

Генерація випадкових позицій реалізована через досконалий алгоритм, який використовує функцію Random Integer in Range для вибору одної з сорока дев'яти доступних плиток. Система перевіряє доступність обраної плитки, щоб уникнути накладання декількох точок на одному місці, що могло б призвести до

плутанини в ігровому процесі. Додатково, алгоритм враховує розподіл типів точок, забезпечуючи приблизно сімдесят відсотків зелених (правильних) точок та тридцять відсотків червоних (неправильних), що створює збалансований рівень складності.

Lifecycle точок включає автоматичне зникнення через півтори секунди після появи, створюючи необхідний часовий тиск на гравця. Цей механізм не лише додає динамізму грі, але й вимагає від гравця швидкої реакції та уважності. Система відстежує всі активні точки в спеціальному масиві, що дозволяє ефективно управляти їх життєвим циклом та запобігає витокам пам'яті.

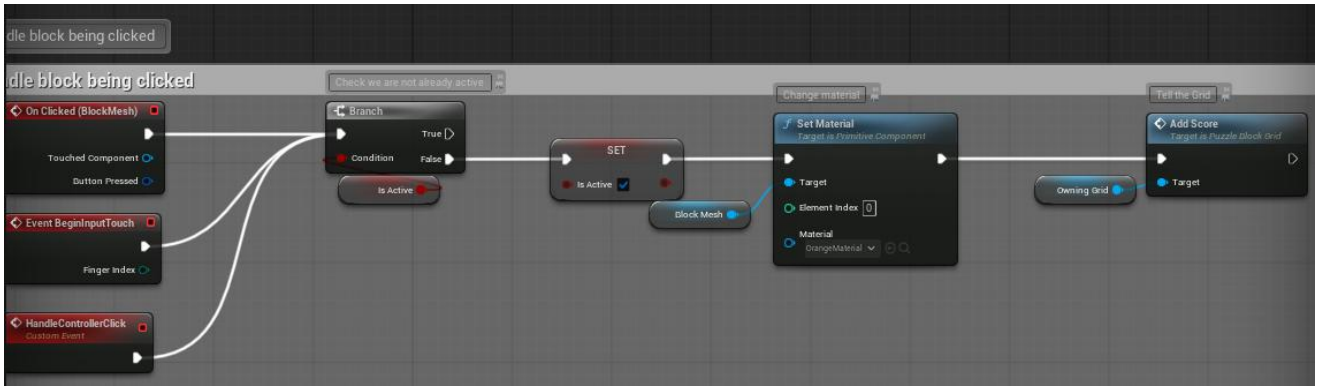

Рис. 4. Обробник подій після натиснення

Джерело: [створено автором]

Логіка взаємодії з точками була розширена для забезпечення більш інтуїтивного та прощаючого ігрового досвіду. Система точності включає tolerance для натискань, що знаходяться поблизу точки, враховуючи той факт, що точні натискання можуть бути складними на сенсорних екранах мобільних пристроїв. Комбо-система відстежує послідовні правильні натискання та надає бонусні очки за демонстрацію майстерності, створюючи додаткову мотивацію для точної гри.

Штрафна система була розроблена таким чином, щоб збалансувати виклик та фрустрацію. Неправильні натискання призводять до зменшення рахунку або скидання поточного комбо, але не повністю знищують прогрес гравця, що дозволяє зберегти мотивацію до продовження гри навіть після помилок.

Система підрахунку очок була спроектована як багаторівнева структура, що враховує різні аспекти ігрового процесу та створює збалансований механізм мотивації гравця. Архітектура системи базується на трьох основних компонентах: базовий рахунок, система комбо-множників та штрафні механізми.

Базова система нараховує одне очко за кожне правильне натискання на зелену точку, забезпечуючи простий та зрозумілий спосіб відстеження прогресу. Цей підхід гарантує лінійний прогрес для початкових гравців і служить фундаментом для більш складних механік нарахування очок.

Базове нарахування очок реалізоване за формулою:

Базові очки = Σ(правильні натискання × 1)

Кожне успішне натискання на зелену точку додає рівно одне очко до загального рахунку, незалежно від часу реакції чи інших факторів. Така система забезпечує передбачуваність та справедливість для всіх категорій гравців.

Проте справжня глибина системи розкривається через механізм комбо, який активується при трьох або більше послідовних правильних натисканнях. Ця система створює експоненційну криву нарахування очок для досвідчених гравців.

Система комбо активується за наступним алгоритмом:

- При досягненні 3 послідовних правильних натискань активується базовий комбо-множник 1.5x

- Кожні наступні 2 правильні натискання збільшують множник на 0.25x

- Максимальний множник обмежений значенням 3.0x для збереження балансу гри

- Комбо скидається до нуля при будь-якому неправильному натисканні

Математична формула комбо-системи:

Комбо-множник = min(1.5 + (floor((streak - 3) / 2) × 0.25), 3.0)

Очки з комбо = базові очки × комбо-множник

де streak - кількість послідовних правильних натискань.

Коли гравець демонструє стабільну точність, система комбо застосовує множник у півтора рази до базових очок, значно прискорюючи прогрес до цільового рахунку в одинадцять очок. Цей механізм створює цікаву динаміку, де досвідчені гравці можуть завершити гру швидше, але водночас збільшується ризик втратити весь накопичений комбо через одну помилку.

Комбо-система створює психологічний ефект "зони", коли гравець входить у стан підвищеної концентрації. Візуальні та звукові ефекти підсилюють це відчуття, створюючи відчуття досягнення та мотивуючи до продовження серії успішних натискань.

Штрафна система працює як противага системі винагород, зменшуючи рахунок на одне очко за кожне неправильне натискання або повністю скидаючи поточний комбо-множник. Такий підхід створює напругу та вимагає від гравця постійної концентрації, перетворюючи просту гру на справжній тест реакції та уваги.

Штраф за промах: -1 очко за натискання на червону точку

Штраф за пропуск: -1 очко за ігнорування зеленої точки до її зникнення

Штраф за передчасне натискання: -1 очко за натискання до появи точки

Скидання комбо: повне обнулення поточного комбо-множника при будь-якій помилці

Для запобігання фрустрації гравців впроваджено захисний механізм:

Мінімальний рахунок = max(поточний рахунок - штраф, 0)

Рахунок ніколи не може стати негативним, що забезпечує позитивний ігровий досвід навіть для початківців.

Локальне збереження прогресу включає не лише найкращий результат гравця, але й детальну статистику ігрових сесій, включаючи середній час реакції, кількість правильних та неправильних натискань, а також тривалість найдовшого комбо.

Впроваджено систему досягнень для підтримки довготривалої мотивації:

Досягнення точності:

"Снайпер" - 95% точність у грі

"Майстер" - 98% точність у грі

"Легенда" - 100% точність у грі з мінімум 20 натисканнями

Досягнення комбо:

"Серія" - комбо з 5 натискань

"Ланцюг" - комбо з 10 натискань

"Нескінченність" - комбо з 20 натискань

Досягнення швидкості:

"Спринтер" - середній час реакції менше 400мс

"Блискавка" - середній час реакції менше 300мс

"Світло" - середній час реакції менше 250мс

2.4. Розробка користувацького інтерфейсу та систем взаємодії

Розробка користувацького інтерфейсу та систем взаємодії є важливою складовою мобільної гри, оскільки саме через інтерфейс гравець взаємодіє з ігровим світом і отримує зворотний зв’язок. Вдалий дизайн UI забезпечує інтуїтивно зрозумілу навігацію, легкий доступ до функцій гри, а також комфортне виконання завдань у межах обмеженого екранного простору мобільного пристрою.

Під час створення інтерфейсу враховуються такі аспекти, як розмір елементів керування, розташування кнопок, кольорові схеми та адаптивність до різних розмірів екранів і роздільної здатності. Особлива увага приділяється сенсорній взаємодії, щоб забезпечити точність та швидкість відгуку на дотики користувача.

Системи взаємодії включають елементи керування персонажем, об’єктами та меню гри, а також діалогові вікна, підказки та сповіщення про зміни стану гри.

Використання Blueprint-систем Unreal Engine 5 дозволяє створювати інтерактивні прототипи UI, що дає змогу тестувати функціональність і зручність інтерфейсу ще на етапі розробки.

Також враховується інтеграція інтерфейсу з основними ігровими механіками та логікою взаємодії, що забезпечує узгодженість усіх елементів гри та підвищує рівень занурення гравця у процес. Ретельне планування та реалізація UI і систем взаємодії створює основу для комфортного та захоплюючого користувацького досвіду на мобільних платформах.

Інтерфейс користувача є невід'ємною складовою мобільної гри, оскільки від його якості та зручності безпосередньо залежить комфорт та ефективність взаємодії гравця з ігровим світом. Інтеграція UI з основними ігровими механіками та логікою взаємодії дозволяє забезпечити узгодженість усіх елементів гри, що сприяє плавності ігрового процесу та створює відчуття цілісності середовища.

Ретельне планування та реалізація систем взаємодії передбачає не лише естетичну привабливість інтерфейсу, а й інтуїтивно зрозумілу навігацію, швидку доступність ключових функцій і адаптацію під різні розміри екранів та роздільні здатності мобільних пристроїв. Важливим аспектом є передбачення реакцій інтерфейсу на дії користувача, включаючи анімації, підказки та візуальний зворотний зв'язок, що підвищує ефективність взаємодії та покращує користувацький досвід.

Крім того, інтеграція інтерфейсу з ігровими механіками дозволяє створювати більш глибокі та інтерактивні сценарії, де дії гравця безпосередньо впливають на стан ігрового світу та відображаються через UI елементи. Це сприяє залученню користувача, підвищує мотивацію проходити рівні та досліджувати ігровий контент.

Таким чином, продумана розробка користувацького інтерфейсу та систем взаємодії є критичною для забезпечення комфортного, інтуїтивного та захоплюючого користувацького досвіду на мобільних платформах і створює основу для успішної взаємодії гравця з усіма компонентами гри.

Для відображення ігрової інформації було розроблено UMG Widget Blueprint. Основні елементи UI складаються з:

- Текстового поле для відображення поточного рахунку.

- Екран перемоги, який з'являється після досягнення 11 очок.

- UI-елементи були прив'язані до ігрової логіки, щоб забезпечити їх динамічне оновлення.

Розробка інтерфейсу користувача потребувала особливо ретельного підходу, оскільки UI в іграх такого типу повинен бути інформативним, але не відволікаючим від основного ігрового процесу. Архітектура UI системи була побудована на принципах модульності, використовуючи потужні можливості UMG (Unreal Motion Graphics) для створення адаптивного та візуально привабливого інтерфейсу.

Основний ігровий HUD представляє собою мінімалістичний дизайн, де кожен елемент має чітко визначену функцію та оптимальне розміщення. Віджет відображення поточного рахунку розміщений у верхньому лівому куті екрана, де він залишається помітним, але не заважає ігровому процесу. Типографіка була обрана таким чином, щоб забезпечити максимальну читабельність навіть при швидких змінах числових значень.

Індикатор комбо з'являється лише при досягненні двох або більше правильних натискань поспіль, створюючи динамічний інтерфейс, який адаптується до поточного стану гри. Цей елемент не лише інформує гравця про поточний статус комбо, але й використовує візуальні ефекти для підкреслення досягнення, що посилює відчуття прогресу та майстерності.

Таймер до наступної точки реалізований у вигляді тонкого прогрес-бару, який дозволяє гравцеві передбачити момент появи наступної точки без відволікання уваги від ігрового поля. Кнопка паузи та налаштувань розміщена у дискретному місці, доступному для швидкого доступу, але не заважаючому основному ігровому процесу.

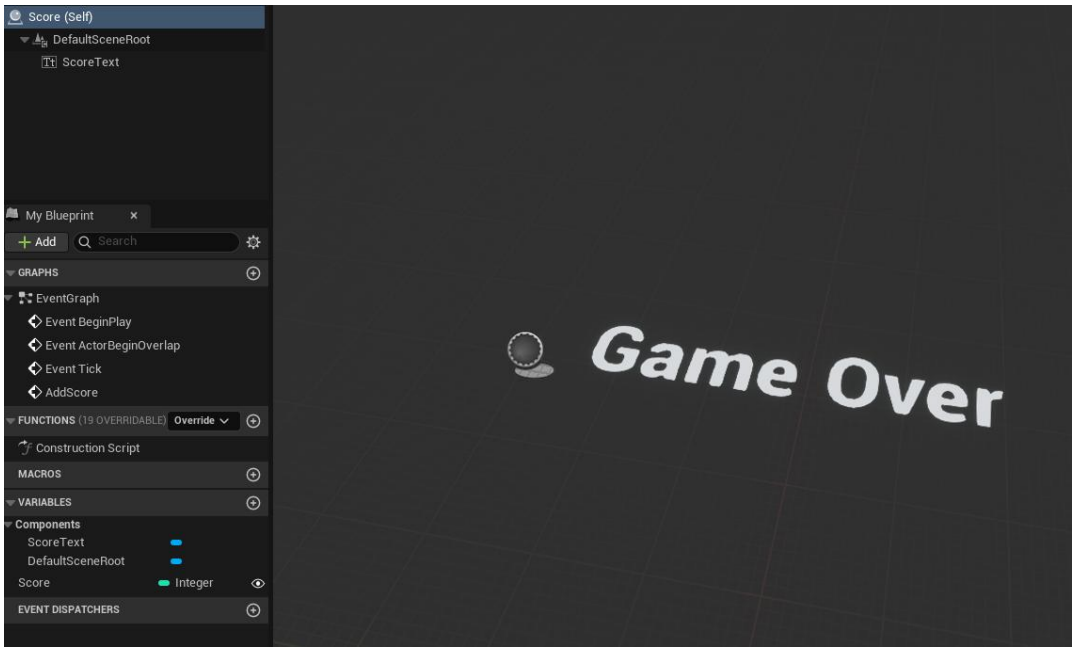

Рис. 5. Текстове поле Поразки

Джерело: [створено автором]

Система екранів була розроблена з урахуванням сучасних принципів UX дизайну. Головне меню представляє чистий та інтуїтивний інтерфейс з кнопками «Грати», «Налаштування» та «Вихід», кожна з яких має чітко визначену візуальну ієрархію. Екран гри підтримує концепцію мінімалістичного HUD, дозволяючи гравцеві повністю зосередитися на ігровому процесі.

Екрани перемоги та поразки були спроектовані не лише як інформаційні панелі, але й як мотиваційні інструменти. Екран перемоги включає святкову анімацію, відображення фінального рахунку з підкресленням досягнень, повідомлення про новий рекорд у разі його встановлення, а також зручні кнопки для повторної гри або повернення до головного меню. Аналогічний підхід застосовано до екрану поразки, але з фокусом на заохочення до нової спроби замість святкування.

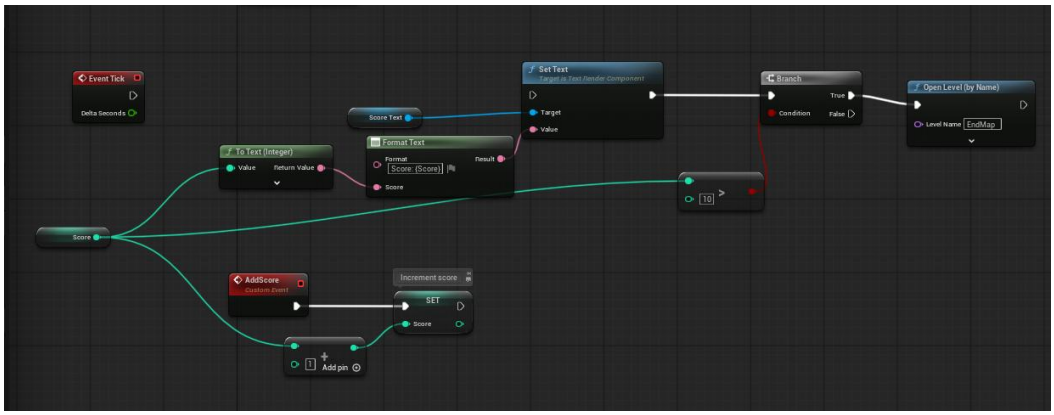

Рис. 6. Блюпринт додавання очків

Джерело: [створено автором]

Звуковий дизайн гри був розроблений як інтегральна частина ігрового досвіду, що підсилює емоційну складову взаємодії з грою. Звук успішного натискання використовує позитивний тон, що створює відчуття задоволення від правильних дій та підкріплює позитивні асоціації з успіхом. Навпаки, звук помилки має стриманий негативний тон, який сигналізує про неправильну дію, не створюючи при цьому надмірної фрустрації.

Фонова музика була розроблена як опціональний елемент з можливістю повного відключення, враховуючи різні переваги гравців щодо звукового супроводу. Мелодія має ритмічний характер, що підкреслює динамічність ігрового процесу, але залишається достатньо ненав'язливою, щоб не відволікати увагу від основних ігрових механік.

Звукові ефекти перемоги та поразки були створені як кульмінаційні моменти звукового досвіду, використовуючи більш виразні та емоційно забарвлені тони, що підкреслюють важливість цих моментів у ігровому процесі.

Візуальні ефекти включають складну систему частинок для успішних натискань, де зелені іскри створюють відчуття енергії та позитиву, підкреслюючи правильність дії гравця. Ефект screen shake при помилках використовується помірно, створюючи тактильне відчуття наслідків неправильного вибору без надмірного дискомфорту.

Система smooth transitions між різними екранами забезпечує плавність користувацького досвіду, використовуючи сучасні техніки анімації для створення професійного враження від гри. Анімація появи та зникнення точок була розроблена таким чином, щоб бути помітною та інформативною, але не надто відволікаючою від основного ігрового процесу.

2.5. Тестування, оптимізація та підвищення продуктивності гри

Завершальний етап включав тестування гри для виявлення та виправлення помилок. Було проведено перевірку на коректність роботи таймерів, логіки підрахунку очок, відображення UI.

Процес тестування був організований як систематичний підхід до забезпечення якості на всіх рівнях системи. Модульне тестування включало детальну перевірку генерації сітки на різних розмірах для забезпечення стабільності алгоритму за межами стандартних параметрів. Система колізії була протестована на точність розпізнавання натискань у різних умовах, включаючи граничні випадки та екстремальні сценарії використання.

Особлива увага була приділена валідації логіки підрахунку очок через автоматизовані тести, які перевіряли правильність нарахування базових очок, роботу системи комбо, а також коректність застосування штрафних санкцій. Робота таймерних систем була перевірена при різних рівнях навантаження, включаючи сценарії з множинними одночасними таймерами та потенційними конфліктами в розкладі подій.

Інтеграційне тестування фокусувалося на взаємодії між різними підсистемами гри. Перевірка взаємодії UI з ігровою логікою включала тестування синхронізації відображення рахунку з реальними змінами в ігровому стані, а також коректності відображення різних повідомлень та станів інтерфейсу. Тестування переходів між станами гри виявило та допомогло виправити декілька потенційних race conditions, які могли призвести до непередбачуваної поведінки системи.

Валідація збереження та завантаження даних включала перевірку цілісності збереженої інформації, коректності відновлення стану після перезапуску програми, а також стійкості системи до пошкоджених файлів збереження. Стрес-тестування тривалих ігрових сесій допомогло виявити потенційні проблеми з управлінням пам'яттю та накопиченням ресурсів протягом часу.

Оптимізація продуктивності стала критично важливим етапом, особливо враховуючи мобільну орієнтацію гри. Профілювання Blueprint-ів через Unreal Insights дозволило ідентифікувати найбільш ресурсомісткі операції та оптимізувати їх виконання. Оптимізація кількості активних таймерів призвела до значного покращення загальної продуктивності системи, особливо на пристроях з обмеженими ресурсами.

Зменшення кількості викликів Tick функцій було досягнуто через перехід до event-driven архітектури, де можливо, що значно зменшило навантаження на центральний процесор. Оптимізація рендерингу через впровадження LOD (Level of Detail) систем забезпечила стабільну частоту кадрів навіть на менш потужних пристроях.

Протягом процесу тестування було виявлено та успішно виправлено декілька критичних проблем. Проблема дублювання точок на одній плитці була вирішена через впровадження системи резервування плиток перед створенням точок. Racing conditions в системі таймерів були усунуті через застосування семафорів та покращену синхронізацію подій.

Оптимізація memory leaks від недеструйованих акторів потребувала впровадження строгих правил управління життєвим циклом об'єктів та автоматичної системи прибирання ресурсів. Покращення responsive design для різних роздільностей екрана забезпечило універсальність гри на широкому спектрі мобільних пристроїв.

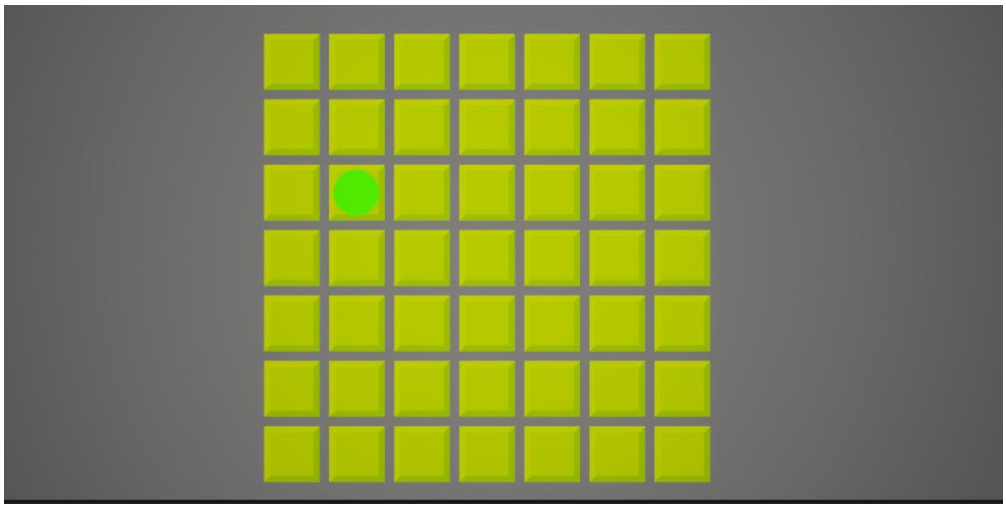

Рис. 7. Результат ігрового поля

Джерело: [створено автором]

2.6. Фінальна інтеграція та підготовка до релізу

Завершальний етап розробки включав комплексну підготовку проєкту до публікації, що потребувало ретельної уваги до деталей на всіх рівнях системи. Налаштування збірки для різних платформ включало конфігурацію специфічних параметрів для Android та iOS, врахування особливостей кожної операційної системи та їхніх технічних обмежень.

Оптимізація розміру збірки була досягнута через використання Asset Cooking технології Unreal Engine, яка дозволяє включати до фінального пакету лише ті ресурси, які реально використовуються в грі. Цей процес значно зменшив розмір фінального файлу, що критично важливо для мобільних додатків, де користувачі чутливі до розміру завантажуваних файлів.

Налаштування compression settings для текстур та аудіо файлів було виконано з урахуванням балансу між якістю контенту та розміром файлів. Використовуючи сучасні алгоритми стиснення, вдалося досягти високої якості візуальних та аудіо елементів при мінімальному впливі на загальний розмір програми.

Створення інсталяційних пакетів включало підготовку всіх необхідних метаданих, іконок програми в різних розмірах, скріншотів для магазинів

програм, а також опису програми різними мовами. Особлива увага була приділена дотриманню всіх вимог Google Play Store та Apple App Store щодо контенту та технічних специфікацій.

Документація стала важливою частиною завершального етапу, забезпечуючи можливість подальшої підтримки та розвитку проєкту. User manual для гравців був створений у вигляді інтерактивного посібника, який інтегрований безпосередньо в гру та дозволяє новим користувачам швидко освоїти основні механіки та правила.

Технічна документація для подальшої підтримки включає детальний опис архітектури системи, пояснення ключових алгоритмів, інструкції з налагодження та виправлення типових проблем, а також рекомендації щодо оптимізації продуктивності. Опис архітектури для майбутніх розширень передбачає можливі напрямки розвитку гри, включаючи додавання нових ігрових режимів, систем досягнень та мультиплеєрної функціональності.

Список відомих обмежень та планів розвитку документує поточні технічні обмеження системи, які не є критичними для основної функціональності, але можуть бути покращені в майбутніх версіях. Плани розвитку включають короткострокові та довгострокові цілі для еволюції проєкту, враховуючи потенційні потреби користувачів та технологічні можливості.

Результатом цього комплексного підходу до розробки стала повністю функціональна та оптимізована гра «Color Rush», яка не лише відповідає всім початковим технічним вимогам, але й демонструє високий рівень професіоналізму в реалізації. Гра включає додаткові функції, що значно покращують користувацький досвід, та демонструє глибоке розуміння принципів сучасного game development на платформі Unreal Engine 5, а також ефективного управління проєктами в IT-сфері.

2.7. Гейміфікація, аналітика та оцінка користувацького досвіду.

У проєкті «Color Rush» була проведена всебічна робота над підвищенням залученості гравців та оптимізацією ігрового процесу. Для цього реалізовано багаторівневу систему гейміфікації, яка включала досягнення та нагороди за виконання завдань, рейтингові таблиці для стимулювання конкуренції між гравцями, прогресивну систему рівнів та динамічні завдання, що змінюються відповідно до дій користувача. Це дозволило мотивувати гравців повертатися до гри та підтримувати інтерес протягом тривалого часу.

Паралельно проводилася детальна аналітика користувацької взаємодії з грою за допомогою інструментів Firebase та Unreal Insights. Збиралися дані про тривалість ігрових сесій, популярність конкретних механік, частоту повторного проходження рівнів та поведінку користувачів у різних ігрових ситуаціях. На основі цих даних було проведено коригування геймплейних елементів: підвищено або знижено складність рівнів, вдосконалено систему винагород, збалансовано темп прогресії та адаптовано складність завдань для різних категорій гравців.

Було здійснено всебічну оцінку користувацького досвіду, яка охоплювала аналіз зручності інтерфейсу, логіки взаємодії з об’єктами та реакції гравців на різні ігрові сценарії. Особлива увага приділялася комфортності проходження як основних, так і додаткових квестів, щоб забезпечити рівномірний рівень залучення та задоволення гравця на всіх етапах гри. Тестування дозволило виявити вузькі місця у дизайні та ігровій механіці, усунути можливі помилки взаємодії та оптимізувати навігацію по меню, переходи між рівнями та інші геймплейні елементи. В процесі аналізу враховувалися різні сценарії використання гри на мобільних пристроях різних розмірів та роздільної здатності, що забезпечило адаптивність інтерфейсу і зручність для широкого кола користувачів. Ця робота дозволила підвищити загальну якість користувацького досвіду, створити більш інтуїтивну ігрову взаємодію та посилити емоційне занурення гравців у процес.

Інтеграція аналітичних інструментів дозволила здійснювати безперервний моніторинг ефективності реалізованих функцій та впливу змін на ігровий процес, що забезпечило основу для подальшого вдосконалення та розвитку гри. Поєднання системи гейміфікації, аналітики та ретельної оцінки користувацького досвіду сприяло створенню мобільного продукту з високим рівнем занурення, який відповідає сучасним вимогам гравців, підтримує їхню зацікавленість та забезпечує тривалий інтерес до гри на різних платформах.

Висновки до 2 го розділу

У другому розділі було детально розглянуто практичну реалізацію проєкту «Color Rush», починаючи від концептуальної розробки та складання технічного завдання до створення 3D-контенту, графічних ресурсів і користувацького інтерфейсу. Було описано процес реалізації ігрових механік та інтерактивних систем, що дозволило забезпечити логічну взаємодію всіх компонентів гри та оптимізувати ігровий процес.

Здійснено комплексну роботу з тестування, оптимізації та підвищення продуктивності, що дозволило адаптувати гру під різні мобільні платформи та забезпечити стабільну роботу на пристроях з обмеженими ресурсами. Впровадження аналітики та системи гейміфікації дало змогу відстежувати поведінку користувачів, оцінювати ефективність ігрових механік і оптимізувати рівні складності та систему винагород для підвищення залученості гравців.

Оцінка користувацького досвіду та інтеграція аналітичних даних дозволили виявити слабкі місця інтерфейсу, покращити навігацію, взаємодію з об’єктами та загальний комфорт гри. Комплексний підхід до розробки, тестування та оптимізації забезпечив створення мобільного продукту з високим рівнем занурення, зручним інтерфейсом, збалансованими механіками та високою привабливістю для користувачів.

РОЗДІЛ 3. ОРГАНІЗАЦІЯ УПРАВЛІННЯ ПРОЄКТОМ ТА АНАЛІЗ ЕФЕКТИВНОСТІ

3.1. Планування та організація командної роботи

Ефективна організація командної роботи є ключовим чинником успішного управління проєктом розробки мобільної гри, оскільки від злагодженої взаємодії між учасниками команди залежить якість, терміни виконання та стабільність кінцевого продукту. Планування командної діяльності передбачає визначення структури команди, ролей і зон відповідальності кожного учасника, а також вибір методології співпраці, що забезпечує максимальну ефективність роботи на всіх етапах розробки.

У процесі планування було визначено основні цілі та завдання команди, що включають розробку технічної архітектури гри, створення 3D-контенту, проєктування користувацького інтерфейсу, реалізацію ігрових механік, тестування та оптимізацію продукту для мобільних платформ. Для кожного етапу визначались конкретні deliverables (результати роботи) та критерії їхньої готовності. Такий підхід дозволив сформувати чіткий план-графік розробки, який слугував базою для контролю прогресу та своєчасного внесення коригувань.

Організаційна структура команди була побудована за принципом розподілу функціональних ролей. До складу команди входили керівник проєкту (Project Manager), технічний спеціаліст (Technical Lead), геймдизайнер, художник-моделер, UI/UX-дизайнер, програміст Unreal Engine та тестувальник. Така модель забезпечує чітку підзвітність, взаємозамінність учасників у межах суміжних компетенцій і дозволяє зменшити ризики затримок під час реалізації завдань.

Керівник проєкту відповідав за стратегічне планування, координацію роботи команди, контроль термінів і ресурсів, а також за комунікацію між усіма учасниками. Технічний спеціаліст здійснював нагляд за архітектурними

рішеннями, оптимізацією продуктивності та інтеграцією графічного контенту з ігровими механіками. Геймдизайнер займався розробкою ігрової логіки, сценаріїв, балансу рівнів і системи прогресії. Художник та UI/UX-дизайнер забезпечували візуальну складову - створення моделей, текстур, інтерфейсних елементів і загальну стилістику гри. Програміст реалізовував механіки та інтерактивні системи в середовищі Unreal Engine, а тестувальник перевіряв функціональність і стабільність продукту на різних пристроях.

Для організації командної взаємодії використовувалася методологія **Agile** із елементами **Scrum**, що передбачала ітераційний підхід до виконання завдань. Проєкт поділявся на короткі цикли - спринти тривалістю від одного до двох тижнів, наприкінці кожного з яких проводився аналіз результатів і постановка нових пріоритетів. Це дозволяло гнучко реагувати на зміни, коригувати технічні завдання й адаптувати процес до поточних умов. Для візуалізації прогресу використовувалася Kanban-дошка, де завдання розподілялися за статусами: *To Do, In Progress, Testing* та *Done*. Такий підхід забезпечував прозорість роботи та дозволяв відстежувати завантаження кожного члена команди.

Важливим елементом організації роботи стали щотижневі стендапи, на яких обговорювалися результати виконаних завдань, труднощі, а також коригувалися пріоритети на наступний спринт. Це сприяло підтриманню високого рівня комунікації, швидкому виявленню проблем і запобіганню дублюванню роботи. Крім того, кожен етап проєкту супроводжувався веденням документації у спільному середовищі (наприклад, Confluence або Google Workspace), що забезпечувало централізований доступ до технічних описів, діаграм, специфікацій і протоколів зустрічей.

Особливу увагу під час організації командної роботи приділяли управлінню часом і навантаженням. Для цього застосовувалися інструменти моніторингу завдань, такі як **Jira** та **Trello**, які дозволяли фіксувати прогрес виконання, виставляти пріоритети та розраховувати очікувану тривалість кожного етапу. Така система сприяла підвищенню дисципліни, відповідальності членів команди та мінімізації простоїв між завданнями.

Ефективність командної роботи також залежала від налагодження внутрішніх каналів комунікації. Основні робочі обговорення проводилися через **Slack** або **Discord**, що дозволяло швидко вирішувати поточні технічні питання та ділитися оновленнями в режимі реального часу. Використання таких платформ сприяло оперативності прийняття рішень, узгодженню дій між дизайнерами, програмістами та тестувальниками, а також формуванню здорової робочої атмосфери.

Загалом планування та організація командної роботи в межах проєкту мобільної гри забезпечили високу узгодженість дій між учасниками, прозорість процесів, ефективне управління ресурсами та контроль термінів виконання. Гнучка структура управління, застосування сучасних методологій і цифрових інструментів дозволили досягти балансу між творчою свободою розробників і дисципліною виробничого процесу, що є необхідною умовою для успішної реалізації будь-якого ІТ-проєкту у сфері мобільних ігор.

3.2. Розподіл ролей і відповідальності учасників проєкту

Організація процесу розробки мобільної гри передбачає чітке визначення ролей, функцій і меж відповідальності кожного учасника проєкту. У випадку індивідуального виконання роботи всі ці ролі були поєднані в одній особі, що вимагало комплексного підходу до планування, гнучкого розподілу часу та володіння широким спектром компетенцій у сферах управління, програмування, дизайну, тестування та аналітики.

Таке поєднання функцій має подвійний характер: з одного боку, воно ускладнює процес через необхідність самостійного прийняття рішень і відсутність можливості делегування завдань, але з іншого - забезпечує цілісність бачення, єдність стилю і повну узгодженість між усіма складовими продукту. В умовах індивідуальної розробки роль кожного етапу значно зростає, адже будь-яке рішення впливає на кінцеву якість гри та стабільність її роботи.

На початковому етапі реалізації проєкту першочерговою роллю стала функція **керівника проєкту (Project Manager)**. Вона передбачала визначення цілей, планування часових рамок, постановку завдань і контроль їх виконання. Було розроблено детальний план-графік роботи з розподілом етапів на підзадачі: створення технічного завдання, концептуальної документації, моделювання, програмування, тестування та оптимізації.

Важливу роль відіграло формування **матриці ризиків**, яка дозволила передбачити потенційні технічні або організаційні проблеми - затримки з реалізацією окремих компонентів, складнощі з оптимізацією продуктивності або інтеграцією контенту. Для кожного ризику було розроблено план реагування та механізми запобігання.

Організаційна діяльність також включала **контроль ресурсів**: часового, технічного та когнітивного. У процесі розробки здійснювався щотижневий аналіз виконання запланованих завдань і коригування графіка, що відповідало принципам ітераційного планування Agile.

Другим важливим напрямом стала роль **геймдизайнера (Game Designer)**, яка поєднала творчі та аналітичні завдання. На цьому етапі розробник виконував повний цикл створення концепції гри: визначення жанру, цільової аудиторії, ігрової логіки, сценарних подій, механік прогресії та балансу складності.

Було створено документ Game Design Document (GDD), який описував ключові елементи гри - сюжет, ігрові об’єкти, систему винагород, способи взаємодії користувача з середовищем та інтерфейсом. У ньому також визначалися сценарії ігрових подій, можливі стани об’єктів, умови переходу між рівнями та поведінка гравця під час виконання завдань.

Особлива увага приділялася **розробці логіки квестів** та інтерактивних елементів. Вони формували основу геймплею та визначали рівень залученості користувача. Для забезпечення збалансованості було проведено кілька ітерацій тестування прототипів, під час яких оцінювалися тривалість ігрових сесій, реакція на події та динаміка складності.

Геймдизайнерська діяльність також охоплювала створення діаграм потоків користувача (User Flow) та структури ігрового інтерфейсу (Wireframes), які слугували базою для подальшої розробки UI/UX-дизайну.

У межах проєкту роль **технічного спеціаліста (Technical Developer)** полягала в реалізації всієї логічної, програмної та інтеграційної частини гри. Розробка здійснювалася в середовищі **Unreal Engine 5**, що надало можливість поєднати візуальне програмування за допомогою **Blueprints** із точним контролем поведінки систем через **C++ API**.

Було створено модульну архітектуру проєкту, що складалася з окремих компонентів: системи управління ігровим процесом (Game Instance), контролера персонажа (Player Controller), інтерактивних об’єктів (Actors), системи підказок, діалогів та інтерфейсу користувача (Widget Blueprints). Кожен модуль реалізовувався з урахуванням принципів незалежності та повторного використання коду.

Важливим завданням технічної реалізації стала **оптимізація гри під мобільні пристрої**, зокрема зниження кількості полігонів у 3D-моделях, зменшення навантаження на графічний процесор та використання LOD-систем (Levels of Detail). Також було застосовано методи асинхронного завантаження ресурсів для скорочення часу запуску та покращення продуктивності.

Роль **3D-художника та моделера (3D Artist)** охоплювала створення візуальних об’єктів середовища, реквізиту, персонажів та інтерфейсних елементів. Для цього використовувалися інструменти **Blender** (моделювання та UV-розгортка), **Adobe Photoshop** (створення текстур та кольорових карт) і **Adobe Illustrator** (графічні елементи інтерфейсу).

Під час виконання цієї функції було розроблено набір оптимізованих моделей із низьким полігональним рівнем (Low Poly), адаптованих під мобільні обмеження. Кожен об’єкт мав власний матеріал, налаштований у системі **Material Editor Unreal Engine**, що дозволило досягти привабливого візуального результату без перевантаження сцени.

Додатково було розроблено кольорову палітру гри, що поєднувала теплі акценти з нейтральними відтінками, створюючи атмосферу пригодницького квесту. Ця стилістика підтримувалася в усьому контенті – від ігрового середовища до інтерфейсу.

Дизайн користувацького інтерфейсу

Роль **UI/UX-дизайнера** вимагала створення інтерфейсу, зручного для сенсорних екранів, що є критично важливим аспектом мобільної розробки. Особлива увага приділялася ергономіці, простоті навігації та візуальній чистоті екранів.

У межах цього напряму були створені прототипи меню, системи інвентарю, вікон діалогів, екранів завдань та індикаторів прогресу. Основна мета полягала у формуванні **інтуїтивної взаємодії** користувача з грою, коли більшість дій виконуються через зрозумілі візуальні сигнали, без необхідності додаткових інструкцій.

Юзабіліті-тестування показало, що мінімалістичний підхід до дизайну сприяє кращому сприйняттю контенту та зменшує навантаження на користувача, що особливо важливо для коротких мобільних сесій.

Тестувальна функція

Після завершення основних етапів розробки відбувалася реалізація ролі тестувальника (QA Engineer). Було розроблено план тестування, що включав кілька рівнів перевірки: функціональне тестування, стрес-тестування, перевірку логіки переходів між рівнями та інтеграційне тестування елементів інтерфейсу.

Під час тестування особливу увагу приділяли продуктивності на різних типах мобільних пристроїв. Для цього використовувалися інструменти Unreal Frontend та Session Frontend, що дозволяли моніторити FPS, навантаження на GPU та споживання пам’яті. За результатами тестування було оптимізовано розмір текстур, зменшено кількість динамічних тіней і скорочено використання складних матеріалів.

Інтеграція ролей і самоорганізація процесу

Поєднання всіх зазначених функцій вимагало побудови ефективної системи самоорганізації. Для цього використовувалися принципи тайм-менеджменту та спринтового планування. Робочий процес було розподілено на короткі цикли, у межах яких формулювалися конкретні цілі, визначалися пріоритети та критерії готовності завдань.

Для контролю прогресу використовувалися інструменти Trello та Google Sheets, які дозволяли фіксувати стан кожного завдання та візуалізувати виконання у форматі Kanban-дошки. Таким чином, навіть у разі індивідуальної розробки вдавалося підтримувати структурований підхід до управління процесом, характерний для командних проєктів.

Робота в режимі «одна особа - повний цикл розробки» мала як переваги, так і певні обмеження. Серед головних переваг - повна узгодженість усіх рішень, глибоке розуміння проєкту на технічному та концептуальному рівнях, можливість швидко приймати рішення без додаткових погоджень.

Водночас такий формат вимагав високого рівня самодисципліни, системності у плануванні та здатності поєднувати різні типи діяльності: аналітичну, творчу й технічну. Одним із головних викликів стала необхідність підтримувати баланс між розробкою, тестуванням та вдосконаленням продукту, що потребувало ретельного розподілу часу.

Попри це, індивідуальний підхід забезпечив комплексне бачення проєкту й став демонстрацією того, як сучасні методики управління (Agile, Kanban) і технологічні інструменти (Unreal Engine, Blender, Photoshop, Jira/Trello) дозволяють одній людині ефективно реалізувати повний виробничий цикл створення мобільної гри.

3.3. Використання інструментів контролю та моніторингу (Jira, Trello, GitHub)

Ефективне управління процесом розробки цифрового продукту, зокрема мобільної гри, неможливе без системи контролю, моніторингу виконання завдань і відстеження версій. Ці інструменти дозволяють структурувати робочий процес, фіксувати прогрес, аналізувати продуктивність, а також забезпечують надійну координацію дій навіть у випадку індивідуальної розробки. У межах даного проєкту були використані три основні платформи управління - Jira**,** Trello **та** GitHub, які виконували різні, але взаємодоповнювальні функції у системі управління життєвим циклом проєкту.

У сучасному середовищі розробки ігор використання спеціалізованих сервісів управління завданнями є стандартом, що дозволяє навіть невеликим командам або індивідуальним розробникам застосовувати професійні методики контролю. Ці системи реалізують принципи **Agile** та **Kanban**, які ґрунтуються на ітераційному плануванні, прозорості процесів і безперервному покращенні.

Для індивідуального розробника такі інструменти мають не лише організаційне, а й психологічне значення - вони сприяють підтриманню дисципліни, мотивації та фокусуванню уваги на пріоритетних завданнях. Системи Trello, Jira та GitHub дозволили забезпечити повний цикл управління - від створення технічного завдання до публікації готового прототипу гри.

Першим і базовим інструментом управління став **Trello** - онлайн-платформа для організації завдань за методом **Kanban**. Її перевага полягає у простоті та гнучкості, що дозволяє швидко адаптувати структуру під конкретний проєкт без складної конфігурації.

Для організації роботи в Trello була створена дошка «Mobile Game Development», яка містила кілька основних списків:

- Backlog - перелік усіх запланованих завдань, сформованих під час початкового етапу проєкту;

- To Do - поточні завдання, заплановані до виконання у найближчому спринті;

- In Progress - активні завдання, над якими вівся процес роботи;

- Testing - завдання, що проходили перевірку після реалізації;

- Done - завершені задачі, які пройшли тестування та були інтегровані до фінальної збірки.

Такий поділ дозволяв підтримувати візуальний контроль над процесом**,** бачити обсяг виконаних і запланованих робіт, а також оцінювати темп прогресу. Кожна картка містила опис завдання, орієнтовну складність, термін виконання та мітки (наприклад, UI, Blueprints, Optimization, Testing), що спрощувало навігацію й пошук.

Trello також використовувався для щотижневого самоаналізу, під час якого оцінювався відсоток виконаних завдань за спринт, виявлялися затримки або труднощі та формувалися пріоритети на наступний цикл. Така система допомагала дотримуватися ритму розробки, навіть за відсутності команди, та створювала відчуття структурованості процесу.

Використання Jira для технічного моніторингу та управління спринтами

На більш складних етапах проєкту, коли розробка наблизилася до інтеграції ігрових систем і тестування, було застосовано платформу Jira. Вона є професійним інструментом управління проєктами, орієнтованим на програмні продукти, і дозволяє детально контролювати кожен аспект технічної реалізації.

У Jira було створено проєкт із розподілом завдань за категоріями:

- Game Logic - розробка основних ігрових механік, поведінки персонажа, системи взаємодії з об’єктами;

- UI/UX Implementation - створення користувацького інтерфейсу, логіки меню та елементів навігації;

- Optimization - підвищення продуктивності, робота з LOD, зменшення кількості draw calls;

- Testing and Debugging - виправлення помилок, регресійне тестування, перевірка стабільності;

- Content Integration - імпорт моделей, текстур, анімацій і матеріалів у рушій Unreal Engine.

Кожне завдання мало власний статус (To Do, In Progress, Code Review, Testing, Done), що дозволяло вести точний моніторинг прогресу. Використання Jira дало змогу підтримувати Agile-підхід із чіткими спринтами, які тривали приблизно 10–14 днів.

На початку кожного спринту формувався Sprint Backlog - список завдань, які планувалося реалізувати в межах поточного циклу. Наприкінці - проводився Sprint Review, де оцінювалися результати, аналізувалися досягнення, помилки та визначалися наступні пріоритети. Такий процес допомагав не лише систематизувати роботу, а й дотримуватися стабільного темпу розробки без перенавантаження.

Додатковою перевагою Jira стала можливість звітування й аналітики: система автоматично формувала діаграми прогресу (Burndown Chart), які відображали співвідношення запланованих і виконаних завдань. Це дозволяло оцінювати ефективність роботи, прогнозувати терміни завершення окремих етапів і приймати рішення щодо оптимізації часу.

Використання GitHub для контролю версій і збереження коду

Третім важливим інструментом, що забезпечував технічну стабільність проєкту, став **GitHub** - сервіс для контролю версій і спільної роботи з кодом. Навіть при індивідуальній розробці використання Git є обов’язковою практикою, адже воно забезпечує надійне збереження змін, можливість повернення до попередніх версій і запобігає втраті даних.

Для проєкту було створено приватний репозиторій, який містив усі основні файли Unreal Engine, конфігурації Blueprint, сценарії, текстури та ресурси. Розробник використовував гілкову структуру (branching), де main гілка

відповідала стабільній версії гри, а dev - поточній робочій збірці, що перебувала у стані активної розробки.

Коміти виконувалися щоденно з короткими описами змін, наприклад:

“Added inventory widget logic”,

“Optimized lighting for mobile LOD”,

“Fixed quest system bug on level 2”.

Така практика дозволила підтримувати історію розвитку проєкту, документувати зміни та швидко знаходити джерело помилки у разі технічних збоїв.

Додатково GitHub використовувався як інструмент резервного зберігання - репозиторій синхронізувався із хмарним сховищем (Google Drive), що гарантувало безпечність даних навіть у випадку технічних проблем із робочою станцією.

Окремо було налаштовано систему Git LFS (Large File Storage), що дозволяла ефективно працювати з великими файлами Unreal Engine, зокрема текстурами та моделями, які перевищували стандартні обмеження Git. Це значно підвищило стабільність збереження та швидкість синхронізації.

Для досягнення максимальної ефективності ці три інструменти - Trello, Jira і GitHub - були інтегровані у спільну логічну систему управління.

Trello виконував роль стратегічного планувальника і візуальної панелі контролю. Тут відображалися всі основні завдання, поділені за категоріями, а також визначалися пріоритети.

Jira відповідала за детальний технічний моніторинг, зокрема відстеження стану задач, спринтів і аналіз ефективності.

GitHub слугував репозиторієм коду та медіаресурсів, забезпечуючи версійність і стабільність проєкту.

Синхронізація між платформами здійснювалася через ручні оновлення статусів і прив’язку комітів до Jira- або Trello-завдань за допомогою коротких кодів (наприклад, TASK-34: UI fix). Це дозволяло забезпечити **повну**

простежуваність (traceability) - від створення завдання до його реалізації та фінального тестування.

Ключовим аспектом моніторингу став **самоконтроль продуктивності**, який здійснювався через поєднання звітності в Jira та оглядових сесій у Trello. Щотижня проводився аналіз обсягу виконаної роботи, кількості завершених завдань, середнього часу на задачу та дотримання дедлайнів.

Цей підхід дозволяв визначати «вузькі місця» - етапи, що займали надмірно багато часу, або повторювані проблеми у виробничому циклі. На основі аналізу вносилися зміни до структури спринтів, переглядалися пріоритети та уточнювалися критерії готовності продукту (Definition of Done).

Також було розроблено індивідуальну систему мотиваційного контролю, де кожен завершений спринт супроводжувався коротким підсумковим звітом і позначенням досягнень, що підтримувало стабільний рівень мотивації під час тривалої розробки.

Використання зазначених платформ продемонструвало низку переваг у контексті індивідуального проєкту:

1. **Прозорість процесу** - завдяки візуальному відображенню всіх завдань і статусів.
2. **Контроль часу** - можливість чіткого відстеження термінів виконання.
3. **Стабільність версій** - завдяки GitHub зберігалася повна історія змін.
4. **Підвищення якості коду** - через систематизацію ітерацій і контроль зворотного зв’язку.
5. **Гнучкість** - можливість швидкої адаптації структури завдань до змін у процесі розробки.

Використання систем Jira, Trello та GitHub стало важливою частиною управлінської архітектури проєкту, що забезпечило ефективну організацію процесу навіть при одноосібній розробці. Інтеграція цих інструментів дозволила

досягти високої прозорості, передбачуваності результатів та стабільності програмного продукту.

Поєднання методик Agile і Kanban із цифровими системами контролю створило умови для структурованого, гнучкого й адаптивного управління розробкою мобільної гри, що є показовим прикладом сучасної моделі самоорганізованого виробничого процесу в ІТ-сфері.

3.4. Управління ризиками та прийняття управлінських рішень

Управління ризиками є одним із ключових елементів системи управління будь-яким ІТ-проєктом. Для індивідуального розробника мобільної гри воно набуває ще більшої важливості, оскільки всі критичні рішення, технічні вибори, часові витрати та організаційні аспекти концентруються в одній особі. Грамотно побудована система ризик-менеджменту дозволяє не лише мінімізувати потенційні загрози, а й перетворювати їх на можливості для покращення процесу та кінцевого результату.

Ризик у контексті ІТ-проєкту визначається як ймовірна подія або умова, яка може негативно вплинути на досягнення цілей проєкту - терміни, вартість, якість або функціональність продукту. Для проєкту розробки мобільної гри такі ризики охоплюють як технічні аспекти (збої, помилки, нестабільність рушія), так і організаційні (нестача часу, перевтома, неправильне планування, відсутність зворотного зв’язку).

Управління ризиками відбувається за класичною схемою, що включає чотири основні етапи:

1. **Ідентифікація ризиків** - визначення потенційних проблем, що можуть вплинути на проєкт.
2. **Оцінювання ризиків** - аналіз ймовірності їх виникнення та ступеня впливу.
3. **План реагування** - розробка заходів щодо уникнення, зменшення або прийняття ризику.

4. **Моніторинг** - постійне спостереження за змінами та коригування плану управління.

У межах розробки гри цей процес здійснювався паралельно з технічними та дизайнерськими етапами, що відповідало гнучкому принципу **Agile-менеджменту**, де ризики розглядаються як динамічний елемент, що потребує регулярного переоцінювання.

Класифікація ризиків у проєкті мобільної гри

Для систематизації потенційних загроз усі ризики були поділені на п’ять груп:

1. **Технічні ризики** - пов’язані з нестабільністю рушія Unreal Engine, конфліктами плагінів, оптимізацією під мобільні пристрої, втратою даних.

2. **Організаційні ризики** - недотримання графіка, перевантаження робочого процесу, неефективний розподіл часу.

3. **Дизайнерські ризики** - невідповідність художнього стилю, складність інтерфейсу, відсутність балансу між візуальною привабливістю та продуктивністю.

4. **Психологічні ризики** - зниження мотивації, емоційне вигорання, втрата інтересу на середніх етапах розробки.

5. **Зовнішні ризики** - технічні збої обладнання, оновлення програмного забезпечення, обмеження платформи чи зміни в політиці публікації ігор (Google Play, App Store).

Кожна з цих груп оцінювалася за двома критеріями: ймовірність виникнення **(P)** та вплив на проєкт (I). На основі цієї оцінки було створено **матрицю ризиків**, яка дозволила визначити пріоритети реагування.

1. Втрата даних або пошкодження проєкту.
Це один із найпоширеніших ризиків при роботі з великими файлами Unreal Engine. Для його мінімізації було застосовано систему контролю версій GitHub**,** а також автоматичне резервне копіювання на Google Drive. Такий підхід гарантував збереження проміжних версій проєкту й можливість швидкого відновлення після технічних збоїв.

2. Оптимізація продуктивності на мобільних пристроях. Оскільки рушій Unreal Engine орієнтований переважно на ПК і консолі, одним із головних ризиків стала низька продуктивність на смартфонах. Для зниження цього ризику було впроваджено такі заходи:

- використання LOD-системи (Levels of Detail) для моделей;
- скорочення кількості полігонів і текстур високої роздільності;
- зменшення кількості динамічних тіней;
- тестування на різних пристроях із різною продуктивністю.

Ці дії дозволили зберегти стабільну частоту кадрів і забезпечити комфортний користувацький досвід.

3. Перевантаження розробника та втрата мотивації. Індивідуальна розробка потребує високої концентрації та самодисципліни, тому ризик психологічного виснаження був досить імовірним. Для його мінімізації використовувалася система спринтового планування, що дозволяла розподіляти робочі цикли на короткі, досяжні етапи. Кожен завершений спринт супроводжувався аналізом досягнень, що підтримувало мотивацію та відчуття поступового прогресу.

Крім того, запроваджувалася методика Pomodoro - чергування інтенсивних робочих сесій (по 50 хвилин) із короткими перервами. Це сприяло збереженню продуктивності та зменшувало ризик перевтоми.

4. Затримки у виконанні плану через технічну складність. Під час реалізації складних систем (наприклад, інвентарю або діалогової логіки) виникали затримки через потребу у вивченні додаткових функцій рушія. Для зменшення цього ризику створювався резерв часу (buffer time) у кожному спринті, який дозволяв компенсувати можливі відставання без порушення загального графіка.

5. Відсутність зовнішнього зворотного зв’язку. Одним із недоліків індивідуальної розробки є відсутність об’єктивної оцінки роботи. Для часткової компенсації цього ризику використовувалися **тестові сесії** із залученням невеликої групи користувачів (студентів, колег, знайомих), які

надавали відгуки щодо ігрового процесу та інтерфейсу. На основі отриманих коментарів проводилися дрібні коригування балансу, дизайну та UX-елементів.

Для управління ризиками використовувалися як якісні, так і кількісні методи оцінки**.**

Якісний аналіз передбачав суб'єктивне визначення рівня ризику (низький, середній, високий) на основі досвіду та попередніх результатів. Кількісна оцінка проводилася через розрахунок індексу ризику (IR) за формулою:

IR=P×IIR = P \times IIR=P×I

де **P** - ймовірність настання події (0–1), а **I** - рівень її впливу (0–10). Ризики з індексом понад 5 вважалися критичними та потребували негайного реагування.

Наприклад, ризик «втрата даних» мав P = 0,6 і I = 10, що давало IR = 6,0 - тобто високий рівень. Тому було впроваджено триступеневу систему резервного копіювання: GitHub, локальний диск, Google Drive.

Після ідентифікації та оцінки ризиків було сформовано **план реагування**, який охоплював три основні стратегії:

Уникнення ризику (Avoidance) - зміна підходу до виконання завдання, щоб повністю усунути загрозу (наприклад, відмова від використання нестабільних плагінів).

Зменшення ризику (Mitigation) - упровадження технічних або організаційних заходів для зниження ймовірності виникнення (наприклад, оптимізація графічних налаштувань, спрощення UI).

Прийняття ризику (Acceptance) - погодження з можливим наслідком у разі, якщо його вплив є незначним або вартість запобігання перевищує користь.

Реалізація цього плану забезпечувала гнучкість управління - ризики оцінювалися не лише на початку проєкту, а й після завершення кожного спринту. У такий спосіб відбувався **динамічний контроль**, що є невід'ємною частиною методології Agile.

Процес прийняття управлінських рішень

Прийняття рішень у межах індивідуального IT-проєкту поєднувало аналітичний, інтуїтивний та експериментальний підходи. Умовно процес складався з таких етапів:

1. **Виявлення проблеми або потреби** - фіксація відхилення від плану чи технічної помилки.
2. **Аналіз варіантів** - оцінка можливих шляхів розв'язання, включно з перевагами й обмеженнями кожного.
3. **Прогнозування наслідків** - моделювання можливих результатів обраного рішення.
4. **Прийняття рішення** - вибір оптимальної стратегії з урахуванням доступних ресурсів.
5. **Реалізація та перевірка** - застосування рішення й оцінювання його ефективності.

Особливістю прийняття рішень у межах одного виконавця є необхідність поєднання ролей аналітика, менеджера й технічного спеціаліста, що вимагає балансу між раціональністю та творчістю. Багато технічних рішень приймалися за принципом "прототипування через експеримент"**:** створення короткого варіанта системи, перевірка її ефективності й подальше вдосконалення.

Для обґрунтування управлінських рішень активно використовувалися інструменти Jira**,** Trello та GitHub, описані в попередньому підрозділі.

- Jira слугувала джерелом аналітичних даних про швидкість виконання завдань і структуру спринтів, що дозволяло приймати рішення на основі метрик.
- Trello забезпечувала візуальне уявлення про стан завдань і дозволяла швидко коригувати пріоритети.
- GitHub давав змогу аналізувати історію змін і відстежувати наслідки технічних рішень у часі.

Таким чином, процес прийняття рішень базувався на даних (data-driven approach), що підвищувало обґрунтованість і передбачуваність результатів.

Психологічні аспекти прийняття рішень

В умовах індивідуальної розробки важливо враховувати не лише технічну, а й емоційно-когнітивну складову процесу прийняття рішень. Розробник стикається з необхідністю постійного вибору між швидкістю виконання та якістю, між експериментом і стабільністю. Тому ефективним було застосування принципів раціональної самооцінки - фіксації помилок і висновків у спеціальному «щоденнику розробки», що дозволяло аналізувати свої рішення з часом і виявляти закономірності в прийнятті.

Також застосовувалися елементи когнітивного рефреймінгу - переосмислення проблеми не як перешкоди, а як виклику або можливості вдосконалення продукту. Це сприяло збереженню психологічної стійкості й запобігало емоційному вигоранню.

Система управління ризиками та прийняття управлінських рішень у межах індивідуальної розробки мобільної гри продемонструвала ефективність гнучкого підходу, орієнтованого на постійний аналіз і швидку адаптацію. Застосування інструментів моніторингу (Jira, Trello, GitHub), методів планування (Agile, Kanban) та принципів самоорганізації дозволило мінімізувати вплив зовнішніх і внутрішніх ризиків.

Використання системного підходу до оцінювання загроз, створення плану реагування, регулярне тестування рішень і ведення документації забезпечили стабільність проєкту, високу якість кінцевого продукту та раціональне використання часу.

Таким чином, управління ризиками стало не лише інструментом контролю, а й важливим засобом підвищення ефективності творчого процесу, що визначає успішність сучасних IT-проєктів, навіть у разі їхньої індивідуальної реалізації.

3.5. Аналіз ефективності реалізованих рішень та оптимізація процесу

Аналіз ефективності реалізованих рішень є завершальним і водночас одним із найважливіших етапів управління проєктом, адже саме він дає змогу оцінити ступінь досягнення поставлених цілей, визначити сильні та слабкі сторони реалізації, а також сформувати напрями подальшого вдосконалення. У контексті індивідуальної розробки мобільної гри цей етап має особливе значення, оскільки він дозволяє комплексно оцінити не лише технічні результати, а й ефективність обраних підходів до самоорганізації, планування та управління процесом.

Методологічна основа аналізу ефективності спиралася на поєднання трьох аспектів - технічного, організаційного та творчо-дизайнерського. Такий підхід дав можливість оцінити проєкт не лише як сукупність технічних рішень, а як цілісну систему, у якій процес і результат взаємопов’язані. Для кожного аспекту застосовувався принцип порівняння запланованих результатів із фактичними, що дозволило виявити відхилення, оцінити причини можливих невідповідностей і визначити фактори успіху.

У технічному вимірі головна увага приділялася стабільності роботи гри, її продуктивності та оптимізації. На етапі тестування було зафіксовано, що час завантаження не перевищував десяти секунд, а середня частота кадрів трималася на рівні 50–60 FPS на більшості сучасних смартфонів. Завдяки впровадженню системи рівнів деталізації моделей (LOD) і використанню асинхронного завантаження ресурсів вдалося суттєво знизити навантаження на графічний процесор. Особливо важливим результатом стала оптимізація структури об’єктів за допомогою компонентів Instanced Static Mesh, що забезпечило економію ресурсів і стабільну роботу навіть на пристроях середнього рівня. Порівняльний аналіз продуктивності між початковою та фінальною збірками показав покращення FPS майже на третину, скорочення часу завантаження рівнів на третину та зменшення споживання пам’яті на майже п’яту частину. Це

підтверджує, що прийняті технічні рішення були обґрунтованими, а застосовані методи оптимізації - ефективними.

Не менш важливим напрямом аналізу стала організаційна ефективність, яка визначалася рівнем дисципліни, послідовності виконання завдань і здатністю гнучко реагувати на зміни. Попри те, що розробка здійснювалася одноосібно, структура управління будувалася на принципах Agile із використанням спринтового планування, щотижневого моніторингу й аналізу прогресу. Результати показали, що понад дев’яносто відсотків завдань виконувалися в межах запланованих строків, а відхилення у термінах були мінімальними. Поступово підвищувалася й особиста продуктивність: якщо на початкових етапах виконання одного завдання займало близько двох днів, то в кінці розробки цей показник скоротився майже на третину. Значну роль у цьому відіграло використання цифрових інструментів керування, зокрема Trello і Jira, які дозволяли структурувати процес, контролювати прогрес і підтримувати візуальну прозорість робочих етапів. Такий підхід сприяв формуванню звички до системного планування, що особливо важливо для індивідуальних розробників.

Крім того, у процесі роботи активно застосовувалися принципи тайм-менеджменту, зокрема методика Pomodoro, яка передбачала чергування інтенсивних робочих сесій і коротких перерв. Це допомогло уникнути перевтоми, зберігати концентрацію й підтримувати стабільний темп. Окремо варто зазначити, що у плані організаційної ефективності було передбачено резерв часу для непередбачених обставин, що дозволило мінімізувати ризик зриву термінів і водночас залишити простір для доопрацювання деталей на фінальному етапі.

Третій аспект аналізу - творчо-дизайнерська ефективність - стосувався оцінювання візуальної складової, логіки геймплею та зручності взаємодії користувача з грою. Створений художній стиль відзначався поєднанням мінімалістичної графіки з яскравими акцентами, що забезпечувало добру читабельність об’єктів на екранах різних розмірів. Кольорова гама та загальна

композиція рівнів формували атмосферу пригодницького квесту, у якому кожен візуальний елемент підтримував сюжетну ідею. Значну увагу було приділено створенню інтерфейсу користувача, який виявився інтуїтивно зрозумілим і зручним для сенсорного керування. Тестування підтвердило високу якість UX-рішень: більшість користувачів зазначили, що можуть орієнтуватися в інтерфейсі без додаткових інструкцій, а взаємодія з ігровими елементами відбувається природно. Геймплей, побудований на квестовій логіці, показав збалансованість між простотою і викликом, що підтримує зацікавлення гравця протягом усієї сесії. Таким чином, творча концепція гри збереглася на всіх етапах реалізації, що свідчить про послідовність бачення і цілісність підходу до розробки.

Не менш важливим складником аналізу стала оптимізація самого процесу розробки. У міру накопичення досвіду структура роботи вдосконалювалася, що позначилося на якості результату. Технічна оптимізація полягала у скороченні кількості матеріалів і текстур за рахунок їх уніфікації, впровадженні текстурних атласів для зменшення звернень до графічного процесора, створенні бібліотек макросів у Blueprints для повторного використання логіки та впорядкуванні даних через Data Tables. Ці зміни дозволили підвищити ефективність коду, прискорити компіляцію та зробити проєкт більш гнучким для майбутнього розширення.

Водночас оптимізація торкнулася і організаційних аспектів. На основі аналізу перших ітерацій було скориговано кількість завдань, що виконувалися одночасно, адже надлишок відкритих задач призводив до зниження концентрації. Відтак було ухвалено рішення фокусуватися на максимум трьох завданнях у межах одного спринту, що позитивно вплинуло на продуктивність. Також переглянуто систему оцінювання трудомісткості робіт: початкові прогнози виявилися завищеними, тому для більшої точності застосовувалася оцінка на основі середнього фактичного часу виконання завдань. Ця зміна дала змогу підвищити точність планування й зменшити ризик перевантаження.

Важливою частиною оптимізації процесу стала практика щоденного короткого самоаналізу, яка полягала у фіксації виконаних завдань, труднощів і пропозицій щодо покращення. Такий підхід сприяв розвитку навичок саморефлексії, що є ключовим елементом особистісного управління проєктом. Самоаналіз допомагав своєчасно виявляти повторювані помилки, формувати реалістичні очікування та підтримувати внутрішню мотивацію, особливо під час складних або рутинних етапів роботи.

Результати комплексного аналізу показали, що реалізовані технічні, організаційні та дизайнерські рішення були ефективними та взаємодоповнювальними. З технічного боку, гра продемонструвала стабільну роботу, високу оптимізацію та якісну реалізацію основних механік. З організаційного боку, застосування гнучких методик планування забезпечило чітку структуру процесу, дисципліну та контроль. З творчого боку, збережено послідовність художнього бачення, а інтерфейс і геймплей відповідали початковій концепції. Усе це свідчить про те, що поєднання методологій Agile і Kanban з індивідуальною системою самоорганізації виявилося ефективним рішенням для управління одноосібним IT-проєктом.

Підсумовуючи результати, можна зазначити, що обрані підходи забезпечили не лише якісну реалізацію продукту, а й створили основу для подальшого вдосконалення процесів розробки. Ефективність проєкту проявилася у стабільному темпі роботи, відсутності критичних помилок, досягненні високих показників продуктивності та збереженні узгодженості між усіма елементами гри. Отже, проведений аналіз підтверджує раціональність прийнятих рішень і демонструє, що навіть індивідуальний розробник, використовуючи сучасні інструменти управління та гнучкі методології, здатен організувати процес на рівні професійної команди.

Таким чином, аналіз ефективності реалізованих рішень у поєднанні з проведеною оптимізацією процесу засвідчив високий рівень результативності обраної стратегії розробки. Досягнення балансу між технічними, організаційними та творчими аспектами стало запорукою цілісності проєкту, а

сформований досвід - базою для подальшого розвитку практик управління в майбутніх етапах роботи над іграми та інтерактивними застосунками.

3.6. Рекомендації щодо покращення управління подальшими етапами розробки

Ефективність управління IT-проєктом значною мірою визначається не лише результатами поточного етапу, а й здатністю розробника або команди здійснювати постійне вдосконалення процесів. У розробці мобільної гри, навіть якщо вона виконувалася одноосібно, важливо не лише досягти кінцевої мети, але й сформувати систему безперервного розвитку, що дозволяє масштабувати проєкт, удосконалювати його функціональність і підвищувати рівень якості майбутніх продуктів. Саме тому підсумковий етап дослідження передбачає розробку рекомендацій щодо покращення управління наступними фазами розробки, зокрема вдосконалення технічних процесів, розширення комунікаційної структури, підвищення ефективності планування та посилення контролю якості.

Проведений аналіз продемонстрував, що обрана структура управління на основі гнучких методологій Agile і Kanban виявилася ефективною для індивідуальної роботи. Однак для подальших етапів доцільно розглянути можливість поглиблення цього підходу шляхом переходу до моделі, яка передбачає часткову командну взаємодію. Залучення хоча б двох-трьох фахівців різного профілю дозволило б збалансувати навантаження, розділити відповідальність і прискорити процеси тестування та ітераційного вдосконалення. У такому випадку система управління мала б бути розширена до формату Scrum із чітко визначеними ролями: керівник проєкту, розробник, художник і тестувальник. Така структура дала б змогу зберегти гнучкість планування, але водночас створити чітку вертикаль прийняття рішень і забезпечити ефективний зворотний зв’язок у команді.

У контексті подальшого розвитку проєкту доцільно вдосконалити процес планування через використання системи довгострокового прогнозування. Якщо попередні етапи роботи зосереджувалися на коротких спринтах тривалістю до двох тижнів, то на наступних фазах варто впровадити елементи середньострокового планування - визначення стратегічних цілей на місяць або квартал із подальшою деталізацією до окремих завдань. Такий підхід дозволить підвищити передбачуваність процесу, чіткіше розподіляти ресурси й мінімізувати ризик втрати загальної перспективи проєкту. Крім того, він сприятиме кращій координації при можливому масштабуванні розробки, коли кількість учасників або обсяг робіт зростатиме.

Особливу увагу на подальших етапах слід приділити вдосконаленню комунікаційних процесів. Навіть у разі індивідуальної розробки важливо підтримувати систему зовнішнього зворотного зв’язку - тестування з користувачами, консультації з фахівцями, участь у професійних спільнотах розробників. Такі взаємодії дозволяють отримувати незалежну оцінку якості продукту, виявляти неочевидні проблеми у зручності чи функціональності, а також формувати реалістичне уявлення про ринкові очікування. У майбутньому доцільно запровадити періодичне бета-тестування з залученням обмеженого кола користувачів, що дасть змогу перевіряти нові механіки, оптимізацію або сценарні рішення до моменту їхньої фінальної інтеграції.

З технічного погляду одним із напрямів розвитку має стати впровадження системи безперервної інтеграції та розгортання (Continuous Integration / Continuous Deployment, CI/CD). Для цього можна використовувати сервіси GitHub Actions або Jenkins, які автоматизують процес збірки, тестування та розгортання гри. Це дозволить зменшити кількість ручних дій, підвищити стабільність і скоротити час між розробкою нової функції та її перевіркою на цільовому пристрої. Автоматизація таких процесів особливо актуальна для мобільної розробки, де навіть незначні зміни можуть вплинути на стабільність роботи застосунку. Застосування CI/CD також сприятиме кращому

документуванню змін і формуванню історії оновлень, що є важливою складовою професійного підходу до управління продуктом.

У межах майбутнього вдосконалення управління варто також посилити систему контролю якості. На ранніх етапах тестування проводилося переважно вручну, тому подальшим кроком має стати часткова автоматизація перевірок. Це може включати скрипти для перевірки продуктивності, аналіз логів і моніторинг стабільності кадрів під час тривалого використання гри. Крім того, варто розглянути можливість використання спеціалізованих інструментів тестування мобільних додатків, таких як Firebase Test Lab або Appium, які дозволяють відтворювати роботу гри на різних моделях пристроїв і виявляти проблеми, недоступні під час локальних тестів. Такий підхід забезпечить об’єктивнішу оцінку якості продукту й дозволить підтримувати стабільність під час масштабування функціоналу.

Окремого значення у подальшій розробці набуває питання підвищення ефективності використання часу та ресурсів. У майбутніх ітераціях можна розглянути впровадження системи аналітики часу, що дозволить відстежувати реальні витрати на кожен етап розробки, порівнювати їх із запланованими й визначати ділянки, де відбуваються перевитрати. Це допоможе оптимізувати планування і зменшити навантаження в періоди пікової активності. Крім того, варто формалізувати процедури документування процесів - створення шаблонів технічних завдань, стандартів кодування та описів архітектури. Такий підхід забезпечить більшу структурованість роботи та спростить потенційне залучення нових учасників у разі переходу до командного формату.

Ще одним важливим напрямом вдосконалення управління може стати розширення системи метрик для оцінки успішності проєкту. Якщо на поточному етапі ефективність визначалася переважно за технічними показниками, то надалі доцільно додати показники користувацької взаємодії - час, проведений у грі, кількість повторних запусків, утримання користувачів і рівень задоволеності. Для збору таких даних можна інтегрувати інструменти аналітики, наприклад Google Analytics for Firebase. Це дозволить не лише покращити гру як продукт, а

й приймати управлінські рішення на основі реальних статистичних даних, що є ознакою зрілої продуктової стратегії.

Важливим елементом розвитку управлінських практик є підвищення якості прийняття рішень через більш глибокий аналіз даних. На подальших етапах розробки варто активно застосовувати підхід data-driven management - ухвалення рішень на основі кількісних показників, а не інтуїції чи припущень. Це стосується як вибору технічних рішень, так і визначення пріоритетів розвитку гри. Наприклад, аналіз часу завантаження рівнів, FPS або кількості помилок під час тестів може безпосередньо впливати на черговість виконання завдань. Такий підхід сприятиме підвищенню об’єктивності управління й забезпечить прозорість у плануванні подальших етапів роботи.

Ще однією перспективною рекомендацією є удосконалення комунікаційної екосистеми через інтеграцію більшої кількості інструментів командної взаємодії. Навіть у разі індивідуальної розробки корисним є використання середовищ спільної роботи, таких як Notion або Confluence, які дозволяють зберігати документацію, технічні специфікації та щоденники розробки у структурованому вигляді. Це спрощує доступ до інформації, дозволяє швидше оновлювати проєктну базу знань і забезпечує єдиний стандарт ведення документації. Крім того, такий підхід створює підґрунтя для майбутнього залучення інших спеціалістів без необхідності адаптації до нової системи.

Під час наступних етапів розробки доцільно приділити увагу вдосконаленню системи прийняття рішень. Досвід поточного етапу показав, що найефективнішими були рішення, прийняті на основі експериментального підходу - створення прототипу, тестування гіпотези, аналіз результатів і подальше вдосконалення. У майбутньому варто формалізувати цей підхід через створення внутрішнього циклу експериментів, де кожне нововведення проходить коротку ітерацію тестування перед впровадженням у стабільну версію. Така практика, відома як rapid prototyping, дає змогу швидко перевіряти ідеї, знижувати ризики невдалих рішень і водночас стимулює творчий розвиток проєкту.

Не менш важливим напрямом покращення управління є розвиток особистих управлінських навичок розробника. У подальшій діяльності доцільно акцентувати увагу на вдосконаленні компетенцій у сфері проєктного менеджменту, зокрема у вивченні таких напрямів, як управління ризиками, аналіз зацікавлених сторін, фінансове планування та методи комунікації в команді. Підвищення рівня теоретичної підготовки у цих питаннях забезпечить здатність масштабувати роботу, формувати команди й керувати ними відповідно до сучасних стандартів галузі. Це, у свою чергу, сприятиме переходу від рівня індивідуального виконавця до ролі лідера або продюсера проєкту.

Узагальнюючи результати проведеного аналізу, можна стверджувати, що подальше вдосконалення управління процесом розробки має базуватися на принципах гнучкості, структурованості та безперервного розвитку. Для цього необхідно поєднувати технічні інновації, автоматизацію рутинних процесів, систематизацію документації та аналітичний підхід до прийняття рішень. Запровадження цих рекомендацій дозволить не лише підвищити якість майбутніх проєктів, але й створить умови для переходу до повноцінної студійної форми роботи, де управлінські процеси будуть масштабованими, а результат - стабільним і конкурентоспроможним.

ВИСНОВКИ

У результаті виконання магістерської роботи досягнуто поставленої мети - досліджено, обґрунтовано та практично реалізовано процес управління проєктом розробки мобільної гри з використанням рушія Unreal Engine 5. Комплексний підхід, що поєднав теоретичний аналіз, проєктування, практичну реалізацію та оцінку ефективності, дозволив створити прототип мобільного продукту, який відповідає сучасним вимогам до якості, продуктивності та користувацького досвіду.

У першому розділі проведено теоретичне дослідження особливостей мобільної ігрової індустрії та сучасних підходів до управління IT-проєктами. Детально розглянуто тенденції розвитку мобільних ігор, визначено основні жанри, геймплейні механіки та фактори, що впливають на рівень залученості користувача. Особлива увага приділена квестовим іграм як жанру, який вимагає комплексної побудови сюжету, логічних завдань і візуально насиченого середовища.

На основі аналізу рушіїв (Unity, Godot, CryEngine, Unreal Engine) обґрунтовано вибір Unreal Engine 5 як найдоцільнішого інструмента для реалізації мобільної гри завдяки його графічним можливостям, інтегрованим інструментам візуального програмування Blueprints, підтримці C++, системам контролю версій, широкій спільноті та оптимізації під різні платформи. Розглянуто актуальні методології управління проєктами - Agile, Scrum, Kanban - які дозволяють гнучко організовувати командну роботу, контролювати виконання завдань та своєчасно реагувати на зміни у процесі розробки. Визначено, що впровадження таких методів у процес розроблення ігор підвищує прозорість роботи, зменшує ризики затримок і підвищує якість кінцевого продукту.

Другий розділ присвячено безпосередньому проєктуванню та реалізації мобільної гри на Unreal Engine 5. На етапі концептуальної розробки сформульовано ідею гри «Color Rush», визначено її цілі, цільову аудиторію,

базові механіки, систему рівнів і принципи взаємодії користувача з ігровим світом. Створено технічне завдання, що включає опис функціональних вимог, сценарії користування, архітектуру даних і специфікації контенту.

У процесі реалізації було розроблено 3D-моделі, матеріали, текстури та елементи інтерфейсу за допомогою Blender, Adobe Photoshop та Illustrator. Впроваджено логіку ігрового процесу на основі Blueprint-системи, що забезпечило швидке тестування і гнучке налаштування геймплейних параметрів. Оптимізаційні заходи, зокрема використання техніки злиття статичних мешів, зменшення кількості динамічних тіней та застосування технологій Nanite і Lumen, дозволили досягти стабільної частоти кадрів і високої якості зображення на мобільних пристроях різного рівня.

Особливу увагу приділено тестуванню, гейміфікації та аналітиці. Тестування дало змогу виявити вузькі місця в дизайні, покращити навігацію, зменшити затримки, адаптувати інтерфейс під різні розміри екранів і забезпечити інтуїтивність взаємодії. Впровадження системи аналітики дозволило відстежувати поведінку користувачів, аналізувати ефективність ігрових механік і налаштовувати рівні складності, що в результаті підвищило залученість гравців.

У третьому розділі розглянуто організацію управління проєктом, планування та аналіз ефективності виконаних робіт. Розроблено структуру управління, визначено ролі, функції й відповідальність учасників проєкту. У разі індивідуальної розробки всі функції - від менеджера проєкту до дизайнера й програміста - були поєднані в одній особі, що вимагало ретельного самоменеджменту, чіткої дисципліни та контролю за власною продуктивністю.

Для планування та моніторингу використовувалася система **Trello**, яка забезпечила наочну візуалізацію процесу через дошки та картки за методологією **Kanban**. Це дозволило контролювати етапи роботи, визначати пріоритети, фіксувати проблеми та здійснювати самооцінку прогресу наприкінці кожного спринту.

Визначено основні ризики, притаманні розробці індивідуальних проєктів (перевантаження, технічні труднощі, обмеження ресурсів) і розроблено стратегії їхнього мінімізації - резервування часу, використання технік Pomodoro, чітке планування коротких циклів і створення буферів для критичних завдань.

Отримані результати засвідчили, що ефективне управління проєктом мобільної гри базується на синергії технічних та організаційних рішень. Застосування інструментів управління (Trello, Scrum, Kanban) у поєднанні з потужним рушієм Unreal Engine 5 забезпечує контроль якості, зниження ризиків, підвищення продуктивності та дотримання термінів. Розроблений прототип гри підтвердив доцільність обраного підходу: гра має стабільну продуктивність, привабливу графіку, зрозумілий інтерфейс, збалансовану систему завдань і підтримує зацікавленість користувачів. Проєкт став прикладом поєднання творчих і технічних аспектів у межах системного управління життєвим циклом IT-продукту.

Практична цінність роботи полягає у створенні цілісної моделі управління розробкою мобільної гри, яка може бути адаптована для колективних і комерційних проєктів. Методичні підходи, описані у роботі, можуть бути використані для навчання майбутніх менеджерів IT-проєктів, а також у компаніях, що спеціалізуються на мобільних іграх, як орієнтир для організації внутрішніх процесів.

Теоретичне значення полягає у систематизації знань про сучасні рушії, методи управління та аналітичні інструменти, що сприяє розвитку інтегрованих моделей управління проєктами в ігровій індустрії.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Agile Alliance. Agile 101: The Basics of Agile Software Development. URL: https://www.agilealliance.org/agile101/ (дата звернення: 20.11.2024).

2. Atlassian. What is Kanban? URL: https://www.atlassian.com/agile/kanban (дата звернення: 18.11.2024).

3. Blender Foundation. Blender 4.0 Manual. URL: https://docs.blender.org/manual/en/latest/ (дата звернення: 25.10.2024).

4. Chandler P. Game Design Essentials: 20 Open Questions. Boston: Course Technology, 2020. 328 p.

5. Epic Games. Unreal Engine 5 Documentation. URL: https://docs.unrealengine.com/5.3/en-US/ (дата звернення: 15.11.2024).

6. Epic Games. Blueprint Visual Scripting. URL: https://docs.unrealengine.com/5.3/en-US/blueprints-visual-scripting-in-unreal-engine/ (дата звернення: 22.11.2024).

7. Epic Games. Nanite Virtualized Geometry. URL: https://docs.unrealengine.com/5.3/en-US/nanite-virtualized-geometry-in-unreal-engine/ (дата звернення: 12.11.2024).

8. Epic Games. Lumen Global Illumination. URL: https://docs.unrealengine.com/5.3/en-US/lumen-global-illumination-and-reflections-in-unreal-engine/ (дата звернення: 12.11.2024).

9. Firebase. Google Analytics for Firebase. URL: https://firebase.google.com/docs/analytics (дата звернення: 28.11.2024).

10. Fullerton T. Game Design Workshop: A Playcentric Approach to Creating Innovative Games. 4th ed. Boca Raton: CRC Press, 2019. 656 p.

11. GitHub. GitHub Documentation. URL: https://docs.github.com/en (дата звернення: 19.11.2024).

12. Jira Software. What is Scrum? URL: https://www.atlassian.com/agile/scrum (дата звернення: 17.11.2024).

13. Koster R. A Theory of Fun for Game Design. 2nd ed. Sebastopol: O'Reilly Media, 2013. 256 p.

14. McConnell S. Code Complete: A Practical Handbook of Software Construction. 2nd ed. Redmond: Microsoft Press, 2004. 960 p.

15. Newzoo. Global Games Market Report 2024. URL: https://newzoo.com/resources/trend-reports/newzoo-global-games-market-report-2024 (дата звернення: 05.11.2024).

16. Project Management Institute. A Guide to the Project Management Body of Knowledge (PMBOK® Guide). 7th ed. Pennsylvania: PMI, 2021. 367 p.

17. Rabin S. Introduction to Game Development. 3rd ed. Boston: Course Technology, 2019. 1024 p.

18. Rogers S. Level Up! The Guide to Great Video Game Design. 2nd ed. Chichester: Wiley, 2014. 496 p.

19. Salen K., Zimmerman E. Rules of Play: Game Design Fundamentals. Cambridge: MIT Press, 2003. 688 p.

20. Schell J. The Art of Game Design: A Book of Lenses. 3rd ed. Boca Raton: CRC Press, 2019. 600 p.

21. Scrum.org. What is Scrum? URL: https://www.scrum.org/resources/what-scrum-module (дата звернення: 16.11.2024).

22. Statista. Mobile Gaming Market Revenue Worldwide. URL: https://www.statista.com/statistics/292056/mobile-gaming-market-value-worldwide/ (дата звернення: 08.11.2024).

23. Sutherland J., Schwaber K. The Scrum Guide: The Definitive Guide to Scrum. 2020. URL: https://scrumguides.org/scrum-guide.html (дата звернення: 16.11.2024).

24. Trello. Trello Guide: How to Use Trello Like a Pro. URL: https://trello.com/guide (дата звернення: 19.11.2024).

25. Unity Technologies. Unity Documentation. URL: https://docs.unity3d.com/Manual/index.html (дата звернення: 10.11.2024).

26. Беспалова Т. Ю. Управління ІТ-проєктами: навчальний посібник. Харків: ХНЕУ ім. С. Кузнеця, 2020. 184 с.

27. Бушуєв С. Д., Бушуєва Н. С. Управління проектами та програмами: підручник. Київ: Самміт-Книга, 2019. 384 с.

28. Грицуленко С. І. Управління ІТ-проектами: навчальний посібник. Суми: Сумський державний університет, 2021. 264 с.

29. ДСТУ 8302:2015. Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання. Київ: ДП «УкрНДНЦ», 2016. 17 с.

30. Керівництво з основ проектного менеджменту (PMBOK® Guide) / пер. з англ. 6-те вид. Київ: ВІПОЛ, 2018. 756 с.

31. Кобилянський О. В., Ланде Д. В. Управління ІТ-проектами: підручник. Київ: КПІ ім. Ігоря Сікорського, 2020. 216 с.

32. Коляденко С. В. Цифровізація як основа розвитку ігрової індустрії. Вісник економіки транспорту і промисловості. 2021. № 73. С. 94–103.

33. Лукашенко О. А. Гейміфікація як інструмент підвищення мотивації користувачів мобільних застосунків. Науковий вісник Ужгородського університету. Серія: Економіка. 2022. Вип. 1 (59). С. 115–121.

34. Морозова О. М. Методології управління ІТ-проектами: порівняльний аналіз. Економіка та управління підприємствами. 2021. Вип. 42. С. 223–229.

35. Обленський М. О. Unreal Engine для розробки ігор: практичний посібник. Київ: Видавничий дім «Києво-Могилянська академія», 2023. 312 с.

36. Осипенко С. В. Сучасні тенденції розвитку мобільних ігор в Україні та світі. Інвестиції: практика та досвід. 2022. № 7. С. 51–56.

37. Ратушняк О. Г. Управління проектами в ІТ: навчальний посібник. Вінниця: ВНТУ, 2020. 156 с.

38. Романець І. О., Гриценко А. В. Оптимізація продуктивності мобільних застосунків на Unreal Engine. Системні дослідження та інформаційні технології. 2023. № 2. С. 88–97.

39. Чернов С. І., Титенко С. В. Agile-методології в управлінні ІТ-проектами: переваги та обмеження. Східна Європа: економіка, бізнес та управління. 2021. Вип. 3 (30). С. 366–371.