ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«УНІВЕРСИТЕТ ЕКОНОМІКИ ТА ПРАВА «КРОК»

КВАЛІФІКАЦІЙНА РОБОТА

Тема: «Комп’ютерна гра-скролер «Space Switch» з використанням data-oriented design»

Ступінь вищої освіти – бакалавр
Спеціальність – 122 «Комп’ютерні науки»
Освітня програма «Комп’ютерні науки»

ПОЯСНЮВАЛЬНА ЗАПИСКА

Виконав: здобувач 4 курсу
групи КН-21
Віталій БУГАЄНКО
Керівник: доцент кафедри комп’ютерних
наук, к.військ.н.
Володимир ТРОЦЬКО

Засвідчую, що кваліфікаційна робота оформлена відповідно до ДСТУ 3008:2015 та не містить запозичень з праць інших авторів без відповідних посилань.
Здобувач: _____
(підпис)

м. Київ – 2025 рік

ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД

«УНІВЕРСИТЕТ ЕКОНОМІКИ ТА ПРАВА «КРОК»

ЗАТВЕРДЖУЮ:
завідувач кафедри
комп'ютерних наук
_____Сергій МІЧКІВСЬКИЙ
«____»____20___р

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

Бугаєнко Віталій Олегович

Тема роботи	Комп'ютерна гра-скролер «Space Switch», з використанням data-oriented design
Номер та дата наказу про затвердження теми	№121-7 від 24 грудня 2024 року
Коротка постановка завдання	Розробити комп'ютерну гру на Unity з використанням архітектурної парадигми ECS
Посилання на джерела інформації (не більше п'яти найменувань, які рекомендує науковий керівник)	1. Unity: ECS Concepts // Unity Docs – URL: https://docs.unity3d.com/Packages/com.unity.entities@0.1/manual/ecs_core.html (дата звернення: 16.02.2025) 2. Marios Koutroumpas: A simple guide to get started with Unity ECS // Medium– URL: https://medium.com/gitconnected/a-simple-guide-to-get-started-with-unity-ecs-b0e6a036e707 (дата звернення: 16.02.2025) 3. MY.GAMES: Explaining (and making the leap) to ECS in Unity // Medium URL – https://medium.com/my-games-company/explaining-and-making-the-leap-to-ecs-in-unity-b6d786464d72 (дата звернення: 16.02.2025)
Вимоги до кваліфікаційної роботи	Кваліфікаційна робота має передбачити теоретичне, системотехнічне або експериментальне дослідження складного спеціалізованого завдання або практичної проблеми в галузі комп'ютерних наук, яке характеризується комплексністю та невизначеністю умов і потребує застосування теорій і методів інформаційних технологій.

Дата видачі завдання 27 грудня 2024 р.

Керівник Володимир ТРОЦЬКО

Здобувач освітнього ступеня бакалавра Віталій БУГАЄНКО

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання	Примітка
	Підготовчий етап		
1	Вибір напряму дослідження	02.12.2024 р.	*виконано*
2	Формування теми та призначення керівника	16.12.2024 р.	*виконано*
3	Затвердження теми кваліфікаційної роботи	23.12.2024 р.	*виконано*
4	Затвердження завдання на кваліфікаційну роботу	27.12.2024 р.	*виконано*
	Основний етап		
5	Розробка концепції кваліфікаційної роботи	13.01.2025 р.	*виконано*
6	Підбір та вивчення джерел інформації з напряму дослідження. Огляд існуючих аналогів	20.01.2025 р.	*виконано*
7	Затвердження розширеної постановки завдання. Підготовка та подання керівникові розділу 1 кваліфікаційної роботи	10.03.2025 р.	*виконано*
8	Проєктування. Підготовка та подання керівникові розділу 2 кваліфікаційної роботи	24.03.2025 р.	*виконано*
9	Підготовка доповіді для експертизи стану виконання кваліфікаційної роботи (проміжний контроль)	31.03-04.04.2025 р.	*виконано*
10	Реалізація. Підготовка та подання керівникові розділу 3 кваліфікаційної роботи	07.04.2025 р.	*виконано*
11	Підготовка та подання керівнику першого варіанту всієї кваліфікаційної роботи	14.04.2025 р.	*виконано*
12	Доопрацювання кваліфікаційної роботи з урахуванням зауважень керівника та представлення керівникові доопрацьованого варіанту кваліфікаційної роботи	21.04.2025 р.	*виконано*
	Завершальний етап		
13	Представлення рукопису для перевірки на плагіат	28.04-04.05.2025 р.	*виконано*
14	Підготовка презентації та доповіді на передзахист	05.05-11.05.2025 р.	*виконано*
15	Передзахист кваліфікаційної роботи	12.05-16.05.2025 р.	*виконано*
16	Доопрацювання роботи за результатами передзахисту	19.05-06.06.2025 р.	*виконано*
17	Експертиза роботи керівником та зовнішнім експертом	09.06-15.06.2025 р.	*виконано*
18	Доопрацювання доповіді та презентації для захисту	09.06-15.06.2025 р.	*виконано*
19	Захист кваліфікаційної роботи	16.06-22.06.2025 р.	*виконано*

Керівник Володимир ТРОЦЬКО

Здобувач освітнього ступеня бакалавра Віталій БУГАЄНКО

Бугаєнко В.О. Комп'ютерна гра-скролер «Space Switch» з використанням data-oriented design

Пояснювальна записка кваліфікаційної роботи за спеціальністю 122 – Комп'ютерні науки (освітня програма – Комп'ютерні науки) СО Бакалавр. – ВНЗ .Університет економіки та права .КРОК., Навчально-науковий інститут інформаційних та комунікаційних технологій, кафедра комп'ютерних наук, Київ, 2025.

У даній роботі описано розробку комп'ютерної гри на Unity з використанням архітектурної парадигми ECS

Ключові слова: Розробка, комп'ютерна гра, аркада, Unity, gamedev.

Табл. 2. Рисунок. 23. Бібліограф. 31.

Buhaienko V.O. Computer game-scroller "Space Switch" using data-oriented design (complex work)

Explanatory note of qualification work in specialty 122 - Computer Science (educational programme - Computer Science), Bachelor's degree - University of Economics and Law "KROK", Educational and Research Institute of Information and Communication Technologies, Department of Computer Science, Kyiv, 2023.

It is described the development of a computer game on Unity using the ECS architectural paradigm

Keywords: Development, computer game, arcade, Unity, gamedev.

Table 2. Figure. 23. Bibliography. 31

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

ААА-ігри – високобюджетні відеоігри, які створюються великими студіями з використанням значних ресурсів і сучасних технологій.

AI (Artificial Intelligence) – штучний інтелект, що використовується у відеоіграх для створення поведінки NPC, процедурної генерації та інших механік.

Data-Oriented Design (DOD) – підхід до програмування, що фокусується на ефективній обробці даних процесором, оптимізуючи розташування та доступ до даних у пам’яті.

ECS (Entity Component System) – data-oriented архітектурна парадигма, що використовується у розробці відеоігор для оптимального управління ігровими об'єктами, їх компонентами та системами.

GameDev (Game Development) – процес розробки відеоігор, що включає програмування, графічний дизайн, звукорежисуру, сценарну розробку, тестування та маркетинг.

NPC (Non-Player Character) – персонаж у відеогрі, що керується штучним інтелектом, а не гравцем.

OOP (Object-Oriented Programming) – об'єктно-орієнтоване програмування, методологія розробки програмного забезпечення, що базується на використанні об'єктів і класів.

Pay-To-Win (P2W) - модель монетизації відеоігор, у якій гравці можуть отримати значну перевагу над іншими шляхом купівлі внутрішньоігрових предметів або покращень за реальні гроші.

Prefab-структура – це заготовка об'єкта в Unity, яку можна багаторазово використовувати у проєкті. Вона зберігає налаштування, вигляд і поведінку елементів, що дозволяє швидко створювати однакові об'єкти без повторної ручної роботи.

Pipeline - це послідовність етапів обробки даних або виконання процесів у розробці відеоігор, що включає рендеринг, анімацію, обробку текстур та інші завдання.

Research - процес дослідження, аналізу та збору інформації для розробки та вдосконалення ігрових механік, штучного інтелекту та інших компонентів гри.

UI (User Interface) - це графічний інтерфейс користувача, який забезпечує взаємодію гравця з грою або програмою через візуальні елементи.

Аркадна гра – динамічна гра з простим керуванням і короткими ігровими сесіями, зазвичай орієнтована на досягнення високого рахунку. ААА-ігри – високобюджетні відеоігри, які створюються великими студіями з використанням значних ресурсів і сучасних технологій.

Геймплей – процес гри, тобто те, як саме користувач взаємодіє з механіками, світом і цілями гри.

Інпут – введення від гравця (натискання кнопок, рух мишки, дотики тощо).

Матчер – система або алгоритм, який підбирає об'єкти за певною логікою (наприклад, збіги у грі на відповідність).

Проджектайл – снаряд, який рухається у просторі після запуску (наприклад, куля чи ракета).

Скролер (жанр гри) – жанр відеоігор, де персонаж або середовище рухається у певному напрямку (горизонтально чи вертикально), створюючи ефект "прокручування" екрану.

Спавн – поява об'єкта у світі гри, зазвичай у заданій точці або за певних умов.

Фіча – окрема функція або можливість у грі чи програмі, що додає нову поведінку або контент.

ВСТУП

Актуальність теми. Розвиток комп'ютерних ігор є однією з найбільш динамічних сфер сучасних інформаційних технологій. Індустрія відеоігор постійно еволюціонує, використовуючи новітні підходи до розробки, що дозволяє створювати більш масштабні, продуктивні та гнучкі проєкти. Різноманітність жанрів сьогодні вражає і захоплює користувачів, даючи змогу насолодитися різноманітними продуктами ігрового ринку. Особливої популярность завжди мали аркадні ігри, які є доволі прості у освоєнні, проте цікаві навіть для сотого проходження. Піджанр таких ігр - скролери за рахунок динаміки та різноманіття залишаються у топах навіть на поточний день. Унікальні механіки, особливо поведінки ворогів привертають увгау тисяч гравців, створюючи унікальні геймплейні ситуації та можливості для вдосконалення майсетрності.

З роками, не дивлячись на те, що core геймплею залишався незмінним, підходи до його розробки змінювались дуже швидко. Одним із ключових аспектів сучасного ігрового розробництва є вибір архітектурної парадигми, яка визначає структуру коду, ефективність його виконання та можливості розширення проєкту. Інтрументарій для розробників має задовільняти низці умов, постійно ужосконалюючи швидкість розробки, її якість та спрощуючи експлуатаційний період, дозволячи легко впроваджувати нові механіки для задовільнення потреб користувачів. Для цього були винайдені цілі архітектури, які включають у себе правила та інструкції для представлення даних та взаємодій між ними.

Однією з інноваційних архітектур у розробці ігор є ECS – підхід, що сприяє модульності, ефективності та масштабованості програмного коду. Він відокремлює дані від логіки обробки, що дозволяє підвищити продуктивність гри, особливо у проєктах з великою кількістю об'єктів.

Гра Space Switch - є ідейним продовжувачем класичних скролерів, яка розширує їх механіки та можливості, особливо у сфері опису поведінки

ворогів за допомогою використання архітектурноої парадигми ECS. Впровадження такого підходу сприяє розширенню ігрових можливостей і створенню унікального користувацького досвіду.

Отже, дослідження та розробка комп'ютерної гри із використанням ECS є актуальними для галузі ігрової індустрії, оскільки цей підхід дозволяє створювати продуктивніші, масштабовані та гнучкіші ігрові системи. У межах кваліфікаційної роботи буде розглянуто процес розробки гри Space Switch, що використовує ECS, та її ключові механіки.

Мета дослідження. Метою проєкту є розробка програмного забезпечення комп'ютерної гри Space Switch, що використовує архітектурний підхід ECS для реалізації механік взаємодії між об'єктами.

Для досягнення мети виконано такі завдання:

- аналіз сучасних підходів до розробки відеоігор та архітектури ECS;
- визначення вимог до дизайну гри;
- проєктування структури гри та її дизайну;
- реалізація UI / UXза допомогою ECS у Unity;
- тестування та оптимізація продуктивності гри.

Об'єкт дослідження: написання гри-скролера за парадигмою ECS, вплив парадигми на процес та результат розробки.

Предмет дослідження: комп'ютерна гра Space Switch, побудована на базі архітектурного підходу ECS.

Методологічна основа роботи. Дослідження базується на використанні аналітичних, порівняльних і експериментальних методів. Аналітичний метод застосовано для дослідження існуючих архітектур відеоігор. Порівняльний аналіз дозволив оцінити ефективність ECS у порівнянні з іншими архітектурними підходами. Експериментальні методи використовувалися під час розробки та тестування гри.

Практичне значення. Результати роботи можуть бути використані для розробки високопродуктивних відеоігор із використанням ECS. Використання цієї архітектури дозволяє досягти високої продуктивності при обробці великої

кількості ігрових об’єктів, що робить її придатною для розробки як інді-проєктів, так і великих комерційних ігор.

Структура та обсяг пояснювальної записки. Кваліфікаційна робота складається зі вступу, трьох розділів, висновків та списку використаних джерел. Загальний обсяг пояснювальної записки становить 51 сторінці, з них основний зміст викладено на 40 сторінках. Робота містить 23 рисунка та 2 таблиці, що ілюструють ключові аспекти розробки гри.

РОЗДІЛ 1
ПОСТАНОВКА НА ЗАВДАННЯ КОМП'ЮТЕРНА ГРА "SPACE SWITCH" З ВИКОРИСТАННЯМ DATA-ORIENTED DESIGN

1.1 Опис предметної області

У сучасному світі ігри відіграють важливу роль у культурі та повсякденності більшості людей на Землі [1]. Індустрія розробки ігор виділяється як один із секторів, що розвиваються найшвидше, оскільки поєднує в собі багато аспектів роботи, таких як кодування, звуковий дизайн та сценарій, для створення цікавих і захопливих результатів.

За останнє десятиліття ігрова індустрія значно розширила своє охоплення аудиторії по всьому світу [2]. Мобільний геймінг став невід'ємною частиною буденності навіть людей похилого віку, що спонукає розробників та компанії створювати стратегії, які відповідають потребам різних груп користувачів. На тлі цього процесу ігрова індустрія постійно розвивається, намагаючись встигнути за напливом нових користувачів та утриманням старих юзерів.

Великий обсяг ринку спонукає розробників оптимізувати свої продукти, щоб залучити якомога ширшу базу користувачів. Аркадні ігри залишаються одним з найпоширеніших типів простих ігор на ринку (рис 1.1).

Аркадні ігри популярні завдяки своїй простоті, динаміці та миттєвому задоволенню від гри – не потрібно довго вчитуватись у правила, вчитися як у багатьох інших жанрах. А швидкий темп, азарт і притаманна епічність геймплею захоплює абсолютно всіх гравців, спонукаючи спробувати продукт. Вони ідеально підходять для коротких ігрових сесій, що добре поєднується з сучасним стрімким стилем життя багатьох людей.

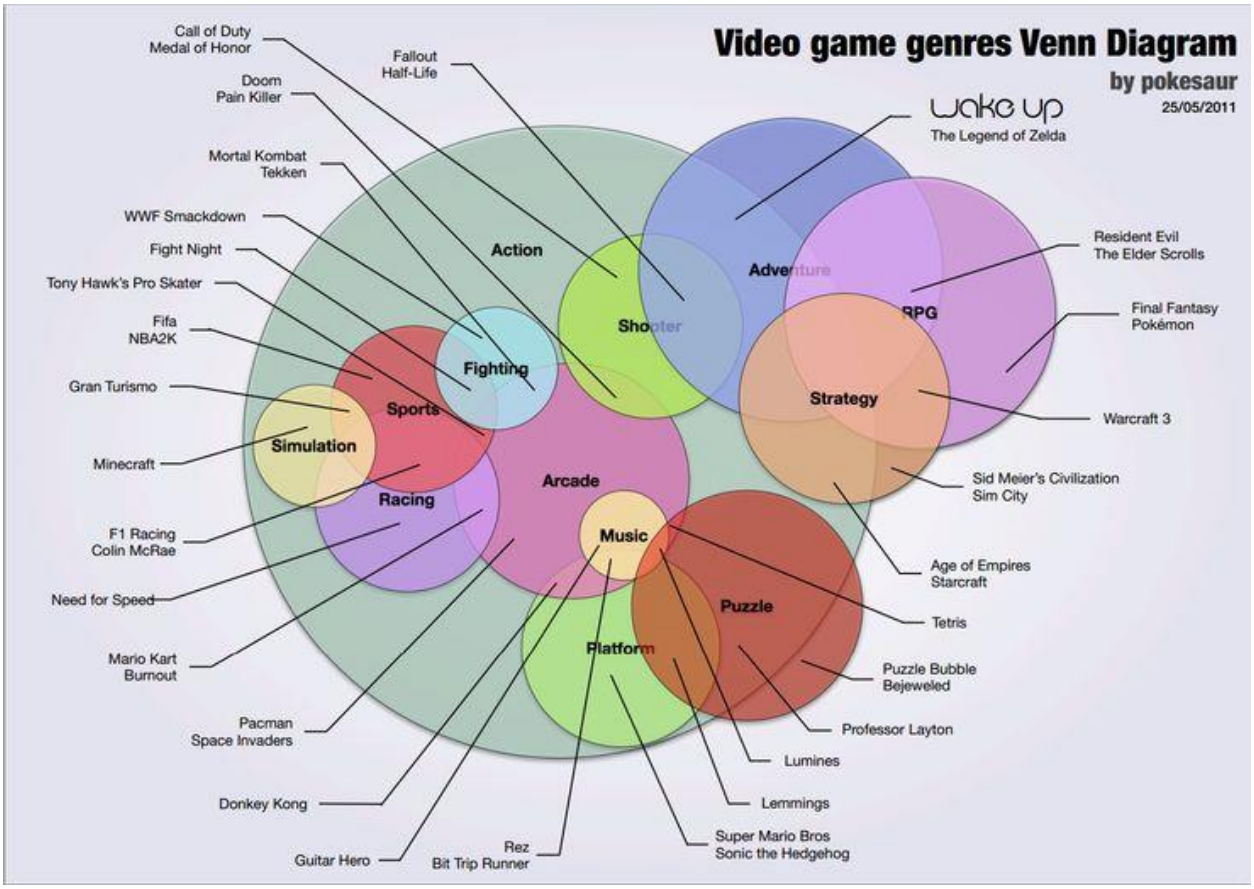

Рисунок 1.1 - Діаграма спектру жанрів ігрової індустрії

Джерело [3]

Проте, не дивлячись на простоту на перший погляд, аркадні ігри також потребують великих затрат у виробництві, проходячи великий цикл розробки гри:

- прототипування – створення базової механіки гри для тестування ігрового процесу;
- проектування гри – розробка концепції, написання документації, розробка геймдизайну;
- програмування – реалізація механік гри на рушії;
- створення графіки та звуку – розробка візуального стилю та музичного супроводу;
- тестування та оптимізація – пошук та виправлення багів, оптимізація продуктивності;

- реліз та підтримка – випуск гри, маркетинг та оновлення.

Під час розробки гри треба врахувати багато аспектів спільного pipeline, частини якого дозволяють процесу розробки гри мати комерційну ефективність (рис 1.2).

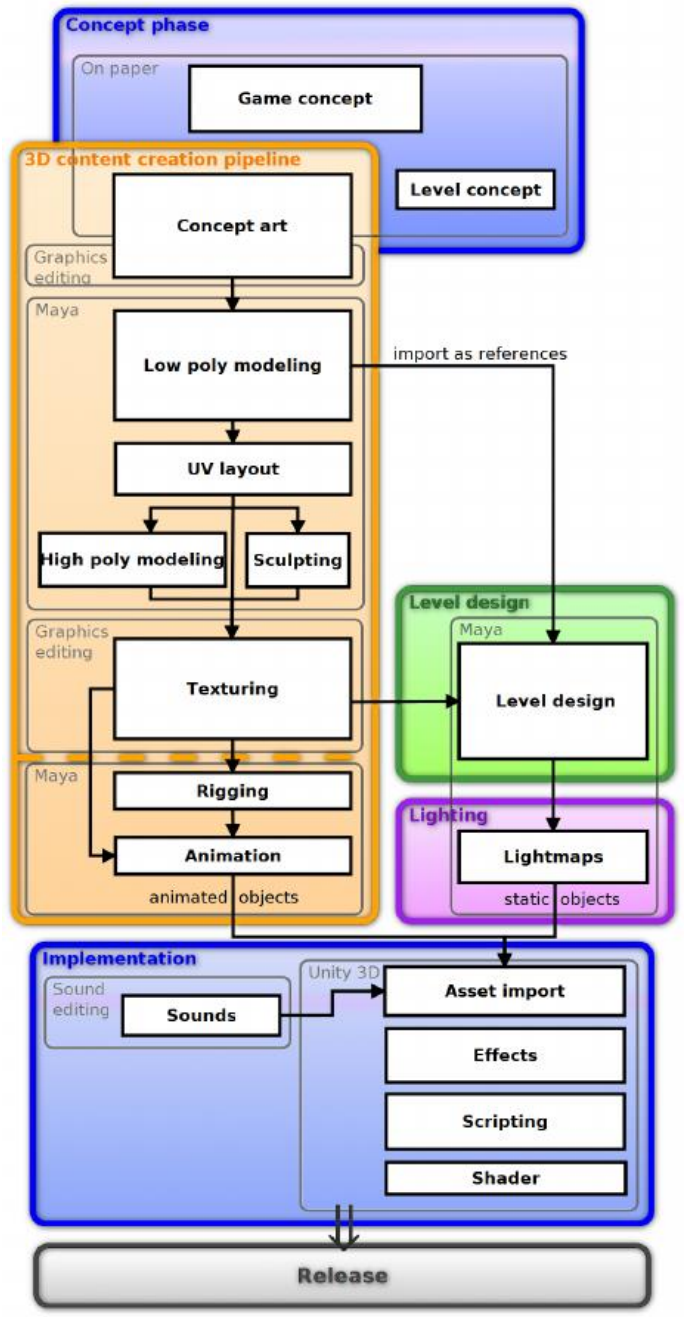

Рисунок 1.2 – Діаграма, яка іллюструє процеси розробки комерційної ігри

Джерело: [4]

У умовах жорсткого pipeline розробки ігор ефективне використання ресурсів та сприйняття даних від одного етапу розробки до іншого стає критично важливим, особливо коли мова йде про масштабованість, простоту підтримки та швидкість розробки проекту. Ключову роль у менеджменту даних грає етап написання коду продукту, адже ті дані, які зручні для сприйняття людиною не завжди є такими для машин, що спонукає шукати підходи до реалізації програмного забезпечення з урахуванням особливостей передачі та обробки даних саме для технічних приладів.

Для цього впродовж десятиліть були створені різні принципи дизайну програмного забезпечення, які дозволяють перенести інформацію з людської мови вимог на програмну мову виконання. Найпопулярнішим рішенням досі є наприклад ООП [5]. Він є популярним напрямом сприйняття бізнес логіки та даних для переносу їх у світ програмного коду, адже пропонує інтуїтивний, на перший погляд принцип сприйняття даних через абстракції об’єктів та взаємодій між ними. Цей підхід користується неабиякою популярністю та дозволяє вирішувати проблематику даних у багатьох сферах IT на сьогодні.

Проте сфера game dev має низку своїх особливостей. Предметна область, яка у багатьох інших сферах є як правило сталою, у контексті гри може змінюватись кардинально за малий проміжок часу. Наприклад, навряд чи під час розробки банківської системи потрібно буде міняти принципи захищеності або оперування з грошима клієнтів, адже це ключові та сталі вимоги. Розробка гри , не дивлячись на те, що є бізнес процесом спирається більше на ідею, або представлення якогось процесу, який на думку розробників та замовника дозволить задовільнити поставленим вимогам. Під час же розробки ігрового продукту, принципи реалізації цієї ідеї можуть змінюватися після ігрових тестів, звичайних демонстрацій, коли стає зрозуміло, що вони не дають бажаного результату.

У цьому контексті традиційні об’єктно-орієнтовані підходи, попри свою зручність у моделюванні поведінки об’єктів, часто не забезпечують необхідної

гнучкості в масштабуванні, адже вони базуються на детермінованості предметної області з чіткими інтерфейсами взаємодії між даними.

У випадках розробки ігор усе більшої популярності набуває Data-Oriented Design [6] – архітектурний підхід, що фокусується не на структурі об’єктів та їх взаємодії, а на ефективному зберіганні та обробці даних, як окремих повноцінних сутностей. Кожен елемент гри у DOD відображається як сукупність простих, декларативних компонентів, що містять лише дані, без поведінки У традиційних об’єктно-орієнтованих підходах зі зростанням коду зростає і складність відстеження залежності між об’єктами, що може призводити до хаотичної архітектури, де зміни в одному місці спричиняють неочікувані наслідки в іншому. Натомість DOD заохочує поділ логіки на чіткі, незалежні системи та компоненти, що робить структуру проекту прозорою, передбачуваною та простою в навігації. Такий підхід значно знижує ризик надмірної залежності між елементами коду, що спрощує тестування, масштабування і повторне використання компонентів у межах інших ігрових систем.

У контексті аркадних ігор, де одночасно можуть існувати десятки і навіть сотні активних об’єктів (вороги, кулі, ефекти, input-сигнали тощо) які поєднують різноманітні механіки від взаємодії один з одним, Data-Oriented Design дає змогу уникнути типових проблем проектування, зменшивши зв’язки між різними модулями та системами гри, надаючи гнучкість та легкість до експериментування, шляхом надання фундаменту побудови стійкої до змін архітектури.

Правильний підхід до побудови архітектури дозволяє розробникам ефективно керувати ресурсами, запобігати проблемам продуктивності та спрощувати процес оновлення гри. Для реалізації підходу DOD у архітектурі застосунку одним з найпопулярніших рішень є використання Entity–Component–System (ECS) [7] (рис 1.3).

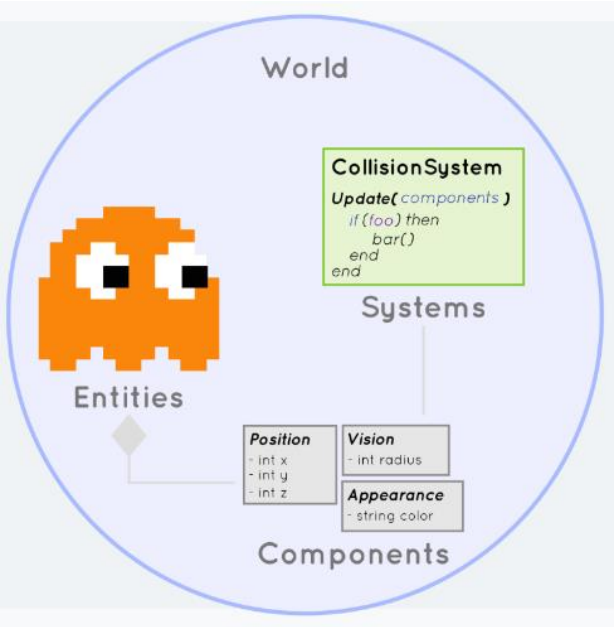

Рисунок 1.3 - Приклад сутності у світі ECS

Джерело [8]

ECS – це архітектурна парадигма, що забезпечує гнучке та продуктивне управління ігровими об'єктами. Її основна ідея полягає в розділенні даних і логіки: об'єкти (Entity) не містять поведінки, а лише служать контейнерами для компонентів (Component), які зберігають дані. Взаємодія між ними реалізується через системи (System), що виконують необхідні обчислення.

Важливою перевагою ECS у контексті дизайну гри є можливість створення модульних ігрових механік. Наприклад, замість того щоб створювати окремі класи для кожного типу персонажів чи предметів, у ECS можна комбінувати різні компоненти, отримуючи широкий спектр варіантів взаємодії. Такий підхід дозволяє дизайнерам створювати складні системи зі змінною структурою, що адаптуються під потреби геймплею.

На додаток, ECS сприяє оптимізації графічного відображення ігрових сцен. Завдяки розділенню логіки та даних, системи рендерингу можуть працювати незалежно від інших процесів гри, що покращує продуктивність. Це особливо актуально для динамічних ігор з великою кількістю активних об'єктів, таких як Space Switch, де важливо зберігати баланс між швидкістю оновлення кадрів та складністю графічного середовища.

Таким чином, застосування ECS у дизайні гри дозволяє не лише оптимізувати її архітектуру, а й значно розширити можливості творчого процесу, забезпечуючи гнучкість у створенні нових механік та адаптацію гри до змінних умов розробки.Оскільки системи працюють незалежно одна від одної, ECS дозволяє легко розпаралелювати обробку, що є критично важливим для продуктивних ігор. Завдяки чіткому розподілу ролей між Entity, Component і System, зменшується кількість залежностей між об'єктами, що спрощує їх тестування та налагодження.

Entity-Component-System є потужним архітектурною парадигмою, що дозволяє ефективно керувати великими ігровими проєктами. Завдяки гнучкості, оптимізації кешу та можливості паралельного виконання ECS стає стандартом у розробці продуктивних ігор.

Попри складність реалізації, цей підхід дає значні переваги у масштабуванні та підтримці ігор, що робить його перспективним вибором для сучасних розробників. З огляду на тенденції розвитку індустрії, ECS продовжить впроваджуватися в ігрові рушії та стане ще більш поширеним у майбутніх проектах [9].

1.2 Визначення потенційних конкурентних переваг

Жанр скролл–шутерів бере свій початок ще з аркадних автоматів 80–х років, коли такі ігри, як "Galaga" [10] та "R–Type" [11], заклали основи динамічного геймплею та інтенсивних перестрілок. Технологічний прогрес трансформував жанр скролл-шутера, інтегрувавши покращену графіку та функції рольових ігор разом з більш складною ігровою механікою, отримавши більш деталізовану графіку, глибші механіки та елементи RPG [12]. Ігри в цьому жанрі приваблюють численних гравців завдяки активному впровадженню систем модернізації кораблів, а також різноманітним рівням складності та кооперативним режимам гри.

У сучасному ігровому середовищі жанр скролл–шутерів залишається популярним серед гравців завдяки динамічному геймплею, різноманітності

ворогів та можливостям для вдосконалення ігрового процесу. Основними конкурентами проекту "Space Switch" є «Sky Force Reloaded» [13], «Razor2: Hidden Skies» [14] та «Hawk: Freedom Squadron» [15]. Ці ігри мають свої унікальні особливості та відмінності, які приваблюють гравців.

«Sky Force Reloaded»

Цей продукт відрізняється якісною графікою, плавною анімацією та добре збалансованою ігровою механікою. Основні переваги серії «Sky Force» включають:

- висока якість візуального оформлення;
- глибока система прокачування корабля RPG-like;
- відносно чесна монетизація – нема явних pay–to–win механік.
- режим кооперативної гри, який дозволяє проходити рівні разом з іншими.

Однак, серія «Sky Force» має певні недоліки, зокрема обмежену варіативність ворогів та відносно повільний темп додавання нового контенту (рис 1.4). Впродовж гри нові вороги з’являються нечасто – використовуються re-skin вже побачених раніше за механіками ворогів.

Razor2: Hidden Skies

Ця гра орієнтована на більш хардкорну аудиторію і пропонує класичний підхід до жанру shoot ‘em up. Її ключові особливості:

- висока складність гри, що робить її привабливою для любителів випробувань;
- технічна реалізація з акцентом на фізику польоту та поведінку ворогів;
- використання традиційного підходу до рівнів без надмірної казуалізації.

Головним недоліком гри є її обмежена привабливість для широкої аудиторії через високу складність, а також застарілий візуальний стиль (рис 1.5).

Рисунок 1.4 - Sky Force Reloaded скриншот геймплею. Кількісь унікальних ворогів впродовж гри стрімко падає

Джерело: [16]

Рисунок 1.5 - Razor2: Hidden Skies скриншот геймплею. Добре видно елементи застарілого стилю гри. В центрі скупчення ускладнених для візуального сприйняття елементів

Джерело: [14]

Hawk: Freedom Squadron

Ця гра має акцент на кооперативний режим та соціальну взаємодію між гравцями. Основні переваги:

- велика кількість рівнів та місій, які додають реіграбельності;
- мультиплеєрний режим, що залучає гравців до довготривалої гри;
- система щоденних завдань та подій, що підтримує інтерес гравців.

Разом з тим, "Hawk: Freedom Squadron" активно використовує pay-to-win механіки, що може негативно впливати на баланс гри та спричиняти дискомфорт у безкоштовних гравців (рис 1.6).

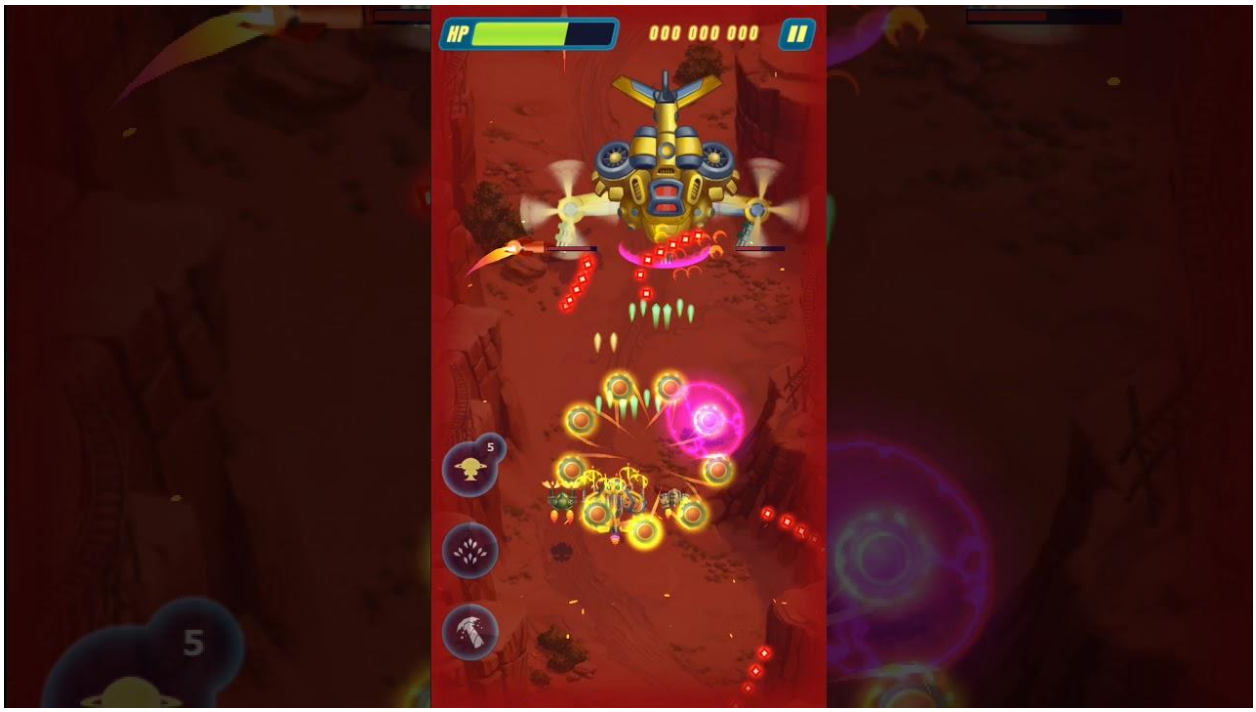

Рисунок 1.6 - Hawk: Freedom Squadron скриншот геймплею. Видно елементи UI, що відповідають за донатні здібності

Джерело: [15]

«Space Switch» матиме низку суттєвих переваг, які дозволять йому конкурувати з вищезгаданими іграми:

1) різноманіття ворогів та екшену;

- на відміну від більшості конкурентів, де зустрічається обмежений набір ворогів, “Space Switch” пропонує широке різноманіття супротивників із унікальними тактиками та механіками;
- Динамічні зміни бойових сценаріїв та комбінації ворогів додають непередбачуваності та збільшують реіграбельність;

2) швидкість впровадження нових геймплейних елементів;

- гнучка архітектура гри дозволяє легко та швидко додавати нові механіки, зброю, рівні та модифікатори;
- гравці матимуть можливість регулярно отримувати новий контент без тривалих очікувань, що сприятиме постійному інтересу до гри;

3) швидкість розробки;

- завдяки використанню сучасних інструментів розробки, проект має високу швидкість впровадження нових оновлень та поліпшень, що забезпечує оперативну реакцію на відгуки гравців;
- гнучкість у зміні механік дозволяє адаптуватися до побажань аудиторії швидше за конкурентів;

4) відсутність pay-to-win системи;

- на відміну від "Hawk: Freedom Squadron", де платний контент може створювати дисбаланс, “Space Switch” не матиме елементів, що дають несправедливу перевагу;
- основний акцент буде зроблено на чесний прогрес, заснований на навичках гравця та його стратегії.

З огляду на проведений аналіз, “Space Switch” має низку ключових переваг, які роблять його привабливим для гравців, особливо тих, хто цінує динамічний ігровий процес без донат-залежності. Завдяки різноманіттю ворогів, швидкому впровадженню нового контенту та справедливій монетизації, він може стати гідним конкурентом на ринку скролл-шутерів.

1.3 Постановка завдання на кваліфікаційну роботу

Завданням кваліфікаційної роботи є розробка програмного забезпечення для гри з використанням Data-Oriented Design на парадигмі ECS. Це програмне забезпечення пропонує користувачам унікальний досвід гри подібного жанру, а розробникам – pipeline, який дозволить швидко і ефективно впроваджувати бізнес логіку. Для реалізації бізнес ідей, необхідно виконати наступні задачі:

- провести проектування структури гри;
- провести research ассетів графіки, звуків тощо;
- провести моделювання геймплейних аспектів;
- реалізувати програмне забезпечення;
- провести тестування.

Для вирішення цієї задачі будуть реалізовані системи, які дозволять впровадити зазначені аспекти:

- система керування гравцем;
- система спавну ворогів;
- система пересування ворогів;
- зручний та інтуїтивний UI;
- система поведінки снарядів.

Усі зазначені системи будуть реалізовані або суто за допомогою інструментів проектування ECS та data-oriented design, або за допомогою комбінації цих інструментів з ООР.

Висновки до розділу 1

Проведено дослідження предметної області, представленої грою у жанрі аркадного скролера та процесом її розробки. Охарактеризовано підхід Data-Oriented Design його переваги для розробки ігор, та предмету роботи зокрема.

Аналіз існуючих конкурентів та їхніх особливостей виявив низку недоліків основних ігор–скролерів, які може компенсувати «Space Switch».

Також було досліджено pipeline конкурентів та виявлено їхні слабкі сторони, зокрема щодо швидкості прототипування. Для розробки проекту обрано альтернативний підхід – використання парадигми ECS.

На основі аналізу визначено ключові завдання проекту: реалізацію незадоволених потреб користувачів, які не врахували конкуренти, а також оптимізацію швидкості розробки. Окреслено основні етапи створення «Space Switch», складено задачі та визначено ключові аспекти розробки в контексті обраної парадигми.

РОЗДІЛ 2
ПРОЕКТУВАННЯ ДИЗАЙНУАРКАДНОЇ ГРИ НА ПАРАДИГМІ ECS

2.1 Проектування процесів

Проектування продукту має надати результат, який буде відповідати застосунку, що зможе задовільнити усі умови - функціональні та нефункціональні. Функціональні вимоги - це вимоги до продукту, основні та конкретні функції, що описують внутрішню структуру і роботу програми, які система повинна виконувати (обчислення даних, маніпулювання даними тощо). Для програмного забеспечення “Space Switch” до таких вимог належать:

- бізнес логіка описана за допомогою мови програмування правила поведінки об’єктів у грі, їх взаємодія та обробка первинних даних та даних, отриманих під час вза’ємодії цих об’єктів між собою;
- системна логіка описана високорівневими шлюзами взаємодії бізнес логіки із зовнішниіми даними, їх корекція\доповнення а також актуалізація за потреби;
- Beatuful Core - це основна частина візуальної презентації гри (particles, моделі, спрайти, шейдера тощо) які і задають художню стилістику грі;
- UI інтерфейс взаємодії користувача.

Нефункціональні вимоги - параметри, що не є безпосереднім функціоналом програми, але які впливають на якість, сприйняття користувачем та характеристики її роботи (безпека, швидкість тощо). Для програмного забеспечення “Space Switch” до таких вимог належать:

- Open Source – це код програмного забеспечення має бути розміщений на платформі для вільного ознайомлення користувачами та контрибуції за бажанням [17];

- дані статичного характеру (які не змінюються під час сеансу гри – конфіги балансу, ворогів, гравця тощо) мають мати можливість знаходитись у хмарному сховищі для майбутнього можливого мастабування;
- гра має працювати стабільно, проходячи увесь цикл користування не минаючи основних етапів. Кількість багів має прагнути до 0;
- модулі програмного забеспечення повинні мати обширні можливості для відного легкого,дешевого та стрімкого масштабування (кількість ворогів, рівнів тощо);
- гра має стабільно працювати на більшості конфігурації ПК на 2025 рік. Мають бути відсутні стрімки просадки у ФПС, неоптимізований батчинг, несжаті текстури.

Для задовільнення усіх вимог, проектування структури “Space Switch” базується на стандартій моделі керування процесами для усіх ігор жанру скроллер. Центральними елементами гри є сутності гравця та ворогів, які представлені космічними кораблями, взаємодія між якими і визначає геймплей та виконання ігрових інтструкцій (рис. 2.1).

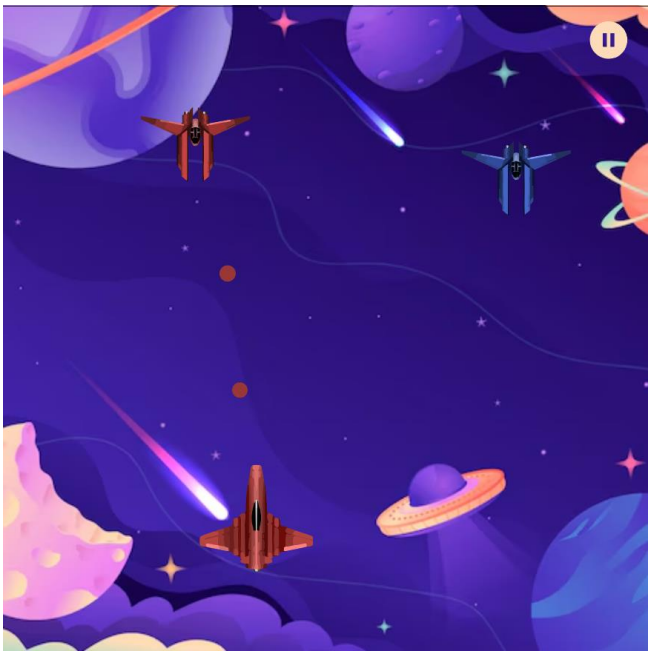

Рисунок 2.1 -Дизайн низькодеталізованого прототипу гри

Джерело: розроблено автором

Гравець, як ключова фігура та актор будує навколо себе ядро взаємодії частин програми (система очок, рух камери, game loop [18] тощо). Вороги відокремлені актори, які діють за своїми стратегіями та правилами, які лише опосередковано залежать від дій гравця, проте впливають на нього безпосередньо, що і визначають основні ігрові прецеденти геймплею , описані на діаграмі (рис. 2.2). Це є основні події, які можуть відбутися та відбуваються у певні моменти між акторами за певних умов..

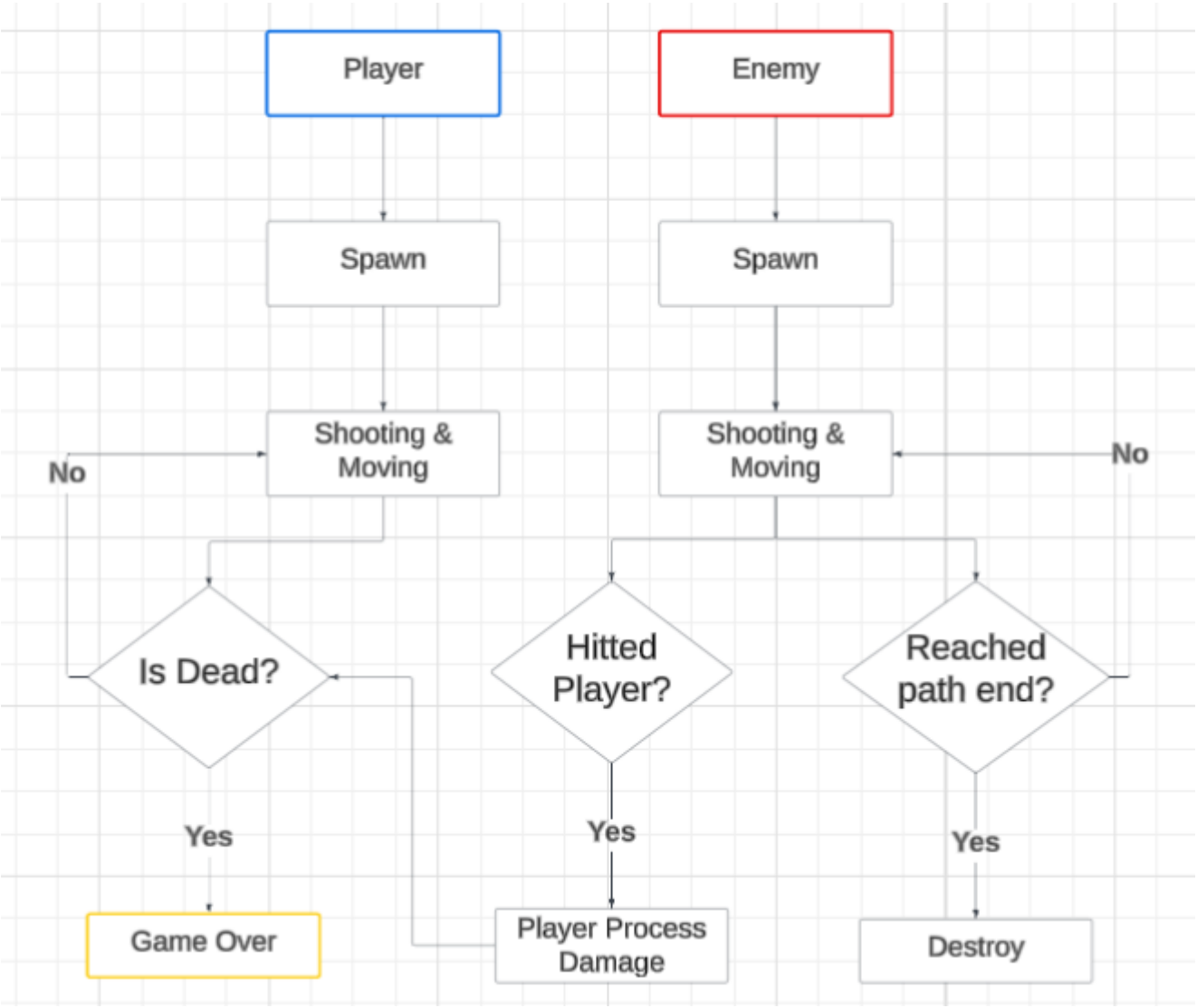

Рисунок 2.2 - Діаграма прецедентів

Джерело: розроблено автором

На діаграмі зображені основні етапи взаємодії юзера у застосунку, можливі дії та наслідки дій гравця. Гравець розпочинає гру та отримує можливість керувати персонажем, стріляти по ворогах, а також зазнавати атак. Його шлях завершується або перемогою, або смертю при втраті всього здоров'я. Вороги, своєю чергою, з'являються в грі через механізм спавну, після чого виконують запрограмовані дії, такі як рух за скриптом та атаки. Їхня участь у грі закінчується, коли вони досягають кінцевої точки маршруту, гинуть від пострілів гравця або знищують самого гравця.

Діаграма (рис. 2.3) ілюструє взаємодію об'єктів у хронологічному порядку.

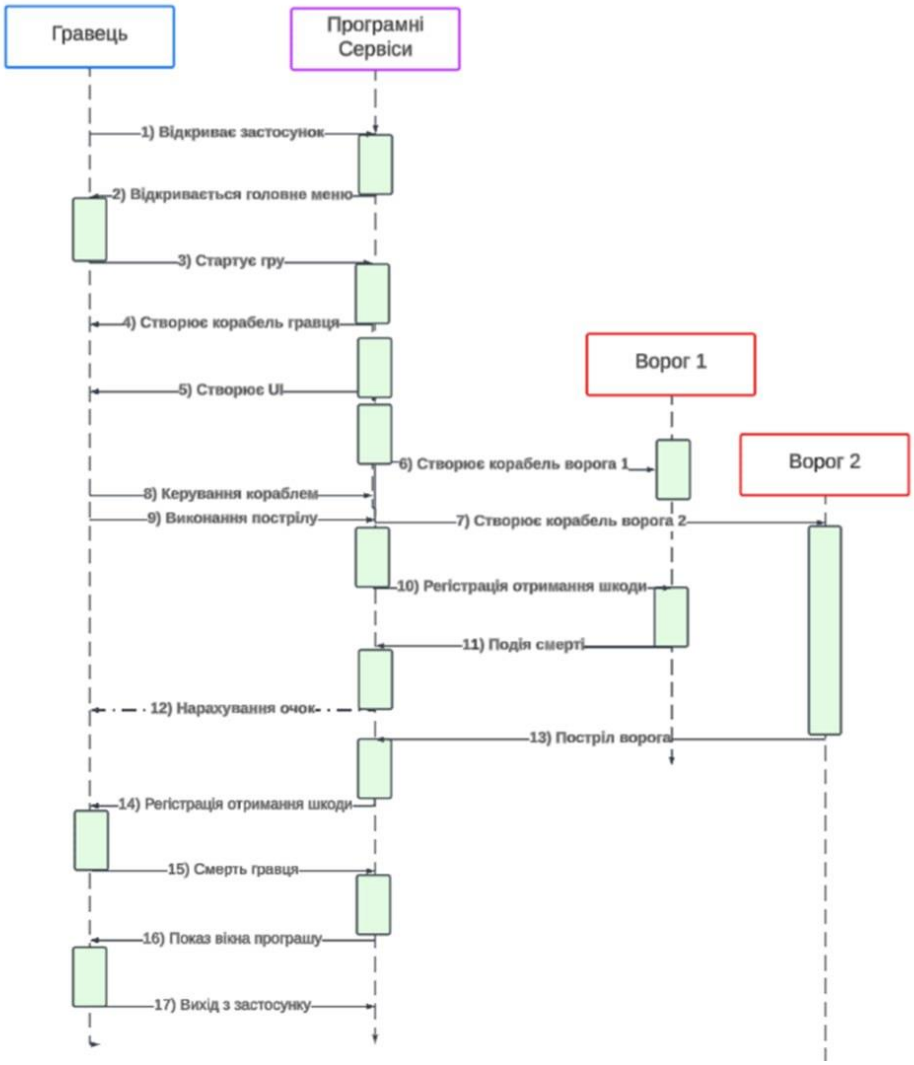

Рисунок 2.3 - Діаграма послідовності

Джерело: розроблено автором

Діаграма послідовності демонструє основний цикл гри гравця (game loop) та його залежність від ворога. Починаються їх цикли паралельно, де обидві сутності рухаються та стріляють, ворог через описану для нього логіку, гравець – через input. У момент виконання своїх інструкцій може трапитись ситуація, коли ворог попадає по гравцеві. Цей кейс має бути перевірений гравцем у системі здоров’я, та у разі, якщо гравець маркується як мертвий – гра закінчується. Інакше – гравець та ворог продовжують свої цикли. Ворог, в свою чергу, у разі закінчення виконання руху по детермінованому шляху буде знищений, якщо його не знищив гравець.

2.2 Моделювання поведінки дизайну

У процесі проектувані ігор, особливо аркадних скролерів, моделювання даних є не лише технічним завданням, а й дизайнерським викликом. Необхідно не просто зберігати стан об'єктів, а забезпечувати таку структуру, яка дозволить гнучко змінювати ігрову логіку, стилістику та візуальне представлення об'єктів без постійного втручання у вже реалізовані частини гри.

У традиційному ООП об’єкти містять як дані, так і поведінку. Це зручно для невеликих, статичних систем, однак в умовах геймдизайну та UI-дизайну, де постійно змінюються вимоги до вигляду, анімацій або візуальних станів, такий підхід створює зайву жорсткість.

На відміну від ООП, архітектурна парадигма ECS, що лежить в основі дата орієнтовного підходу, дозволяє розділити візуальні, функціональні та поведінкові аспекти сутностей (рис 2.4). Кожна сутність у грі – це набір компонентів, які можуть легко оновлюватися або переініціалізовуватись залежно від ігрового стану або стилістичних рішень.

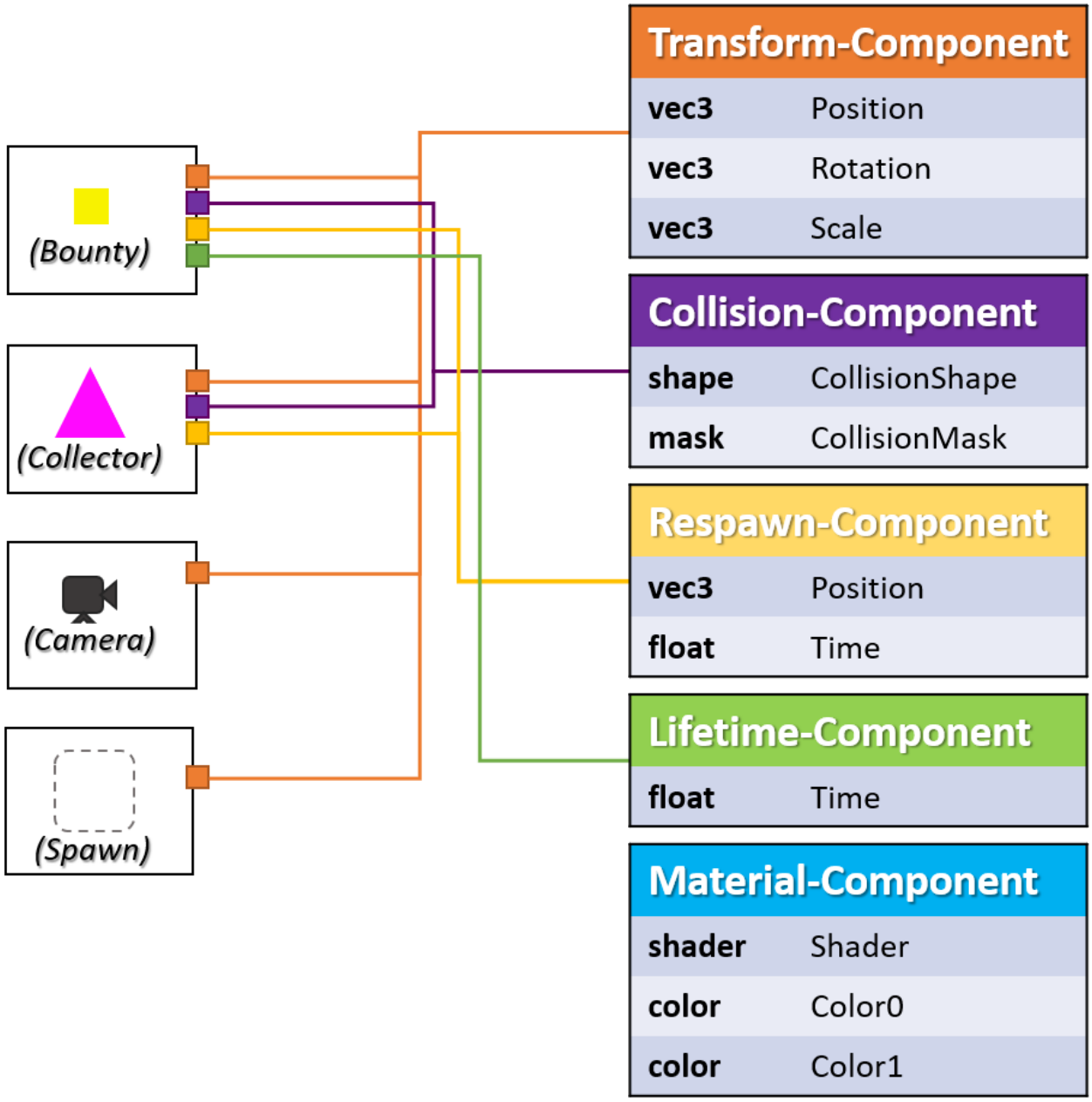

Рисунок 2.4 - Узагальнена структура ECS-моделі у контексті візуального дизайну

Джерело: [19]

У контексті дизайну це означає:

- гнучкість у зміні вигляду: стилізацію об’єкта можна змінити просто підмінивши компонент візуального оформлення;
- адаптивність UI: система може самостійно активувати або деактивувати певні інтерфейсні елементи на основі подій гри;

- легкість тестування візуальних сценаріїв: дизайнери можуть комбінувати компоненти для перевірки різних стилістичних рішень без втручання в основну логіку гри.

Це забезпечує не лише ефективне використання ресурсів, але й створює умови для творчої роботи дизайнера, який може зосередитися на побудові естетичного, зручного ігрового середовища без постійної залежності від технічних обмежень. Саме тому ECS розглядається не лише як інженерне, а і як дизайнерське рішення, яке відкриває нові можливості для створення інтерфейсів та ігрових сцен, що адаптуються під різні сценарії використання.

Оскільки вона ізольована та базується на своїх даних, то результати її роботи може використовувати будь який юзер – гравець, ворог, окремий проджектайл тощо, без необхідності проектування прямих, або абстрактних зв’язків.

2.3 Проектування інтерфейсу

У динамічних аркадних іграх типу скрол-шутерів інтерфейс користувача виконує не лише інформаційну, але й функціональну роль. Через високу швидкість подій, велику кількість рухомих об’єктів і візуальну насиченість сцени, надмірна кількість елементів інтерфейсу або їх неузгоджене оформлення здатні значно ускладнити сприйняття та знизити якість ігрового досвіду.

Досвід інших проєктів у цьому жанрі дозволяє простежити низку поширених проблем [20]. У грі Razor2: Hidden Skies використовується класичний інтерфейс із великою кількістю статичних елементів, що постійно присутні на екрані (рис. 2.5). Таке рішення не враховує контекст ігрових подій і створює ситуацію, коли важлива інформація може губитися серед другорядних візуальних об’єктів. Крім того, графічний стиль UI у Razor2 містить застарілі декоративні рішення – рамки, текстури з низькою прозорістю, дрібні шрифти – що ускладнює сприйняття.

Рисунок 2.5 - Гра Razor2: Hidden Skies в якій використовується застарілий інтерфейс із великою кількістю статичних елементів

Джерело: [14]

У випадку з Hawk: Freedom Squadron інша проблема, хоча інтерфейс візуально яскравіший, він перевантажений елементами, що стосуються внутрішньоігрових покупок, щоденних завдань і подій (рис. 2.6). У результаті увага гравця розсіюється між геймплейними та сервісними блоками, що є критичним недоліком у грі, де темп змін подій дуже високий.

На фоні попередніх кокнурентів лише Sky Force Reloaded демонструює краще рішення – помірну мінімалізацію елементів інтерфейсу (рис. 2.7). Більшість інформації з’являється лише в моменти, коли вона справді необхідна. У таких проєктах візуальна мова інтерфейсу побудована навколо простих іконок, мінімальної кількості кольорів та ефектів прозорості, що дозволяє інтегрувати UI в загальну стилістику гри, не конфліктуючи з нею.

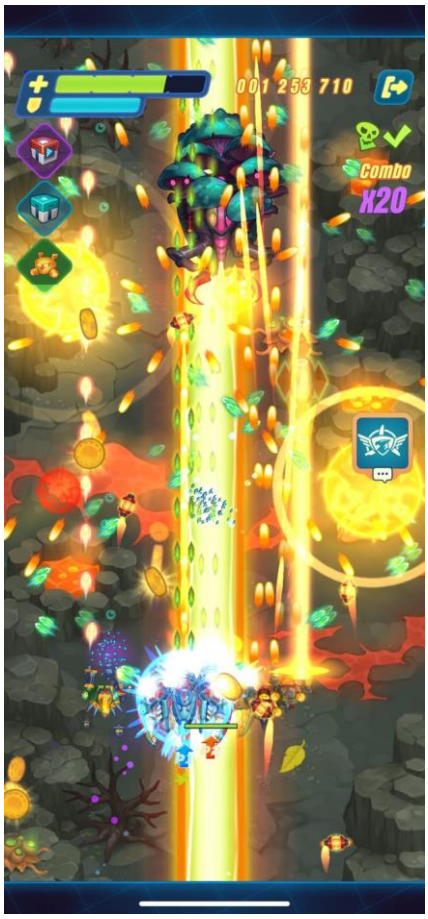

Рисунок 2.6 – Інтерфейс гри Hawk: Freedom Squadron з перевантаженими елементами

Джерело: [15]

Рисунок 2.7 – Гра Sky Force Reloaded з помірною мінімалізацію елементів інтерфейсу

Джерело: [16]

З урахуванням вищезгаданого можна стверджувати, що для гри Space Switch, орієнтованої на активний бойовий процес з високою щільністю подій, найбільш доцільним є підхід до інтерфейсу, в основі якого лежить мінімалізм, контекстність і візуальна чистота.

Щодо стилістики, за рахунок аналізу стало зрозуміло про ефективність стриманого футуристичного оформлення інтерфейсу – з використанням прозорих панелей, неонових підсвіток, простих форм та обмеженої палітри. Такий підхід не лише узгоджується з космічною тематикою, але й забезпечує візуальний контраст із насиченим фоном ігрових сцен.

Крім зовнішнього вигляду, важливою є поведінка інтерфейсу під час гри. Застосування простих анімацій – поступове зникнення та поява, реакція на події (наприклад, спалахи при отриманні ушкоджень або досягненні мети) – сприяє кращому інформуванню без зайвих слів (рис. 2.8). Важливо, щоб такі ефекти виникали лише у потрібний момент, були короткими й не блокували інші дії. Їхня роль – підсилення відчуття контролю, передачі емоційної реакції гри на дії гравця .

Рисунок 2.8 –Анімації вибухів, пострілів і отримання очок у грі Sky Force Reloaded

Джерело: [16]

Інтерфейс Space Switch формується з урахуванням цих принципів: візуальна простота, контекстне відображення інформації, узгодженість зі стилістикою, ефективне використання простору та обмеженої кількості ефектів. Такий підхід дозволятиме не лише зберігати чистоту екрану, а й підтримувати загальну динаміку геймплею.

2.4 Візуалізація ігрових процесів у дизайні

Процеси у грі не будуть обмежуватися програмною логікою – вони мають бути візуально зрозумілими для гравця. Завдання дизайну полягає в тому, щоб зробити ці процеси не лише естетично приємними, але й функціонально прозорими. Для цього застосовуються діаграми активності, інтерактивні візуальні сигнали та структурні шаблони взаємодії.

На діаграмі варіантів використання (рис. 2.9) зображено основні дії, які може виконувати користувач з його точки зору у межах гри Space Switch, а також взаємозв’язки між цими діями. Гравцеві, який ініціює взаємодію із системою, доступні такі варіанти: запуск гри, ініціалізація, зміна параметрів, керування ігровим процесом, зупинка гри, її перезапуск, а також перегляд рекордних результатів. Центральним сценарієм є дія «Грати гру», яка включає можливість зупинки (наприклад, через паузу) та пов’язана із завершенням гри. Такі дії, як ініціалізація та зміна параметрів, є частиною загального підготовчого процесу перед основним ігровим циклом.

На початку ігрової сесії користувач буде взаємодіяти з головним меню – ключовим елементом, який повинен бути простим, але характерним. Вибір між "почати гру" чи "вийти" має бути миттєво зрозумілим, без перевантаження другорядними опціями. Це рішення підтримується відповідними візуальними маркерами: розміщенням, контрастом і анімацією наведення. Після запуску гри поступово будуть завантажуватися всі необхідні елементи: фон, гравець, інтерфейс та супротивники. Ця послідовність дозволить уникнути візуального перевантаження та формує правильне фокусування уваги. Гравець отримує чіткий контекст того, що відбувається на екрані.

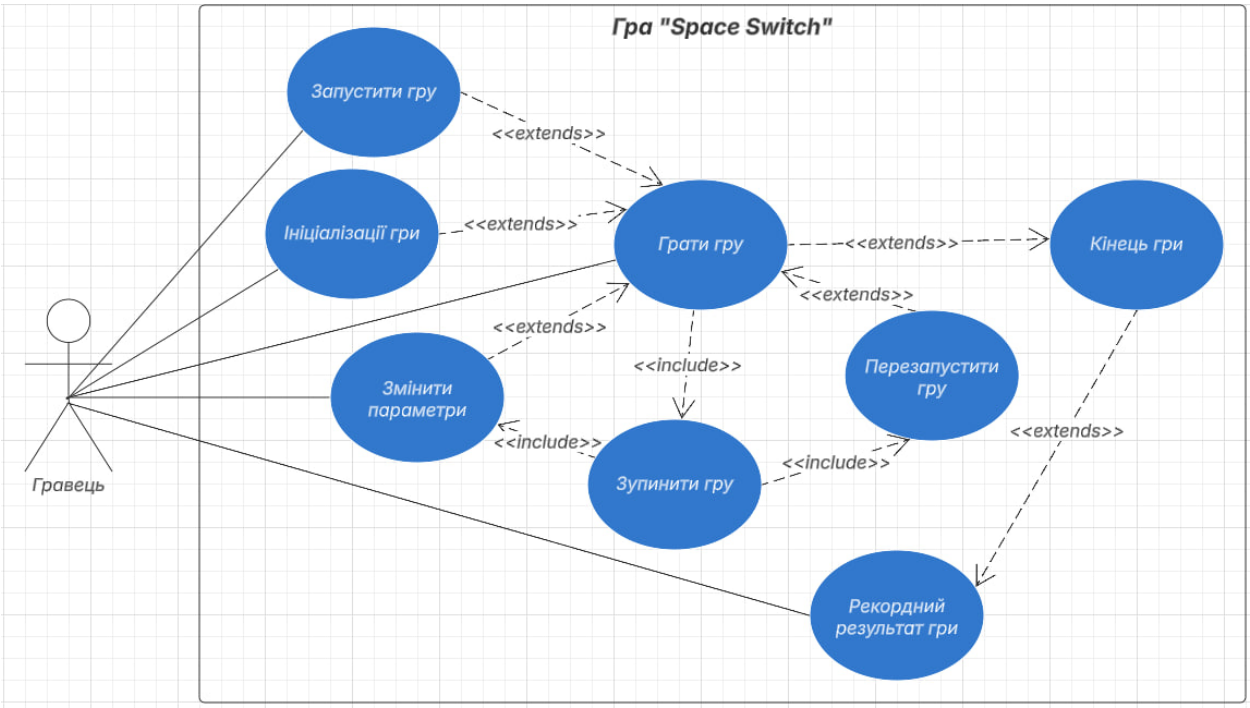

Рисунок 2.9 – Діаграма варіантів використання

Джерело: розроблено автором

Протягом ігрової сесії важливо зберігати візуальну логіку процесів (табл. 2.1). Отримання ушкодження буде супроводжуватися спалахами, тремтінням екрана чи зміною кольору інтерфейсних елементів.

Таблиця 2.1 - Візуальні ефекти у відповідь на ключові події геймплею

Подія	*Візуальний ефект*	*Мета*
Отримання шкоди	Спалах, червоний фрейм	Попередження про ризик
Завершення рівня	Екран-затемнення	Перехідний акцент
Наведення на кнопку	Зміна кольору, анімація	Візуальний фідбек

Це дозволяє швидко зчитати важливу інформацію без додаткових текстових підказок. Такі сигнали мають бути помітними, але не відволікати від геймплею.У випадку смерті гравця буде з’являєтись зрозумілий інтерфейс

із вибором: перезапуск або вихід у головне меню. Тут ключовим залишається принцип простоти й чіткості. Ефекти не мають бути нав'язливими, а логіка переходу – максимально короткою. Вся увага фокусується на рішенні гравця без зайвих подразників.

Таким чином, дизайн візуальних процесів у грі – це більше ніж просто "красиві анімації". Це продумана система реакцій, ритмів і переходів, яка допомагає гравцеві інтуїтивно орієнтуватися у грі. Саме цей підхід дозволяє зберігати високу динаміку аркадного жанру, не втрачаючи зручності й зрозумілості (рис. 2.10).

Рисунок 2.10 – Мапа подорожі клієнта для гри «Space Switch»

Джерело: розроблено автором

Висновки до розділу 2

На етапі проектування було позначено основні вимоги та ключові діючи актори гри. У ході моделювання даних був обраний оптимальний архітектурний для опису акторів та взаємодії динамічних даних. Був обраний мінімалістичний підхід до опису інтерфейсу гри та використані перевірені рішення для структуризації елементів UI.

Запропоновані рішення можуть дозволити забезпечити ефективну роботу застосунку, його масштабованість та зручність використання. Проєктування інтерфейсу враховувало потреби користувачів, що сприятиме покращенню взаємодії з програмою. Таким чином, на базі здійсненого проектування можна побудувати програмний продукт, який задовільнить усім поставленим завданням розділу І.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ ДИЗАЙНУ ГРИ НА DATA-ORIENTED DESIGN

3.1 Прототипування дизайну продукту

Інтерфейс гри створювався безпосередньо у середовищі Unity. Це дозволило поєднати дизайн і логіку взаємодії ще на ранніх етапах розробки. Такий підхід скорочує розрив між дизайнерським задумом і технічною реалізацією, мінімізує повторну роботу та дає змогу одразу бачити результат у реальному ігровому середовищі.

Розробка велась з орієнтацією на MVP. Всі ключові екрани – головне меню, ігрова сцена, пауза, завершення гри – були реалізовані з інтерактивними елементами та базовою логікою навігації. Особлива увага приділялась ролі інтерфейсу як посередника між гравцем і грою: кнопки, повідомлення, підказки повинні були інформувати, не відволікаючи.

Під час побудови структури інтерфейсу було застосовано гексагональну (шестикутну) сітку (рис 3.1) як елемент композиції – це підсилює візуальне враження sci-fi середовища [21], уникає прямолінійних сіток, додає глибини візуальній мові гри. Такі структури використовувались для декоративних елементів у меню, підкладок для текстів, а також як фонові візерунки у HUD.

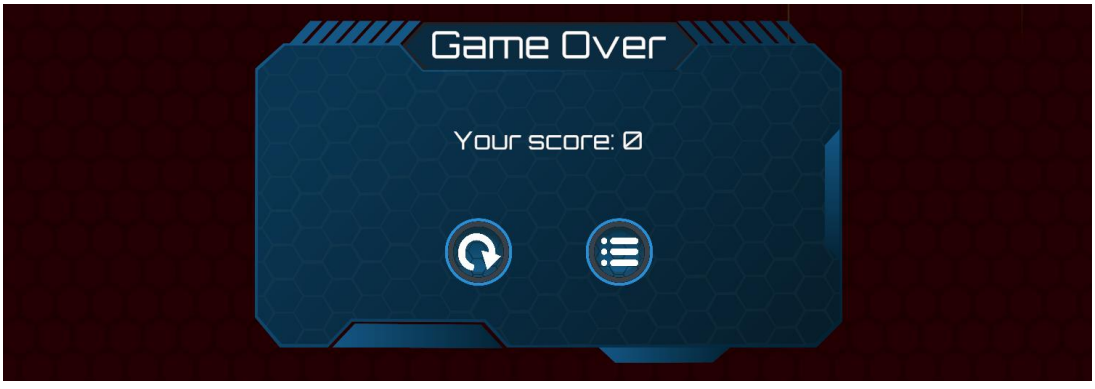

Рисунок 3.1 – Використання гексагональної (шестикутної) сітки в інтерфейсі для підсилює візуальне враження sci-fi середовища

Джерело: розроблено автором

Для заголовків і основних текстових елементів були обрані шрифти Neuropol X (рис. 3.2) і Orbitron, що мають геометричну форму та добре читаються в динаміці. Вони підсилюють футуристичний стиль і при цьому не створюють зайвого візуального шуму [22].

Рисунок 3.2 – Шрифт Neuropol X що мають геометричну форму та добре читаються в динаміці

Джерело: [23]

Навігація розроблялась з урахуванням мінімальної кількості кліків та передбачуваності дій. Гравець завжди має лише два ключові варіанти дій: продовжити або вийти. Розміщення кнопок – за класичною ігровою схемою: центрування важливого, акцент на клавішу підтвердження, деакцент на другорядні опції. Кожен стан гри супроводжується чітким візуальним оформленням – без анімаційних перевантажень, але з відчуттям завершеності дії.

Усі кнопки, спливаючі повідомлення та ефекти реакції змодельовані прямо в середовищі розробки. Це зробило інтерфейс інтерактивний, що дало змогу з самого початку бачити, як користувач взаємодіє з системою, і швидко вносити зміни без затримок між дизайном і реалізацією.

Всі елементи UI розроблялись з урахуванням масштабованості, адаптації під різні роздільності, і відображення у повноекранному режимі (рис 3.3). Були протестовані на коректність відображення при зміні налаштувань графіки та масштабів екрану.

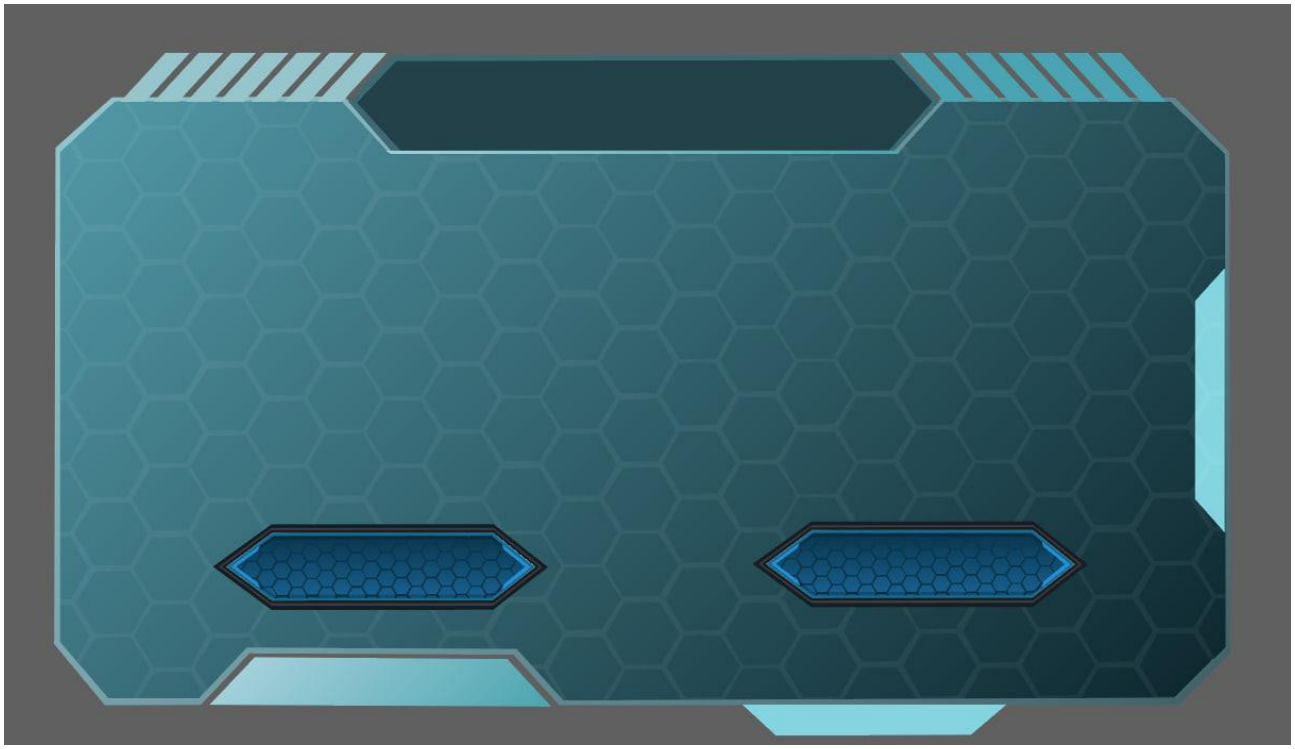

Рисунок 3.3 – Панель паузи, який адаптований під різну роздільность екрану

Джерело: розроблено автором

Таким чином, замість умовної "макетної" стадії було реалізовано робочий функціональний прототип, що дозволяє не тільки тестувати дизайн-рішення, а й створювати гру одночасно з її оформленням. Це відповідає

принципу мінімально життєздатного продукту і значно пришвидшує виробничий цикл.

3.2 Тестування дизайну продукту

Інтерфейс тестувався на зручність, візуальну зрозумілість, реакцію на події та відповідність очікуванням користувача. Було набрано групу тестувальників з 6 людей віком від 15 до 29 років.

Проведено guerrilla-тестування [24] – короткі сеанси гри з коментарями користувачів під час процесу. Більшість тестувальників інтуїтивно знаходили основні функції, однак було виявлено кілька вразливих моментів. Зокрема, кнопка паузи на початковій версії була занадто дрібною та зливалася з фоном. Після тесту її було збільшено, змінено відтінок і додано легку анімацію наведення (рис 3.4).

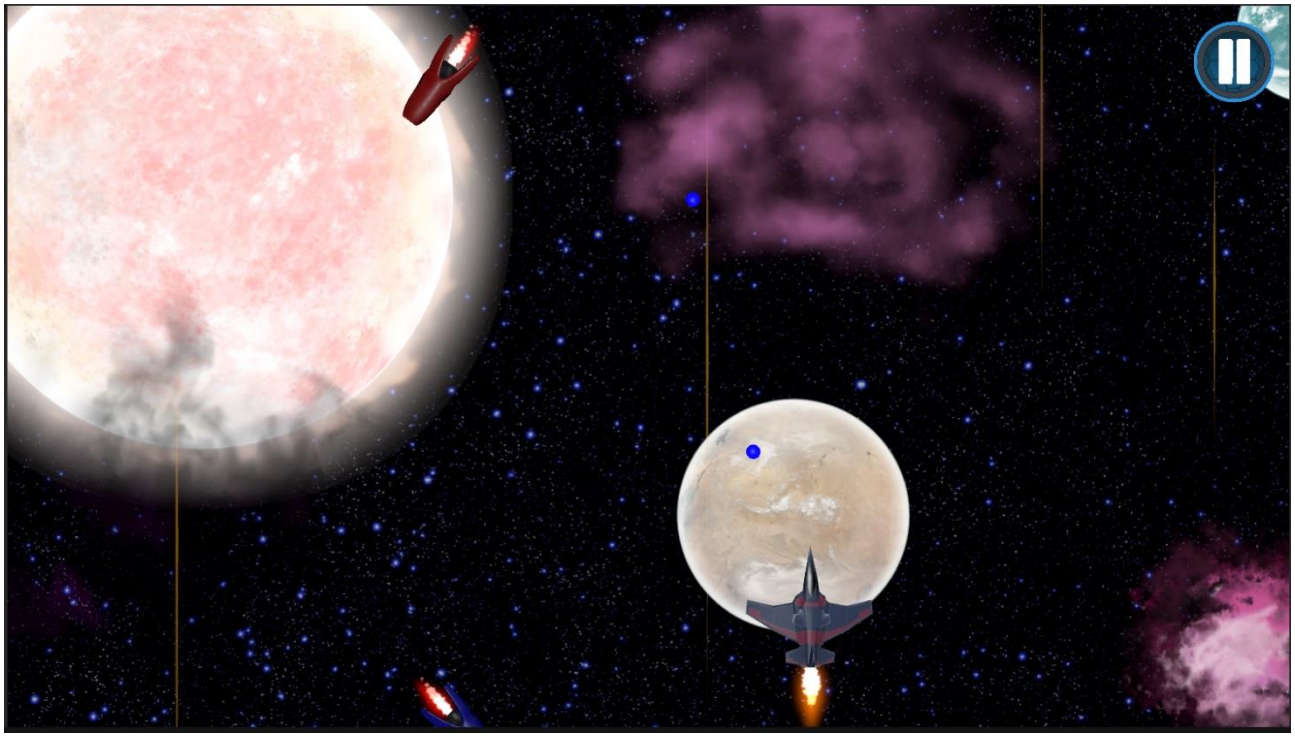

Рисунок 3.4 – Виправлений інтерфейс в якому збільшили кнопку паузи

Джерело: розроблено автором

Крім цього, інтерфейс перевірявся на відповідність основним евристикам Нільсена [25]. Це набір загальноприйнятих принципів, які допомагають виявити проблеми у дизайні інтерфейсу навіть без участі кінцевих користувачів. До них, зокрема, належать: видимість стану системи, відповідність між системою і реальним світом, контроль користувача, запобігання помилкам, стандарти, гнучкість і ефективність використання, естетика й мінімалізм, а також надання ефективної допомоги. Група тестувальників виявила такі невідповідності в межах проєкту:

- недостатній візуальний фідбек при натисканні на деякі кнопки;
- відсутність підтвердження після виходу з гри.

Всі виявлені помилки були зафіксовані у табл. 3.1 і після кожного циклу тестування проходила стадія корекції. Ось приклад фрагменту такого списку:

Таблиця 3.1 - Всі виявлені помилки та їх рішення

Проблема	*Рішення*
Кнопка "Пауза" не помітна	Зміна розміру, додано обведення
Вихід без підтвердження	Додано діалогове вікно
Відсутній фідбек на натискання	Додано анімацію масштабування та звук

Після внесення змін було проведено повторне тестування з тим самим пулом тестувальників. Всі базові сценарії пройдено успішно без уточнюючих питань з боку тестера. Але деякі зауважили щодо колірної палітри панелі паузи та запропонували підібрати більш вдалий колір який гармнійно виглядав з кнопками (рис. 3.5).

Рисунок 3.5 – Панель паузи після зміни кольору який гармонійно виглядає разом з кнопками

Джерело: розроблено автором

Перевірка доступності (WCAG 2.1)

Окрему увагу приділено відповідності інтерфейсу принципам WCAG 2.1 [26] – міжнародному стандарту, що регламентує доступність цифрового контенту. Було враховано:

- достатній контраст між фоном і текстом (не менше 4.5:1 для звичайного тексту);
- мінімальний розмір інтерактивних елементів (не менше 44 пікселів);
- наявність зорового фокусу на активних елементах при взаємодії;
- передбачуваність навігації та узгодженість поведінки інтерфейсу.

Усі ці критерії дозволяють покращити досвід не тільки для користувачів з особливими потребами, а й для широкої аудиторії гравців заголом (рис 3.6)..

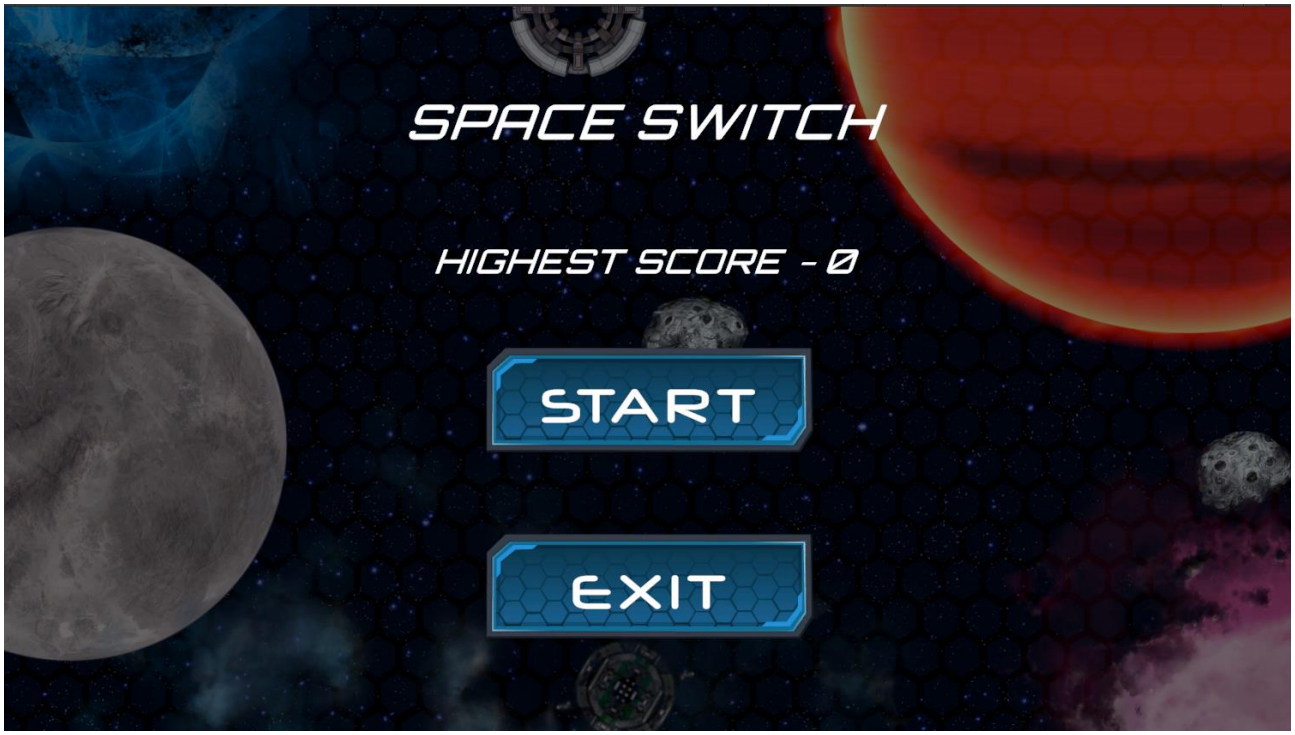

Рисунок 3.6 – Інтерфейс меню з урахуванням принципів WCAG 2.1

Джерело: розроблено автором

Usability-тестування

Usability-тестування [27] проводилось у вигляді короткого сценарію: запуск гри, натискання кнопки паузи, взаємодія з екраном завершення, повернення до головного меню. Тестурів просили виконати ці дії без підказок, а потім – оцінити зручність за 5-бальною шкалою.

Результати показали, що більшість тестувальників оцінюють інтерфейс як інтуїтивно зрозумілий (рис 3.7). Середній час на виконання кожної задачі становив менше 5 секунд, а коефіцієнт помилкових натискань – нижче 10%. Це свідчить про правильну логіку розміщення елементів і відповідність очікуванням користувача.

Рисунок 3.7 – Панель паузи з інтуїтивно зрозумілими кнопками

Джерело: розроблено автором

3.3 Підготовка до передачі готової роботи

Після завершення основної фази проєктування почався етап підготовки інтерфейсної системи до передачі у фінальну збірку гри. Цей процес включав як технічні, так і візуальні уточнення, покликані забезпечити стабільність роботи, узгодженість структури та готовність до інтеграції в кодову базу проєкту.

Усі UI-елементи були згруповані за функціональними блоками: інформаційна панель (HUD), головне меню, спливаючі повідомлення, фінальні екрани тощо. Для кожного з блоків була сформована окрема prefab-структура з внутрішньою ієрархією. Усі об’єкти отримали логічні назви згідно зі стандартами ігрової архітектури, що спростило навігацію всередині сцени та пришвидшило подальшу інтеграцію.

Особливу увагу приділено уніфікації стилістики: усі кнопки, іконки, фрейми були приведені до єдиного розміру, стилю обводок та шрифтів. Це

дозволило уникнути візуальних суперечностей [28] і сформувати цілісний інтерфейс. Окремі компоненти було перевірено на відповідність геймплейним сценаріям – наприклад, HUD мав коректно оновлюватися під час бою, а повідомлення – не перекривати важливі ігрові зони.

Графічні файли експортувались у форматах PNG (рис. 3.8) (для фонових і декоративних елементів) і SVG (для масштабованих іконок), що гарантує чіткість відображення на різних роздільностях екранів. UI реалізовано через Canvas з поділом на логічні секції та адаптоване до роботи у FullHD-режимі без втрати читабельності чи пропорційності елементів.

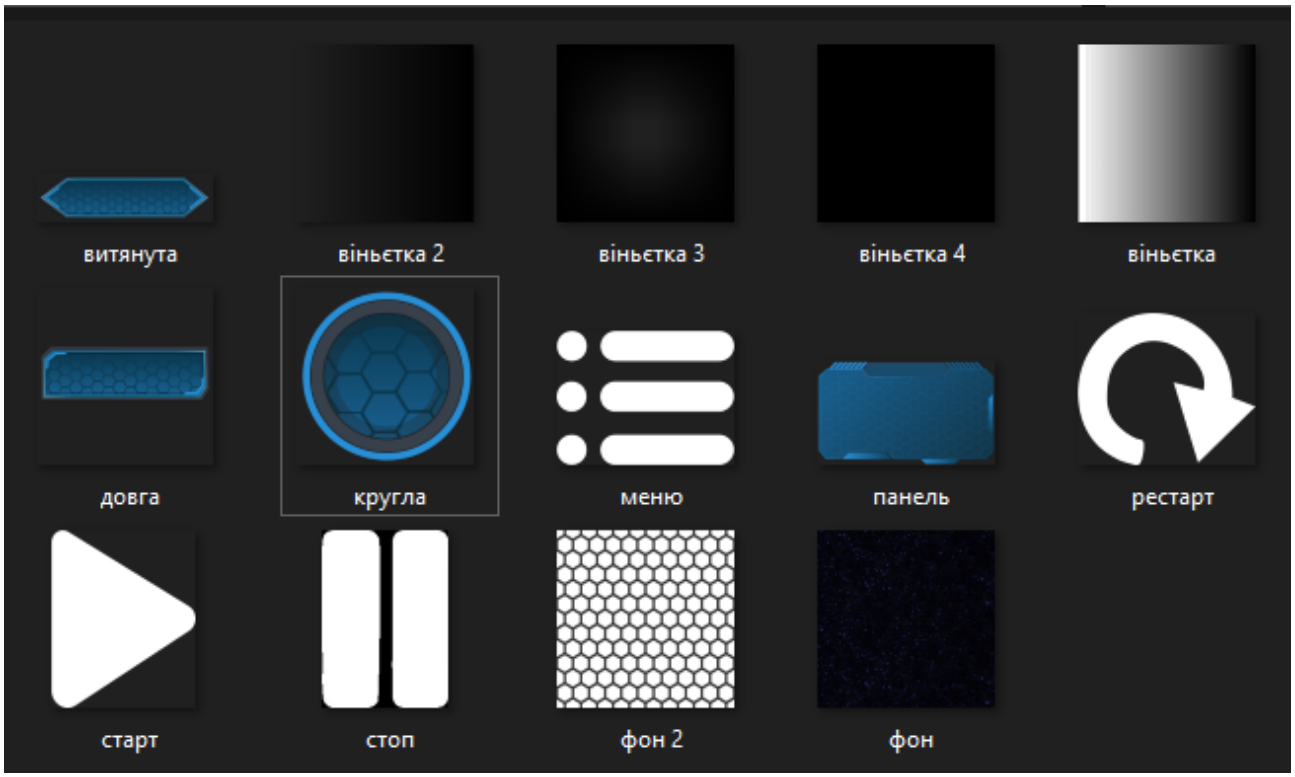

Рисунок 3.8 – Папка зі спрайтами з логічними назвами в форматі PNG

Джерело розроблено автором

Кожен елемент інтерфейсу був структурований у відповідній групі – це забезпечує зручність подальшого супроводу та пришвидшує процес технічної інтеграції. Наприклад, всі елементи HUD були винесені в окрему підсистему з власним контролером, який відповідає за оновлення значень у реальному часі.

Подібний підхід дозволяє уникнути дублювання коду, а також легко масштабувати інтерфейс у разі додавання нових функцій або режимів гри [29].

Окрему увагу було приділено логіці завантаження ресурсів. Статичні елементи (наприклад, фонові зображення або неінтерактивні декоративні об’єкти) кешувалися під час першого запуску, тоді як динамічні (підказки, повідомлення про події, спливаючі вікна) підвантажувалися за потреби через систему Addressables [30]. Це дозволило зменшити час першого запуску гри та підвищити загальну стабільність проєкту [31].

Висновок до розділу 3

На етапі реалізації «Space Switch» було продемонстровано процес створення інтерфейсної системи з акцентом на підготовці дизайну до інтеграції у програмне середовище.

Було описано процес створення візуальних елементів, їх структурування за функціональними блоками, адаптацію під вимоги геймплею та тестування на відповідність принципам доступності та зручності. Продемонстровано приклади інтеграції екранів у гру та описано етапи підготовки інтерфейсної системи до передачі у фінальну збірку.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи було розроблено гру-скролер «Space Switch» з використанням data-oriented design. У ході розробки було проведено аналіз конкурентів та пошук можливостей для задоволення потреб користувачів та всіх вимог, висунутих у ході аналізу.

Під час етапу проектування було, запроваджено використання парадигми ECS та розбити основну частину програми на окремі модулі, що дозволило спланувати стійку до змін архітектуру застосунку.

У межах реалізації інтерфейсу було спроєктовано та протестовано повноцінну систему взаємодії з користувачем, що охоплює всі основні екрани гри. Працюючи безпосередньо в Unity, було створено MVP-прототип з акцентом на візуальну чіткість, швидкість сприйняття та відповідність sci-fi стилю. Особливу увагу приділено типографіці, кольоровій ієрархії, логіці навігації та мікровзаємодіям. Було проведено два види тестування та впроваджено зміни на основі фідбеку від тестувальників. Це дозволило сформувати інтерфейс, що не лише підтримує геймплей, а й підсилює емоційне сприйняття гри.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Herman Narula – A billion new players are set to transform the gaming industry. URL: https://www.wired.com/story/worldwide-gamers-billionplayers (дата звернення: 14.04.2025).
2. James Batchelor – Video games to pass $300bn revenue, 3.8 billion players by 2030 // Games Industry – Midia Research URL: https://www.gamesindustry.biz/midia-research-video-games-to-pass-300bn-revenue-38-billion-players-by-2030 (дата звернення: 19.04.2025).
3. Pokesaur – Game Theory: Video Game Genres Venn Diagram // WordPress URL: https://pokesaur.wordpress.com/2011/05/25/game-theory-video-game-genres-venn-diagram/ (дата звернення: 19.04.2025).
4. ResearchGate – Game production pipeline overview. URL: https://www.researchgate.net/figure/Game-production-pipeline-overview-Concept-content-creation-pipeline-level-design_fig2_267417785 (дата звернення: 19.04.2025).
5. Nataliya Revutska – What is OOP // SoftServe URL: https://career.softserveinc.com/en-us/stories/what-is-object-oriented-programming-oop-explaining-four-major-principles (дата звернення: 19.04.2025).
6. Arpanext – A Comprehensive Guide To Data-Oriented Design For Improved Software Efficiency // Medium URL: https://arpanext.medium.com/a-comprehensive-guide-to-data-oriented-design-for-improved-software-efficiency-6434d520d0e4 (дата звернення: 19.04.2025).
7. Leo – Why is Entity Component System (ECS) so awesome for game development? // Medium URL: https://medium.com/source-true/why-is-entity-component-system-ecs-so-awesome-for-game-development-f554e1367c17 (дата звернення: 19.04.2025).

8. Nidorx – Tiny and easy to use ECS (Entity Component System) library for game programming // GitHub URL: https://github.com/nidorx/ecs-lib (дата звернення: 19.04.2025).
9. Denis Kondratev – Exploring Unity DOTS and ECS: Is it a Game Changer? // Hackernoon URL: https://hackernoon.com/exploring-unity-dots-and-ecs-is-it-a-game-changer (дата звернення: 19.04.2025).
10. Wikipedia – Galaga. URL: https://uk.wikipedia.org/wiki/Galaga (дата звернення: 19.04.2025).
11. Wikipedia – R-Type. URL: https://uk.wikipedia.org/wiki/R-Type (дата звернення: 19.04.2025).
12. Techtarget – role-playing game (RPG). URL: https://www.techtarget.com/whatis/definition/role-playing-game-RPG#:~:text=A%20role%2Dplaying%20game%20(RPG)%20is%20a%20game%20in,BattleTech%2C%20and%20Star%20Wars%20Galaxies. (дата звернення: 19.04.2025).
13. Wikipedia – Sky Force. URL: https://en.wikipedia.org/wiki/Sky_Force (дата звернення: 19.04.2025).
14. Steam – Razor2: Hidden Skies. URL: https://store.steampowered.com/app/34920/Razor2_Hidden_Skies/?l=ukrainian (дата звернення: 19.04.2025).
15. Google Play – Hawk: Freedom Squadron. URL: https://play.google.com/store/apps/details?id=com.my.hawk.air.shooter (дата звернення: 19.04.2025).
16. Steam – Sky Force Reloaded. URL: https://store.steampowered.com/app/667600/Sky_Force_Reloaded/ (дата звернення: 19.04.2025).
17. Oleksiy Pletnov – Open source: що це, для чого і як розпочати // DOU URL: https://dou.ua/lenta/articles/open-source-reasons-to-join/ (дата звернення: 19.04.2025).

18. M.R.M Abdullah – Intro to Game Dev - Understanding the GAME LOOP // Medium URL: https://m-abdullah-ramees0916.medium.com/the-game-loop-f6f5cb68c00 (дата звернення: 19.04.2025).
19. Moises Gamio – Understanding OOP concepts // Codersite URL: https://codersite.dev/understanding-oop-concepts/ (дата звернення: 19.04.2025).
20. Мітрофаненко М. О. – Використання архітектури ECS у розробці ігор в середовищі Unity // eprints.library.odeku.edu.ua. URL: http://eprints.library.odeku.edu.ua/id/eprint/12855/1/Mitrofanenko_M_2023.pdf (дата звернення: 19.04.2025).
21. Steam – Ігри з використанням шестикутної сітки (Hex Grid) // Steam. URL: https://store.steampowered.com/tags/fr/Hex%20Grid/9/?l=ukrainian (дата звернення: 19.04.2025).
22. Fonts Online – Шрифт Neuropol X // fonts-online.ru. URL: https://fonts-online.ru/fonts/neuropol-x (дата звернення: 19.04.2025).
23. Font Meme – Шрифт Orbitron // fontmeme.com. URL: https://fontmeme.com/fonts/orbitron-font/ (дата звернення: 19.04.2025).
24. QATestLab – Автоматизація тестування ігор: навіщо це потрібно та як працює // career.qatestlab.eu. URL: https://career.qatestlab.eu/ua/blog/game-test-automation/ (дата звернення: 19.04.2025).
25. Goldweb Solutions – Експертна оцінка юзабіліті вашого сайту та 10 евристик Якоба Нільсена // goldwebsolutions.com. URL: https://goldwebsolutions.com/uk/blog/ekspertna-otsinka-yuzabiliti-vashogo-sajtu-ta-10-evristik-yakoba-nilsena/ (дата звернення: 19.04.2025).
26. W3C – Керівництво з доступності веб-контенту (WCAG) 2.1, українська версія // w3.org. URL: https://www.w3.org/Translations/WCAG21-ua/ (дата звернення: 19.04.2025).
27. Usability.gov – Usability Testing Basics // usability.gov. URL: https://www.usability.gov/how-to-and-tools/methods/usability-testing.html (дата звернення: 19.04.2025).

28. Avada Media – Розробка дизайну інтерфейсу гри // Avada Media. URL: https://avada-media.ua/ua/blog/razrabotka-dizayna-interfeysa-igry/ (дата звернення: 29.04.2025).
29. Збірник IT та MC 2023 // eprints.zu.edu.ua. URL: http://eprints.zu.edu.ua/37557/1/%D0%97%D0%B1%D1%96%D1%80%D0%BD%D0%B8%D0%BA_%D0%86%D0%A2%D1%82%D0%B0%D0%9C%D0%A1_2023_%D0%B4%D1%80%D1%83%D0%BA.pdf (дата звернення: 29.04.2025).
30. Unity – Addressables. URL: https://docs.unity3d.com/Manual/com.unity.addressables.html (дата звернення: 19.04.2025).
31. Віталій Гикавчук – Розробка прототипа ігрового рушія: проблеми, виклики та прийняті рішення // DOU. URL: https://gamedev.dou.ua/blogs/how-i-create-game-engine-prototype/ (дата звернення: 29.04.2025).