Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *Бігми Ростислава Юрійовича*
(ПІБ)

академічної групи *121-19ск-1*
(шифр)

спеціальності *121 Інженерія програмного забезпечення*
(код і назва спеціальності)

освітньої програми *Інженерія програмного забезпечення*
(назва освітньої програми)

на тему: Розробка програмного забезпечення для функціонування інтерактивної гри «BossFight» з використанням фреймворка Unity 3D та C#

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	проф. Бердник М.Г.			
розділів:				
спеціальний				
економічний	*доц. Касьяненко Л.В.*			
Рецензент	*доц.Шедловський І.А.*			
Нормоконтролер	*доц. Гуліна І.Г.*			

Дніпро
2022

Міністерство освіти і науки України
НТУ «Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
завідувач кафедри
програмного забезпечення комп'ютерних систем
(повна назва)

_____ І.М. Удовик
(підпис) (прізвище, ініціали)

« » _____ 2022 року

ЗАВДАННЯ
на кваліфікаційну роботу
бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *121-19ск-1* *Бігми Ростислава Юрійовича*
(група) (прізвище та ініціали)

тема кваліфікаційної роботи Розробка програмного забезпечення для функціонування інтерактивної гри «BossFight» з використанням фреймворка Unity 3D та C#

затверджена наказом ректора НТУ «ДП» від *18 травня 2022р.* № 268-с

Розділ	Зміст виконання	Термін виконання
Спеціальний	***На основі матеріалів проєктно-технологічної та інших науково-технічних джерел провести аналіз стану рішення проблеми та постановку задачі.*** ***Обґрунтувати вибір та здійснити реалізацію методів вирішення проблеми***	***13.05.2022 р.***
Економічний	***Провести розрахунок трудомісткості розробки програмного забезпечення, витрат на створення ПЗ й тривалості його розробки***	***27.05.2022 р.***

Завдання видав _____ проф. Бердник М.Г.
(підпис) (посада, прізвище, ініціали)

Завдання прийняв до виконання _____ *Бігма Р.Ю.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *14.01.2022 р.*

Термін подання кваліфікаційної роботи до ЕК: *13.06.2022 р.*

РЕФЕРАТ

Пояснювальна записка: 54 с., 6 рис., 1 табл., 3 дод., 25 джерел.

Об'єкт розробки: ігровий програмний додаток «BossFight» з використанням Unity3D та C#.

Мета кваліфікаційної роботи: розробка програмного забезпечення для функціонування інтерактивної гри «BossFight» з використанням фреймворка Unity 3D та C#

У вступі виконується аналіз сучасного стану проблеми, уточнюється постановка завдання, мета кваліфікаційної роботи та галузь її застосування, обґрунтовується актуальність теми.

У першому розділі проводиться дослідження предметної галузі та існуючих рішень, визначається актуальність завдання та призначення розробки, уточнюється постановка завдання.

У другому розділі обирається платформа для розробки, виконується проектування програми і її розробка, наводиться опис алгоритму і структури функціонування системи, визначаються вхідні і вихідні дані, наводяться характеристики складу параметрів технічних засобів, описується робота програми.

В економічному розділі визначається трудомісткість розробленого програмного продукту, проводиться підрахунок вартості роботи пр створенню застосунку та розраховується час на його створення.

Практичне значення полягає у створенні прототипу мобільної гри, в якій користувачеві дається можливість спробувати свої сили у поєдинках з різними супротивниками.

Актуальність інформаційної системи визначається великим попитом на подібні ігри і зростаючою статистикою їх популярності, починаючи з 2016 року.

Список ключових слів: ПРОТОТИП, ГРА, ФРЕЙМВОРК, UNITY, C#, ДОДАТОК, КАМЕРА, ГРАВЕЦЬ, БОС, ПЕРСОНАЖ.

ABSTRACT

Explanatory note: 54 pages, 6 figures, 1 table , 3 appendices., 25 sources.

Object of development: game software application using Unity3D and C #.

Objective of the qualification work: Development of software for the operation of the interactive game "BossFight" using the Unity 3D and C # frameworks

The introduction analyzes the current state of the problem, specifies the task, the purpose of the qualification work and the field of its application, substantiates the relevance of the topic. In the first section the research of the subject area and existing decisions is carried out, the urgency of the task and the purpose of development is determined, the statement of the task is developed.

The second section selects the platform for development, performs program design and development, describes the algorithm and structure of the system, determines the input and output data, provides characteristics of the parameters of technical means, describes the program.

The economic section determines the complexity of the developed software product, calculates the cost of work to create an application and calculates the time to create it.

The practical significance is to create a prototype of a mobile game in which the user is given the opportunity to try their hand at fighting with different opponents.

The relevance of the information system is determined by the high demand for such games and the growing statistics of their popularity since 2016.

List of keywords: PROTOTYPE, GAME, FRAMEWORK, UNITY, C#, APPENDIX, CAMERA, PLAYER, BOSS, CHARACTER.

ЗМІСТ

ВСТУП

В сучасному світі більшу частину часу, не враховуючи сну, людина проводить за роботою. Але без відпочинку працювати неможливо.

Комп'ютерні та мобільні ігри – найбільш розповсюджений спосіб відпочинку, відволікання від життєвих проблем, коротшання часу в дорозі чи просто проведення часу з друзями. На даний момент, найпопулярнішими є ігри жанрів casual та hyper-casual.

Казуальна гра (від англ. Casual game) - гра, призначена для широкого кола користувачів. Казуальні відрізняються простими правилами і не вимагають від користувача витрат часу навчання чи будь-яких особливих навичок; вони дешеві в розробці та при дистрибуції. Багато подібних ігор мають також яскраву і привабливу графіку і мінімум тексту. Казуальним іграм протиставляють «аркадні» ігри зі складними правилами, розраховані порівняно вузьку аудиторію досвідчених гравців.

Гіперказуальна гра – це гра, яка не потребує великих зусиль для ігрового процесу. Особливість гіперказуальних ігор у тому, що в основному вони безкоштовні, мають спрощений інтерфейс користувача, не вимагають спеціального навчання або інструкції для ігрового процесу. В основному застосовується 2D-дизайн з простою колірною схемою і легкі, що не вимагають особливих зусиль механіки, які часто є нескінченно-зацикленими. Часто гіперказуальні ігри обговорюються як бізнес-модель, а не повноцінний жанр мобільних ігор. Через відсутність стійкої внутрішньоігрової економіки та відсутність ціни більшості гіперказуальних ігор дохід здебільшого генерується за рахунок реклами.

Типи внутрішньоігрових рекламних оголошень:

- Відео з винагородою (коли ці рекламні ролики проглядаються, гравець нагороджується внутрішньоігровими нагородами)
- Банерна реклама (реклама, що з'являється внизу екрана користувача)

• Міжсторінкові оголошення (рекламні оголошення, що відображаються між ігровими сеансами)

Саме таку, гіперказуальну, гру я обрав темою своєї роботи.

Тому проблема, розглянута в даній кваліфікаційній роботі, є актуальною та має широке практичне значення.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Загальні відомості з предметної галузі

На даний момент жанр гіперказуальних мобільних ігор заполонив собою простори інтернету. Грою, що започаткувала зростання попиту на цей жанр вважають Helix Jump від Voodoo [1], яка у 2018 році стала найбільш завантажуваною грою у Google Play[2] та посіла друге місце в App Store[3]. З сучасних користувачів смартфонів навряд чи набереться хоч 10% людей, які не грали хоч в одну гру жанру Hyper Casual. Казуальні ігри призначені для широкого кола користувачів, що відрізняються простими правилами і не вимагають від гравців особливої концентрації, витрат часу на навчання чи будь-яких особливих навичок, вони відносно дешеві у розробці та при дистрибуції.

У серії статей The Ascendance of Hyper-casual [4], автором яких є Johannes Heinze, були виділені основні риси гіпер-казуальних ігор, а саме:

– миттєвий доступ до вмісту. Без сюжетних вставок, передзавантаження, вибору рівнів або довгий туторіал - користувач з порога опиняється в епіцентрі гри. Завдяки нехитрому і не перевантаженому деталями геймплею, щоб розібратися в тому, що відбувається, достатньо і декількох секунд. А короткий ігровий цикл дозволяє грати де завгодно і будь-коли;

– мінімалізм у геймплеї та дизайні. Геймплей, часто, настільки простий, наскільки це взагалі можливе. Під стать йому та інші компоненти гри: візуальна складова, управління та аудіо - ніщо не повинно заважати користувачеві повністю сфокусуватися на ігровому процесі;

– проста модель монетизації. Гра не спонукають користувача платити за контент, натомість він «платить» своїм часом. Понад 70% доходу гри посідає рекламу: банери, відеореклама, мотивована реклама [5].

До появи терміну «Гіпер-казуальних ігор» існував їхній аналог – «Супер-казуальні ігри». Таким чином, можна зрозуміти, що це не новий жанр, а лише

нова класифікація затребуваних продуктів. Нижче наведено графік з представленням популярних гіпер-казуальних ігор із кількістю завантажень (рис 1.1).

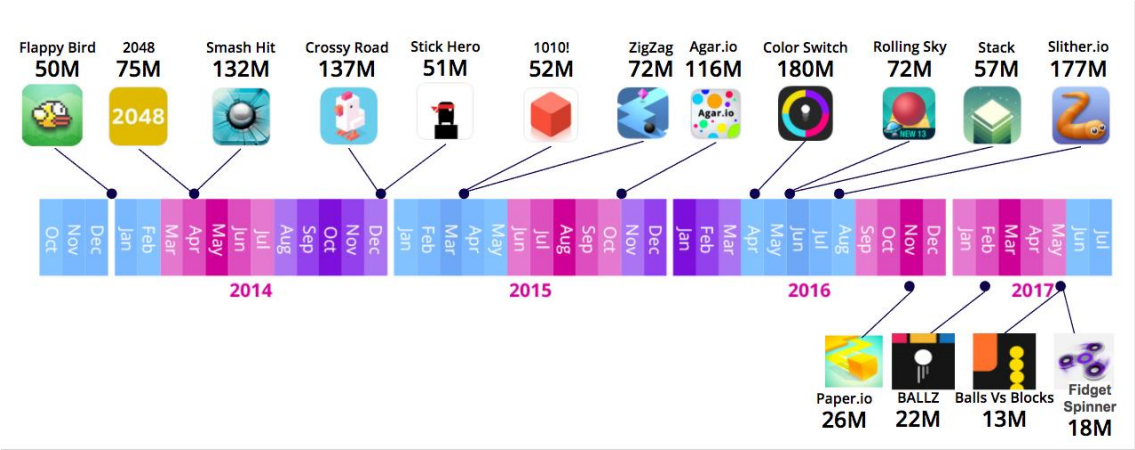

Рис. 1.1. Графік популярності гіпер-казуальних ігор

Як можна побачити, гіпер-казуальні ігри і так цілком непогано почувалися. Але чому у 2018 активно заговорили про якийсь тренд і все закрутилося з новою силою? Справа в тому, що великі компанії нарешті помітили «маленькі ігри» і почали вливати в них значні бюджети:

– Zynga купує Gram Games (творців франшизи Merge!) за $250 млн;

– інвестиційний банк Goldman Sachs вкладає у Voodoo $200 млн., щоб розвивати видавничий напрямок;

– Peak Games (творці Toon Blast) продають Zynga свої казуальні карткові ігри за $100 млн;

– Zynga купує студію-стартап із однієї людини Harpan LLC за $42.5 млн.

У плані окупності за такі ігри найкраще кажуть цифри. Свого часу Flappy Bird приносила творцю по $50.000 на день, за словами CEO Apptopia Елірана Сапіра (Eliran Sapir) Fidget Spinner від Ketchapp на день заробляла $57.000, а Scream Go Hero принесла близько $600.000 трохи більше, ніж за 2 місяці.

Раніше вже зазначалося, що основа монетизації гіпер-казуальних ігор - це реклама разом із великою кількістю установок.

Через те, що життєвий цикл у подібних ігор, як правило, вкрай низький, вони постійно потребують припливу свіжої крові. І в цьому їм допомагає висока віральність та низький CPI. За даними Tenjin, залучення користувачів для гіпер-казуальної гри обходиться в 7-13 разів дешевше, ніж для стандартної мобільної гри, і в середньому становить $0.16 для Android і $0.28 для iOS.

1.2. Призначення розробки та галузь застосування

Система яка розробляється, призначена, в першу чергу, для школярів, студентів та офісних працівників для відпочинку.

1.3. Підстава для розробки

Відповідно до освітньої програми, згідно навчального плану та графіків навчального процесу, в кінці навчання студент виконує кваліфікаційну роботу.

Тема роботи узгоджується з керівником кваліфікаційної роботи, випускаючою кафедрою, та затверджується наказом ректора.

Отже, підставами для розробки (виконання кваліфікаційної роботи) є:

- освітня програма 121 Інженерія програмного забезпечення;
- навчальний план та графік навчального процесу;
- наказ ректора Національного технічного університету «Дніпровська політехніка» № 268-с від 18.05.2022 р;
- завдання на кваліфікаційну роботу на тему “Розробка програмного забезпечення для функціонування інтерактивної гри «BossFight» з використанням фреймворка Unity 3D та C#”.

1.4. Постановка завдання

Розробити прототип гри на мобільну платформу. Система має відповідати сучасним стандартам розробки ігор, мати гарну оптимізацію та цікавий геймплей.

Функціональні можливості:

1. Налаштування плавної камери.
2. Створити основні механіки для бою:
   - стрільба з луку;
   - використання магії;
   - атаки ближнього бою мечем;
   - атаки ближнього бою руками і ногами.
3. Створення зручної системи керування персонажем свайпами.
4. Створення кількох супротивників в якості боссів та їх штучного інтелекту.

1.5. Вимоги до функціональних характеристик

Система має бути реалізована як застосунок на Android та IOS з доступом через GooglePlay та AppStore і запускатись на більшості девайсів [6].

1.6. Вимоги до інформаційної безпеки

На етапі прототипування програма не має своєї бази даних, тому основна безпека повинна бути забезпечена характеристикам бійців.

1.7. Вимоги до складу та параметрів технічних засобів

Гра підтримується на більшості смартфонів з операційною системою Android та IOS, які підтримуються відповідними магазинами додатків.

1.8. Вимоги до інформаційної та програмної сумісності

У якості основної мови для розробки кваліфікаційної роботи повинна бути використана мова C#. Середовище розробки обрано Microsoft Visual Studio - серія продуктів фірми Майкрософт, які включають інтегроване середовище розробки програмного забезпечення та ряд інших інструментальних засобів [7].

У якості фреймворку обрано Unity3D. Це міжплатформне середовище розробки комп'ютерних ігор, розроблене американською компанією Unity Technologies. Unity дозволяє створювати програми, що працюють на більш ніж 25 різних платформах, що включають персональні комп'ютери, ігрові консолі, мобільні пристрої, інтернет-програми та інші [8].

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

2.1. Функціональне призначення програми

Результатом кваліфікаційної роботи має бути мобільний додаток, який надає користувачу можливість спробувати основні механіки гри та перемогти трьох босів.

Додаток має містити дві сцени:

– головне меню – сцена, що містить назву гри та кнопку старту;

– бойова арена – сцена, на якій відбувається безпосередньо гра.

Користувач повинен мати можливість керувати своїм персонажем що передбачає наступний функціонал:

– переміщення персонажу за допомогою свайпів на екрані;

– стрільба з луку по ворогу коли персонаж стоїть на місці;

– застосування спеціального вміння – прискорення швидкості стрільби.

В додатку повинні бути присутні три боси, кожен з яких має свої особливі механіки:

1) Лицар – великий воїн у латах, завдяки чому він має більше здоров’я і повинен мати наступні навички:
   – сильний удар згори вниз;
   – бокові удари зліва направо та справа наліво;
   – серія ударів, що йдуть один за одним.
2) Бойовий маг – чаклун, який повинен мати наступні особливі навички:
   – телепортація від гравця на комфортну для нього дистанцію
   – стрільба вогняними кулями;
   – стрільба вогняним променем, який слідує за гравцем певний час;
   – виклик землетрусу.
3) Монах – майстер бойових мистецтв ногами і повинен мати наступні навички:

– швидке переміщення до персонажу користувача з ударом;
– навички увороту за допомогою сальто назад;
– кругові удари ногами навколо себе.

По результатах бою у гравця може бути два варіанти результату:

1) гравець перемагає в бою і отримує можливість битись з наступним супротивником;
2) гравець програє в бою и може спробувати знову.

Однією з головних потреб у таких проектах є камера, створення якої буде виконано за допомогою cinemachine, що буде описано в пункті 2.3.

2.2. Опис застосованих математичних методів

2.2.1. Математична модель

Завдяки можливостям обраного фреймворку при розробці даного програмного забезпечення не потрібно створювати власні складні формули для обробки фізики персонажів та об’єктів.

2.2.2. Метод оптимізації

Головною проблемою оптимізації таких ігор є навантаження відеопроцесору. Для оптимізації цього аспекту використовується кілька головних способів:

– використання атласів спрайтів та текстур – завантаження зображень на одне полотно, а потім за допомогою SpriteRenderer порізати полотно на частини, кожна з яких буде використана як окреме зображення. Завдяки цьому прийому значно зменшується кількість проуколів (викликів відеопроцесора), чим зменшує навантаження;

– зменшення кількості полігонів в моделях що так само зменшить навантаження на відеопроцесор [9].

2.3. Опис використаних технологій та мов програмування

Список використаних мов програмування, фреймворків та додаткових бібліотек, що було використано при розробці програмного продукту наведено у таблиці 2.1.

Таблиця 2.1

Список основних технологій та мов програмування

Середовище розробки	
Фреймворк	Unity 2021.2.18f1
Мова програмування	C#
Додаткові плагіни	
Cinemachine	Плагін для створення плавної камери
TextMeshPro	Плагін, що містить текстові компоненти з гнучким налаштуванням
Visual Studio Code Editor та Visual Studio Editor	Плагіни, які забезпечують спільну роботу фреймворку Unity та компілятору Visual Studio
Розгортання	
Платформа	PlayMarket

C# - об'єктно-орієнтована мова програмування. Розроблено в 1998-2001 роках групою інженерів компанії Microsoft під керівництвом Андерса Хейлсберга і Скотта Вільтаумота як мова розробки програм для платформи Microsoft .NET Framework та .NET Core.

C# відноситься до сім'ї мов з C-подібним синтаксисом, їх синтаксис найбільш близький до C++ і Java [10]. Мова має статичну типізацію, підтримує поліморфізм, перевантаження операторів (у тому числі операторів явного та неявного приведення типу), делегати, атрибути, події, змінні, властивості,

узагальнені типи та методи, ітератори, анонімні функції з підтримкою замикань, LINQ, винятки, коментарі у форматі XML.

Перейнявши багато від своїх попередників — мов C++, Delphi, Модула, Smalltalk і, особливо, Java — C#, спираючись на практику їх використання, виключає деякі моделі, що зарекомендували себе проблематичні при розробці програмних систем, наприклад, C# на відміну від C++ не підтримує множинне успадкування класів (між тим допускається множинна реалізація інтерфейсів) [11].

Саме ця мова програмування дозволяє найкраще створювати ігри у фреймворку Unity3D завдяки великій кількості плагінів та налаштувань.

Unity - багатоплатформовий інструмент для розробки відеоігор і застосунків, і рушій, на якому вони працюють. Створені за допомогою Unity програми працюють на настільних комп'ютерних системах, мобільних пристроях та гральних консолях у дво- та тривимірній графіці, та на пристроях віртуальної чи доповненої реальності. Застосунки, створені за допомогою Unity, підтримують DirectX та OpenGL.

Основні можливості Unity:

1. Робота з ресурсами. Редактор Unity має інтерфейс, що складається з різних вікон, які можна розташувати на свій розсуд. Завдяки цьому можна проводити налагодження гри чи застосунка прямо в редакторі. Головні вікна — це оглядач ресурсів проєкту, інспектор поточного об'єкта, вікно попереднього перегляду, оглядач сцени та оглядач ієрархії ресурсів.

Проєкт в Unity поділяється на сцени (рівні) — окремі файли, що містять свої ігрові світи зі своїм набором об'єктів, сценаріїв, і налаштувань. Сцени можуть містити в собі як об'єкти-моделі (ландшафт, персонажі, предмети довкілля тощо), так і порожні ігрові об'єкти — ті, що не мають моделі, проте задають поведінку інших об'єктів (тригери подій, точки збереження прогресу тощо). Їх дозволяється розташовувати, обертати, масштабувати, застосовувати до них скрипти. В них є назва (в Unity допускається наявність двох і більше об'єктів з однаковими назвами), може бути тег (мітка) і шар, на якому він повинен

відображатися. Так, у будь-якого предмета на сцені обов'язково наявний компонент Transform — він зберігає в собі координати місця розташування, повороту і розмірів по всіх трьох осях. У об'єктів з видимою геометрією також за умовчанням присутній компонент Mesh Renderer, що робить модель видимою. Різні моделі можуть об'єднуватися в набори (ассети) для швидкого доступу до них. Наприклад, моделі споруд на спільну тему.

Unity підтримує фізику твердих тіл і тканини, фізику типу Ragdoll (ганчіркова лялька). У редакторі є система успадкування об'єктів; дочірні об'єкти будуть повторювати всі зміни позиції, повороту і масштабу батьківського об'єкта. Скрипти в редакторі прикріплюються до об'єктів у вигляді окремих компонентів.

У 2D іграх Unity переважно використовує спрайти. В 3D іграх Unity здебільшого використовує тривимірні моделі (меші), на які накладаються текстури (зумовлюють вигляд поверхні об'єктів), матеріали (зумовлюють як поверхня реагуватиме на різні фактори) та шейдери (невеликі скрипти, за яким вираховується зміна кольору кожного пікселя згідно заданих параметрів, як-от розсіяння відбитого світла). В обох видах застосовуються системи часток для відображення субстанцій, таких як рідини чи дим.

Unity підтримує стиснення текстур, міпмапінг і різні налаштування роздільності екрана для кожної платформи; забезпечує бамп-мапінг, мапінг відображень, паралакс-мапінг, затінення навколишнього світла у екранному просторі, динамічні тіні за картами тіней, рендер у текстуру та повноекранні ефекти обробки зображення, такі як зернистість, глибина чіткості, розмиття в русі, відблиски віртуальних лінз або ореол навколо джерел світла.

2. Рендеринг. Рендеринг зображення відбувається через віртуальну камеру огляду. В робочій області редактора ігрова сцена може розміщуватися як завгодно, а при рендерингу — так, як її видно з камери. В сцені може бути декілька камер, які рухаються за персонажем чи за вказаною траєкторією. Вигляд з камери подається двовимірно або тривимірно (в перспективі або ортографічно).

Фон сцени, видимий через камеру, типово зображає небо, утворене скайбоксом, але може презентувати й інше довкілля [12].

Графічний рушій використовує DirectX (Windows), OpenGL (Mac, Windows, Linux), OpenGL ES (Android, iOS), та спеціальне власне API для Wii. Також підтримуються bump mapping, reflection mapping, parallax mapping, screen space ambient occlusion (SSAO), динамічні тіні з використанням shadow maps, render-to-texture та повноекранні ефекти post-processing.

Unity підтримує файли 3ds Max, Maya, Softimage, Blender, modo, ZBrush, Cinema 4D, Cheetah3D, Adobe Photoshop, Adobe Fireworks та Allegorithmic Substance. В ігровий проєкт Unity можна імпортувати об'єкти цих програм та виконувати налаштовування за допомогою графічного інтерфейсу.

Для написання шейдерів використовується ShaderLab, що підтримує шейдерні програми написані на GLSL або Cg. Шейдер може включати декілька варіантів реалізації, що дозволяє Unity визначати найкращий варіант для конкретної відеокарти. Unity також має вбудовану підтримку фізичного рушія Nvidia PhysX (колишнього Ageia), підтримку симуляції одягу в системі реального часу на довільній та прив'язаній полігональній сітці (починаючи з Unity 3.0), підтримку системи ray casts та шарів зіткнення [13].

3. Скрипти. Скриптова система ігрового рушія зроблена на Mono — вільному відкритому проєкті з реалізації .NET Framework. Програмісти можуть використовувати UnityScript (власна скриптова мова, подібна до JavaScript та ECMAScript), C# або Boo (мова програмування, подібна до Python). Починаючи з версії 3.0, до Unity входить перероблена версія MonoDevelop для зневадження скриптів.

З виходом версії 5.2 у 2015 році передбачена вбудована можливість редагувати скрипти у середовищі Visual Studio.

4. Asset Tracking. В Unity включено систему контролю версій для ігрових об'єктів та скриптів під назвою Unity Asset Server. Система використовує PostgreSQL, роботу зі звуком, побудовану на основі бібліотеки FMOD (з можливістю програвати Ogg Vorbis аудіофайли), відеопрогравач із кодеком

Theora, рушій для побудови ландшафтів рослинності, вбудовану систему карт освітлення (Beast), мережу для мультиплеєру (RakNet) та вбудовані навігаційні меші для пошуку шляху [14].

Cinemachine – це набір модулів для керування камерою Unity. Cinemachine вирішує складну математику та логіку відстеження цілей, компонування, змішування та вирізання між кадрами. Він призначений для значного скорочення кількості трудомістких ручних маніпуляцій і переглядів сценаріїв, які відбуваються під час розробки.

Процедурний характер цих модулів робить Cinemachine стійким до помилок. Коли ви робите налаштування — наприклад, змінюєте анімацію, швидкість транспортного засобу, місцевість або інші ігрові об’єкти у вашій сцені — Cinemachine динамічно коригує свою поведінку, щоб зробити найкращий знімок. Немає потреби, наприклад, переписувати сценарії камери лише тому, що персонаж повертає ліворуч, а не праворуч.

Cinemachine працює в режимі реального часу в усіх жанрах, включаючи FPS, від третьої особи, 2D, бічну прокрутку, зверху вниз і RTS. Він підтримує стільки знімків у вашій сцені, скільки вам потрібно. Його модульна система дозволяє створювати складні моделі поведінки.

Cinemachine добре працює з іншими інструментами Unity, діючи як потужне доповнення до шкали часу, анімації та ресурсів постобробки. Створіть власні розширення або інтегруйте їх зі своїми власними сценаріями камери [15].

2.4. Опис використаної архітектури та шаблонів проектування

Під час розробки додатку були використані наступні аспекти створення програмного забезпечення:

1. SOLID – набір принципів, до яких відноситься принцип єдиної відповідальності (SRP), принцип відкритості/закритості (OCP), принцип заміщуваності (LCP), принцип розділення інтерфейсів (ISP) та принцип інверсії залежностей (DIP) [16].

2. ООП - методологія програмування, заснована на уявленні програми у вигляді совокупності взаємодіючих об’єктів, кожний з яких є екземпляром певного класу, а класи складають ієрархію наслідування. Основними принципами ООП є інкапсуляція, наслідування і поліморфізм [17].

3. Компонентність - основний принцип роботи в Unity. Гра створюється зі сцен, на яких розташовані об’єкти. До об’єктів, в свою чергу, прикріплені певні скрипти. Часто буває так, що на одному об’єкті може бути одночасно велика кількість маленьких скриптів, кожний з яких відповідає за певну частину функціоналу згідно з принципом єдиної відповідальності [18].

При проектуванні архітектури найбільшу увагу звертав на ефективність, гнучкість та розширяємість системи, бо саме ці три принципи є найважливішими в розробці мобільних ігор на Unity.

2.5. Опис структури програми та алгоритмів її функціонування

В основі структури системи лежить клієнтська архітектура, в якій не використовується сервер. Завдяки цьому, гра може функціонувати без доступу до мережі інтернет і дозволяє коректно користуватись додатком у будь-яких умовах [19][21].

За допомогою додатку, користувач в меню має можливість лише запустити гру, але завдяки створеній системі легко імплементуються додаткові можливості, наприклад статистика битв, магазин, тощо [20].

2.6. Обґрунтування та організація вхідних та вихідних даних програми

Враховуючи особливості розроблюваного програмного продукту, користувач не повинен надавати додаткові дані для користування додатком.

В якості вихідних даних виступає реакція додатку на дії користувача та результати боїв між ним і босом.

2.7. Опис розробленого програмного продукту

2.7.1. Використані технічні засоби

Для використання даного програмного додатку смартфон користувача повинен відповідати наступним мінімальним технічним характеристикам:

- операційна система IOS 7 або Android 4.0 і вище;
- 1 Гб оперативної пам’яті і більше;
- процесор Spreadtrum SC7731E (1.3 ГГц) і вище;
- 68 Мб вільного місця на носії.

2.7.2. Використані програмні засоби

Під час розробки були використані наступні програмні засоби:

- VisualStudio Code 2019;
- GitHub, Fork;
- Unity3D;
- Cinemachine.

2.7.3. Виклик та завантаження програми

Для отримання доступу до мобільного додатку слід безкоштовно завантажити його з PlayMarket, після чого запустити через ярлик, який буде створено автоматично.

2.7.4. Опис інтерфейсу користувача

1. Головне меню

Головне меню додатку містить заголовок з назвою гри і велику кнопку START (рис. 2.1), при натисканні на яку користувач переходить до процесу гри.

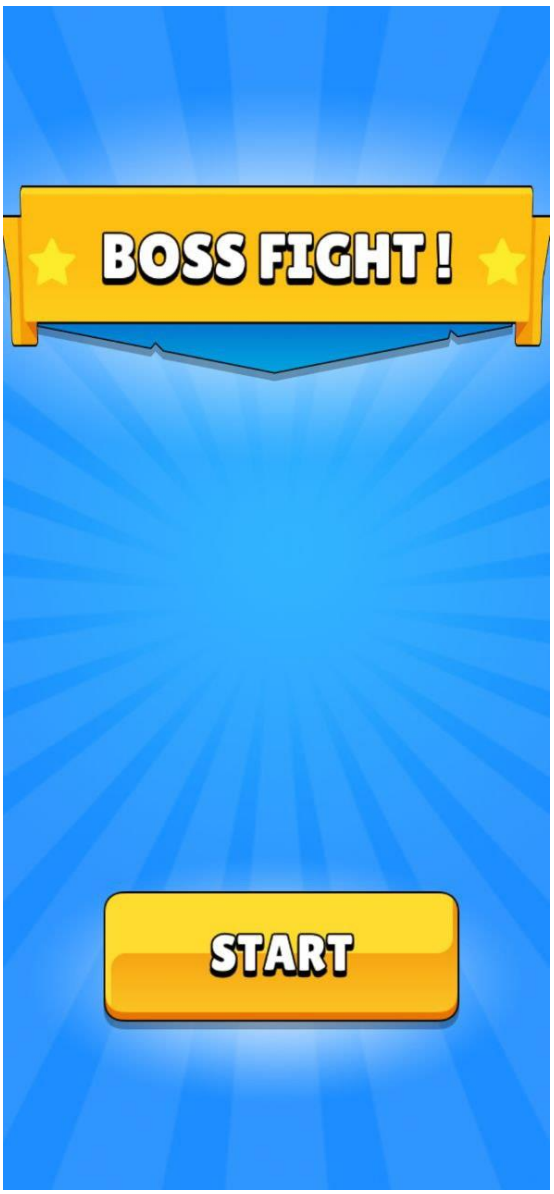

Рис. 2.1. Головне меню

Після натискання кнопки старту користувач потрапляє на сцену бою з босом (рис 2.2). З інтерфейсу користувачеві доступні:

– Панель інформації про боса – його ім’я та здоров’я – у верхній частині екрану;

– Кнопка спеціального вміння в правій нижній частині екрану.

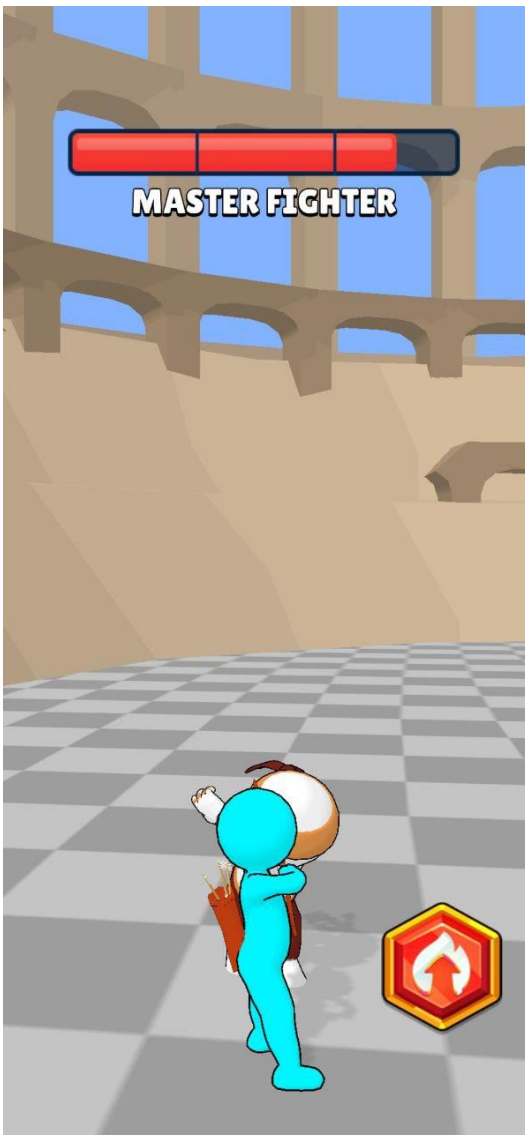

Рис 2.2. Бойова сцена

За результатами дій користувача він може отримати один з двох екранів: екран перемоги (рис. 2.3), якщо користувач зміг подолати супротивника, або екран програшу (рис. 2.4), якщо користувач не зміг подолати супротивника.

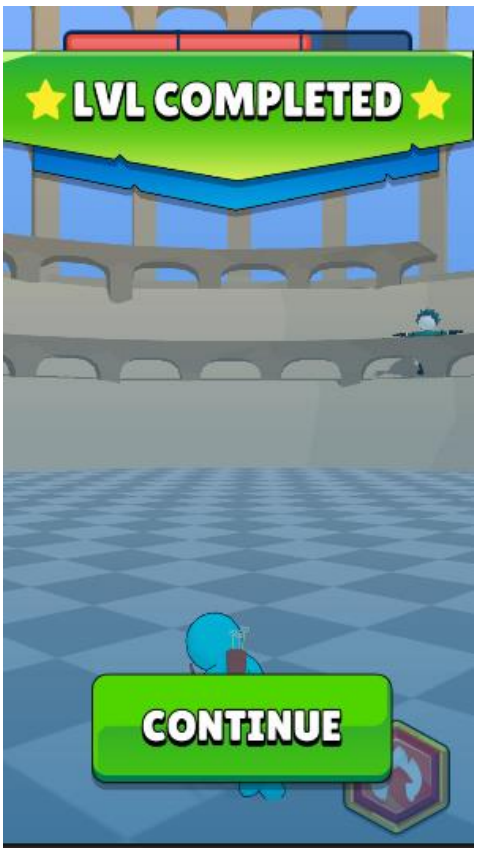

Рис. 2.3. Екран перемоги

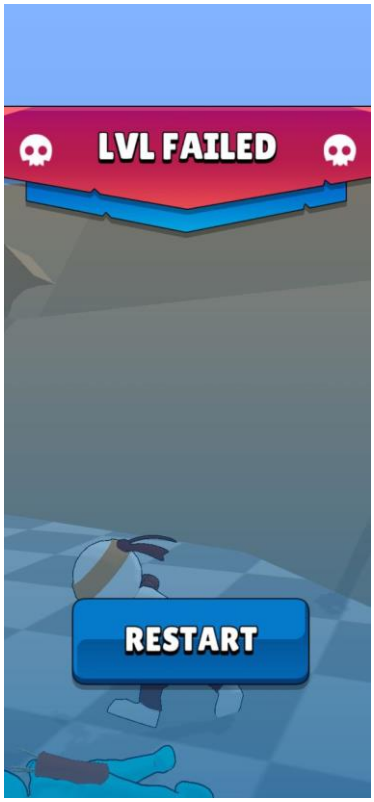

Рис. 2.4. Екран програшу

В верхній частині екрану користувачеві надається інформація про поточний статус гри (перемога чи поразка), а в нижній частині розташована кнопка, яка або перезапускає поточний рівень в разі поразки, або завантажує наступний в разі перемоги.

РОЗДІЛ 3
ЕКОНОМІЧНИЙ РОЗДІЛ

3.1.Обчислення трудомісткості розробки програмного забезпечення

Вхідні дані:

1. q – передбачуване число операторів – 635.

2. C – коефіцієнт складності програми – 1,5.

3. p – коефіцієнт корекції програми в ході її розробки – 0,08.

4. B – коефіцієнт збільшення витрат – 1,2.

5. k – коефіцієнт кваліфікації програміста – 0,8.

6. $С_{ПР}$– середня годинна заробітна плата програміста, грн/год. Згідно зі статистичними даними, для програміста кваліфікації Junior+ Unity Developer[22] – в середньому 1000$ на місяць. При 40 годинному робочому тижні та чинному курсі Національного банку України 1 долар = 29,68гривень [23], отримаємо значення 185,5 грн/год [24].

7. B_k– число виконавців – 1.

8. F_p– місячний фонд робочого часу (при 40 годинному робочому тижні $F_p = 176$ годин).

9. $С_{Мч}$ – вартість машино-години ЕОМ, грн/год. З урахуванням амортизації складових частин ЕОМ та при чинному тарифі на електроенергію для населення [25], при споживанні до 250 КВт – 1,44 грн за КВт/год – 4,5 грн/год.

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки ПЗ може бути розрахована на основі системи моделей з різною точністю оцінки.

Трудомісткість розробки ПЗ можна розрахувати за формулою (3.1):

$$t = t_o + t_u + t_a + t_n + t_{отл} + t_д, \text{ людино-годин,} \qquad (3.1)$$

де t_o – витрати праці на підготовку й опис поставленої задачі (приймається 50),

t_u – витрати праці на дослідження алгоритму рішення задачі,

t_a – витрати праці на розробку блок-схеми алгоритму,

t_n – витрати праці на програмування по готовій блок-схемі,

$t_{\text{отл}}$ – витрати праці на налагодження програми на ЕОМ,

$t_{\text{д}}$ – витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у ПЗ, яке розробляється. Умовне число операторів (підпрограм) визначається за формулою (3.2):

$$Q = q * C * (1 + p), \tag{3.2}$$

де q – передбачуване число операторів,

C – коефіцієнт складності програми,

p – коефіцієнт кореляції програми в ході її розробки.

$$Q = 635 * 1{,}5 * (1 + 0{,}08) = 1\,028{,}7$$

Витрати праці на вивчення опису задачі t_u визначається за формулою (3.3) з урахуванням уточнення опису і кваліфікації програміста:

$$t_u = \frac{Q * B}{80 * k}, \text{людино} - \text{годин}, \tag{3.3}$$

де B – коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі,

k – коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності.

$$t_u = \frac{1\,028{,}7 * 1{,}2}{80 * 0{,}8} = 19{,}29, \text{людино} - \text{годин}.$$

Витрати праці на розробку алгоритму рішення задачі обчислюються за формулою (3.4):

$$t_a = \frac{Q}{20 * k}, \text{людино} - \text{годин.} \quad (3.4)$$

$$t_a = \frac{1\,028{,}7}{20 * 0{,}8} = 64{,}29, \text{людино} - \text{годин.}$$

Витрати на складання програми по готовій блок-схемі визначаються за формулою (3.5):

$$t_n = \frac{Q}{23 * k}, \text{людино} - \text{годин.} \quad (3.5)$$

$$t_n = \frac{1\,028{,}7}{23 * 0{,}8} = 55{,}9, \text{людино} - \text{годин.}$$

Витрати праці на налагодження програми на ЕОМ обчислюються за формулою (3.6):

$$t_{\text{отл}} = \frac{Q}{4 * k}, \text{людино} - \text{годин.} \quad (3.6)$$

$$t_{\text{отл}} = \frac{1\,028{,}7}{4 * 0{,}8} = 321{,}46, \text{людино} - \text{годин.}$$

Витрати праці на підготовку документації обчислюються за формулою (3.7):

$$t_{\text{д}} = t_{\text{др}} + t_{\text{до}}, \quad (3.7)$$

де $t_{\text{др}}$ – трудомісткість підготовки матеріалів і рукопису визначається формулою (3.8).

$$t_{\text{др}} = \frac{Q}{15 * k}, \text{людино} - \text{годин}, \tag{3.8}$$

$$t_{\text{др}} = \frac{1\,028{,}7}{15 * 0{,}8} = 85{,}7, \text{людино} - \text{годин}.$$

$t_{\text{до}}$ – трудомісткість редагування, печатки й оформлення документації визначається формулою (3.9).

$$t_{\text{до}} = 0{,}75 * t_{\text{др}}, \text{людино} - \text{годин}. \tag{3.9}$$

$$t_{\text{до}} = 0{,}75 * 85{,}7 = 65{,}6, \text{людино} - \text{годин}.$$

Звідси витрати праці на підготовку документації:

$$t_{\text{д}} = 85{,}7 + 65{,}6 = 151.3, \text{людино} - \text{годин}.$$

Трудомісткість розробки ПЗ, відповідно, дорівнює:

$$t = 50 + 19{,}29 + 64{,}29 + 55{,}9 + 321{,}46 + 151{,}3 = 690{,}95, \text{людино} - \text{годин}$$

3.2.Обчислення витрат на створення програмного забезпечення

Витрати на створення ПЗ $К_{ПО}$ включають витрати на заробітну плату виконавця програми $З_{ЗП}$ і витрат машинного часу, необхідного на налагодження програми на ЕОМ та визначаються за формулою (3.10):

$$К_{ПО} = З_{ЗП} + З_{МВ}, \textit{грн.} \tag{3.10}$$

Заробітна плата виконавців визначається за формулою (3.11):

$$З_{ЗП} = t * С_{ПР}, \text{грн}, \tag{3.11}$$

де: t – загальна трудомісткість, людино-годин,

$С_{ПР}$ – середня годинна заробітна плата програміста, грн/година.

$$З_{ЗП} = 690{,}95 * 185{,}5 = 128\,171{,}2, \text{грн.}$$

Вартість машинного часу, необхідного для налагодження програми на ЕОМ:

$$З_{МВ} = t_{отл} * С_{мч}, \text{грн,} \qquad (3.12)$$

де $t_{отл}$ – трудомісткість налагодження програми на ЕОМ, год,

$С_{мч}$ - вартість машино-години ЕОМ, грн/год.

$$З_{МВ} = 321{,}46 * 4{,}5 = 1\,446{,}57, \text{грн.}$$

Відповідно, витрати на створення ПЗ становлять:

$$К_{ПО} = 128\,171{,}2 + 1\,446{,}57 = 129\,617{,}77, \text{грн.}$$

Визначені в такий спосіб витрати на створення програмного забезпечення є частиною одноразових капітальних витрат на створення АСУП.

Очікуваний період створення ПЗ:

$$T = \frac{t}{B_k * F_p}, \text{міс,} \qquad (3.13)$$

де B_k - число виконавців,

F_p - місячний фонд робочого часу.

$$T = \frac{690{,}95}{1 * 176} = 3{,}93, \text{міс.}$$

Висновки: трудомісткість розробленого програмного додатку становить 690,95 людино-години. Обчислена вартість роботи по створенню програми

дорівнює 129 617,77 гривень. Визначений затрачений час на створення програмного забезпечення становить 3,93 місяців.

ВИСНОВКИ

На даний час гіперказуальні ігри є найбільш популярним жанром мобільних ігор завдяки ряду переваг, наприклад легкість процесу гри, відносна дешевизна виробництва, низькі технічні вимоги до користувачів.

Метою кваліфікаційної роботи є створення прототипу гіперказуальної гри, використовуючи фреймворк Unity3D та мову програмування C#.

Під час виконання даної роботи був досліджений ринок гіперказуальних ігор на предмет аналогів для сворення нового продукту для залучення користувачів.

Додаток створений за допомогою Unity3D та мови програмування C# з використанням наступних плагінів:

1. Cinemachine – плагін, який покращує роботу з камерою і надає необхідні інструменти для гнучкого та плавного налаштування.

2. TextMeshPro – плагін, який додає нові можливості для створення текстових полів.

3. Visual Studio Code Editor – плагін, який створює зв’язок між редактором коду VisualStudioCode2019 та Unity3D.

4. Firebase – плагін для збору статистики і даних користувача, необхідний для викладання продукту в PlayMarket.

Працездатність даного програмного продукту підтверджується вдалими експлуатаційними випробуваннями.

У проектуванні та реалізації програмного забезпечення використовуються сучасні технології та принципи розробки. Їх використання пришвидшує процес створення та введення в експлуатації програми, та дозволяє гнучко додавати та оновлювати функціонал продукту.

В економічному розділі визначено трудомісткість розробленої інформаційної системи (690,95 люд-год), проведений підрахунок вартості роботи по створенню програми (129 617,77 грн.) та розраховано час на його створення (3,93 міс).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. https://play.google.com/store/apps/developer?id=VOODOO&hl=ru&gl=US
2. https://play.google.com/
3. https://www.apple.com/ua/app-store/
4. https://www.pocketgamer.biz/comment-and-opinion/65256/the-ascendance-of-hypercasual/
5. https://admob.google.com/intl/ru/home/resources/monetize-mobile-game-with-ads/
6. https://www.technodor.info/2019/04/blog-post.html
7. https://visualstudio.microsoft.com/ru/
8. https://unity.com/ru
9. https://docs.unity3d.com/2022.1/Documentation/Manual/optimizing-draw-calls.html
10. https://dic.academic.ru/dic.nsf/ruwiki/249655
11. https://ru.wikipedia.org/wiki/C_Sharp
12. https://wikipedia.org/wiki/%D0%A0%D0%B5%D0%BD%D0%B4%D0%B5%D1%80%D0%B8%D0%BD%D0%B3
13. https://wikipedia.org/wiki/%D0%A8%D0%B5%D0%B9%D0%B4%D0%B5%D1%80
14. C# 8.0 and .NET, Core 3.0 - Mark J. Price
15. https://uk.wikipedia.org/wiki/Unity_(рушій_гри)
16. https://unity.com/ru/unity/features/editor/art-and-design/cinemachine
17. https://wikipedia.org/wiki/SOLID
18. https://habr.com/ru/post/463125/
19. https://docs.unity3d.com/ru/2018.4/Manual/UsingComponents.html
20. https://medium.com/nuances-of-programming
21. https://habr.com/ru/company/1cloud/blog/424911/
22. https://dou.ua/lenta/articles/making-implementation-plan/

23. Статистика зарплат Unity програмістів взята з сайтів work.ua, linkedin.com та ingame.job.ua.

24. https://bank.gov.ua/

25. Тарифы на электроэнергию в 2022 году URL: https://index.minfin.com.ua/tariff/electric/ (дата звернення 06.06.2022).

ДОДАТОК А

Лістинг програми

ДОДАТОК Б

Відгук керівника економічного розділу

ДОДАТОК В

ПЕРЕЛІК ДОКУМЕНТІВ НА МАГНІТНОМУ НОСІЇ

Ім’я файлу	Опис
Пояснювальні документи	
Кваліфікаційна робота_ Бігма.docx	Пояснювальна записка до кваліфікаційної роботи. Документ Word.
Кваліфікаційна робота_ Бігма.pdf	Пояснювальна записка до кваліфікаційної роботи в форматі PDF
Програма	
BossFight.rar	Архів. Містить коди програми і скомпільовану програму
Презентація	
Презентація_ Бігма.pptx	Презентація кваліфікаційної роботи