МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри, канд. тех. наук,
доцент_____ Є. О. Давиденко

«___» _____2023р.

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

МОБІЛЬНИЙ ЗАСТОСУНОК МОНІТОРИНГУ ЗДОРОВ'Я У ВИГЛЯДІ КАЗУАЛЬНОЇ ГРИ НА ПЛАТФОРМІ UNITY

Спеціальність «Інженерія програмного забезпечення»

121 – КРБ.1 – 408.21910801

Студент _____ Ю. С. Афонін
підпис
«__» _____ 2023 р.

Керівник д-р техн. наук, професор _____ І. М. Журавська
підпис
«__» _____ 2023 р.

Консультант канд. техн. наук, доцент _____ А. О. Алєксєєва
підпис
«__» _____ 2023 р.

Миколаїв – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ЗАТВЕРДЖУЮ
Зав. кафедри
_____Є. О. Давиденко
«_____» _____ 2023 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи бакалавра

Видано студенту групи 408 факультету комп'ютерних наук

Афоніну Юрію Сергійовичу

(прізвище, ім'я, по батькові студента)

1. Тема кваліфікаційної роботи:

Мобільний застосунок моніторингу здоров'я у вигляді казуальної гри на платформі Unity

Затверджена наказом по ЧНУ від «17» березня 2023 р. № 60

2. Строк представлення кваліфікаційної роботи «26» червня 2023 р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні

Очікуваним результатом є прототип мобільного ігрового застосунку із функціями моніторингу біомедичних показників. Початково: наявність фітнес-браслету.

4. Перелік питань, що підлягають розробці

- дослідження предметної галузі та аналіз застосунків-аналогів;
- визначення проблем та пошук рішень;
- моделювання та проєктування ПЗ;
- розробка функціоналу ігрового застосунку;
- впровадження технології Bluetooth Low Energy;
- тестування прототипу ігрового застосунку;
- аналіз результатів розробки;

5. Перелік графічних матеріалів

Презентація.

6. Завдання до спеціальної частини

_____

7. Консультанти:

Консультант	Кафедра (організація)	Частина роботи
канд. техн. наук, доц. Алексєєва А. О.	Кафедра екології Медичного інституту ЧНУ ім. Петра Могили	Спеціальна частина з охорони праці

Керівник роботи д-р техн. наук, проф. Журавська Ірина Миколаївна
(посада, прізвище, ім'я, по батькові)

_____
(підпис)

Завдання прийнято до виконання
Афонін Юрій Сергійович
(прізвище, ім'я, по батькові студента)

_____
(підпис)

Дата видачі завдання «____» _____ 2023 р.

КАЛЕНДАРНИЙ ПЛАН

виконання кваліфікаційної роботи

Тема: Мобільний застосунок моніторингу здоров'я у вигляді казуальної гри на платформі Unity _____

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Розробка та затвердження завдання на виконання КРБ	17.03.2023	20.03.2023	виконано
2.	Огляд літератури за темою роботи	21.03.2023	24.03.2023	виконано
3.	Складання календарного плану КРБ	27.03.2023	28.03.2023	виконано
4.	Аналіз предметної області	29.03.2022	31.03.2022	виконано
5.	Визначення проєктних проблем та пошук рішень	03.04.2023	06.04.2023	виконано
6.	Моделювання та проєктування ПЗ	07.04.2023	12.04.2023	виконано
7.	Розробка функціоналу ігрового застосунку, тестування та апробація розробленого ПЗ, аналіз результатів тестування	13.04.2023	04.06.2023	виконано
8.	Розробка спеціальної частини з охорони праці	15.05.2023	31.05.2023	виконано
9.	Оформлення КРБ та презентації	02.06.2023	06.06.2023	виконано
10.	Відгук керівника КРБ	05.06.2023	09.06.2023	виконано
11.	Попередній захист	06.06.2023	07.06.2023	виконано
12.	Рецензування	09.06.2023	13.06.2023	виконано
13.	Завершення оформлення КРБ та презентації	09.06.2023	14.06.2023	виконано
14.	Захист кваліфікаційної роботи	26.06.2023	26.06.2023	виконано

Розробив студент Афонін Юрій Сергійович _____

(*прізвище, ім'я, по батькові)* *(підпис)*

«__» _____ 2023 р.

Керівник роботи д-р техн. наук, проф. Журавська І. М. _____

(*посада, прізвище, ім'я, по батькові)* *(підпис)*

«__» _____ 2023 р.

АНОТАЦІЯ
до кваліфікаційної роботи бакалавра
«Мобільний застосунок моніторингу здоров'я у вигляді казуальної гри на платформі Unity»

Студент гр. 408: Афонін Юрій Сергійович
Керівник: д-р техн. наук, проф. Журавська І. М.

Дана робота присвячена розробці мобільного ігрового застосунку для моніторингу здоров'я на платформі Unity з використанням біомедичних показників із вбудованих сенсорів фітнес-браслета у якості «ігрової валюти».

Об'єктом роботи є процес розробки ігрового застосунку для моніторингу здоров'я користувачів у жанрі Casual на платформі Unity.

Предметом роботи є особливості розробки мобільних ігрових застосунків на платформі Unity з використанням пристроїв, що працюють за технологією Bluetooth Low Energy.

Метою кваліфікаційної роботи є гейміфікація біомедичних показників у вигляді «ігрової валюти» для казуальної гри на платформі Unity та мотивування гравців до здійснення фізичної активності під час гри.

Кваліфікаційна робота складається зі вступу, чотирьох розділів, висновків та переліку джерел посилання.

У вступі описується актуальність теми, визначається мета, об'єкт, предмет роботи та необхідні завдання.

У першому розділі проводиться аналіз предметної галузі, аналогів застосунків для моніторингу здоров'я, їх переваг і недоліків, формується специфікація вимог до програмного забезпечення, що розробляється.

Другий розділ містить детальний опис ігрового процесу застосунку із наведенням сценаріїв використання, діаграм діяльності та інших засобів конструювання програмного забезпечення.

Третій розділ містить діаграму класів, особливості розробки мобільного ігрового застосунку, та організацію Unity-проєкту. Важлива роль відведена опису технології Bluetooth Low Energy та особливостям її використання при взаємодії персональних пристроїв контролю біометричних показників та розроблюваного мобільного ігрового застосунку.

Четвертий розділ містить опис розробки ігрового застосунку (кодування), скриншоти коду, ігрового дизайну та локацій, результати тестування гри.

Висновки містять аналіз проведених робіт та результатів.

КРБ викладена на 65 с., містить 4 розділи, 8 табл., 41 рис., 3 дод., 20 джерел.

Ключові слова: *казуальна гра, ігровий рушій Unity, ігрова валюта, біомедичні показники, Bluetooth Low Energy, фітнес-браслет.*

ABSTRACT
of the Bachelor`s Thesis
"Mobile health monitoring application as a casual game based on the Unity platform"

Student: Afonin Yurii
Supervisor: D.Sc. (Eng.), Professor, Zhuravska I. M.

This work is dedicated to developing a mobile health monitoring game application on the Unity platform using health characteristics from embedded fitness-bracelet sensors as a "game currency".

The object of work: the mobile application development process for players` health monitoring as a casual game on the Unity platform.

The subject of work: core features of creating mobile gaming apps using Unity and Bluetooth Low Energy technology.

Objective: health characteristics gamification in the form of "game currency" for the casual-style game in the Unity platform and motivating players to do physical activity while playing.

The qualification work of the bachelor consists of an introduction, four chapters, conclusions, and a list of sources references.

The introduction defines the relevance of the topic, sets up the object, work objective, subject of research, and brief tasks overview.

The first chapter determines the analytics part of researched app subject area and includes a comparison with similar health monitoring applications, also their advantages and disadvantages. The last part of the chapter describes the formation of requirements specifications for the developed software.

The second chapter includes a detailed description of the gameplay process, including use cases, activity diagrams, and other software design tools.

The third chapter includes a class diagram, the features of mobile game application development, and the organization of the Unity project. The key role is devoted to Bluetooth Low Energy technology description and its important features for mobile game application development.

The fourth chapter contains the description of game application development (coding), screenshots of the code, game design, and locations, as well as the results of game testing.

The conclusions analyze the work and obtained results.

The qualification work of the bachelor is presented on 65 pages, it contains 4 chapters, 8 tables, 41 figures, 3 appendices, 20 sources in the list of references.

Keywords: *casual game, Unity game engine, in-game currency, health characteristics, Bluetooth Low Energy, fitness bracelet.*

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ПЗ	–	програмне забезпечення
ЦА	–	цільова аудиторія
ОС	–	операційна система
ТЗ	–	технічне завдання
API	–	application programming interface
BLE	–	Bluetooth Low Energy
UML	–	unified modeling language
ATT	–	attribute protocol
GATT	–	Generic Attribute Profile
JNI	–	Java Native Interface
UI	–	user interface
UX	–	user experience

ВСТУП

Розробка мобільних ігрових застосунків з імплементацією функції моніторингу здоров’я є одним з перспективних напрямків гейміфікації поточної неігрової діяльності людини задля сприяння емоційній залученості та підвищення мотивації користувачів для досягнення певних результатів в якості власного образу життя. Використання платформи Unity, моделювання та проєктування основних механік та складових зазначених програмних продуктів підвищує ефективність та якість останніх.

Актуальність теми зумовлена важливістю моніторингу та підтримання здоров’я людини у вік стрімкого розвитку інформаційних технологій. Ці складові безумовно повинні бути першими пріоритетами людини, однак через зайнятість, роботу, стрес, інші чинники більшість із нас приділяють йому надто мало часу. Створення програмного забезпечення, що допомагатиме користувачу взаємодіяти з власними біомедичними показниками у зручному та інтерактивному вигляді є одним із затребуваних засобів, що допоможуть покращити якість здоров’я [7].

Мобільні ігрові застосунки стали невід’ємною частиною життя для багатьох людей у всьому світі. Окрім розваг, ігри допомагають дітям розвиватись, опановувати нові навички, знання про світ, відволікають дорослих від проблем, розвивають мислення та навіть допомагають школярам і студентам у вивченні певних дисциплін. Саме тому, таке програмне забезпечення найбільше впливає на звички як підлітків, так і дорослих. Використання ігрових застосунків для моніторингу біомедичних показників є чудовим засобом для мотивування гравців до фізичної активності та підтримання власного здоров’я [4; 6].

Об’єктом роботи є процес моделювання та розробки ігрового застосунку для моніторингу біомедичних показників користувачів з використанням вбудованих сенсорів фітнес-браслета.

Предметом роботи є особливості проєктування мобільних ігрових застосунків на платформі Unity з використанням технології Bluetooth Low Energy.

Метою є гейміфікація процесу отримання біомедичних показників користувача у вигляді «ігрової» валюти за рахунок розробки казуальної гри на платформі Unity та мотивування гравців до здійснення фізичної активності під час гри. Для виконання поставленої мети необхідно виконати такі **завдання**:

– аналіз предметної області;

– визначення функцій програмного забезпечення (ПЗ), специфікації вимог та сценаріїв використання;

– проєктування застосунку, створення UML-діаграм;

– обґрунтування та використання технології Bluetooth Low Energy для зв'язку застосунку із фітнес-браслетом;

– створення мобільного ігрового застосунку з функцією моніторингу здоров'я та прив'язкою до фітнес-браслета.

Практичне застосування: ігровий застосунок можна використовувати для різноманітних цілей – для розваг, відпочинку, як мотиватор для здійснення фізичної активності, у сфері спорту, навчанні тощо [10]. З точки зору програмування застосунок можна використовувати для освітніх цілей як приклад для розбору особливостей використання технології BLE у мобільних ігрових застосунках.

КРБ викладена на 66 сторінках, містить 41 ілюстрацію, 8 таблиць, 3 додатки та 20 використаних джерел.

Апробація результатів кваліфікаційної роботи відбулася під час XI Міжнародної науково-практичної конференції «Free and Open Source Software – FOSS'2023» (Харків, 14–16 лютого 2023 р.) та Всеукраїнської науково-практичної конференції молодих вчених, аспірантів і студентів «Інформаційні технології та інженерія» (Миколаїв, 07–10 лютого 2023 р.).

Публікації. Основні положення та результати кваліфікаційної роботи бакалавра опубліковані у збірниках матеріалів Міжнародної науково-практичної конференції «FOSS’2023» [1] та Всеукраїнської науково-практичної конференції «Інформаційні технології та інженерія» [2].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ВИМОГ

1.1 Огляд застосунків-аналогів моніторингу здоров’я

Першочерговим завданням перед конструюванням будь-якого програмного забезпечення є аналіз предметної галузі та пошук застосунків-аналогів. Такий аналіз є невід’ємною складовою для успішного проєктування будь-якого застосунку, так як це допомагає визначити потрібний функціонал, основні недоліки та переваги конкурентів. Для аналізу аналогів було обрано застосунки Google Fit (табл. 1.1), Zepp Life (табл. 1.2) та Walkr (табл. 1.3).

Google Fit

Застосунок створений для відстеження фітнесу та здоров’я на мобільній платформі Android (табл. 1.1) [11]. Має широкий вибір характеристик для відстеження здоров’я та інтегрується з іншими застосунками. Недоліком виступає відсутність початкової інтерактивної зацікавленості для звичайного користувача, тобто для користування застосунком треба вже мати достатню умотивованість (наприклад, для спортсменів). Інтерфейс застосунку на рис. 1.1.

Таблиця 1.1 – Опис застосунку «Google Fit»

Назва	**Google Fit**
Виробник:	Google LLC
Архітектура	Мобільний застосунок для Android
Мова реалізації	Java, Kotlin
Функції	1. Моніторинг активності, кроків та рівня активності. 2. Збір інформації про пульс, сон та інші параметри 3. Інтеграція з іншими додатками для здоров'я.
Переваги	1. Безкоштовний, інтегрується з іншими додатками, 2. Має інтуїтивний інтерфейс.

Кінець таблиці 1.1

Недоліки	1. Деякі користувачі відмічають проблеми з точністю вимірювань та синхронізацією даних з іншими додатками. 2. Відсутність інтерактивної складової для зацікавлення простих користувачів проблемою моніторингу та підтримання власного здоров’я
Вебсайт	www.google.com/fit

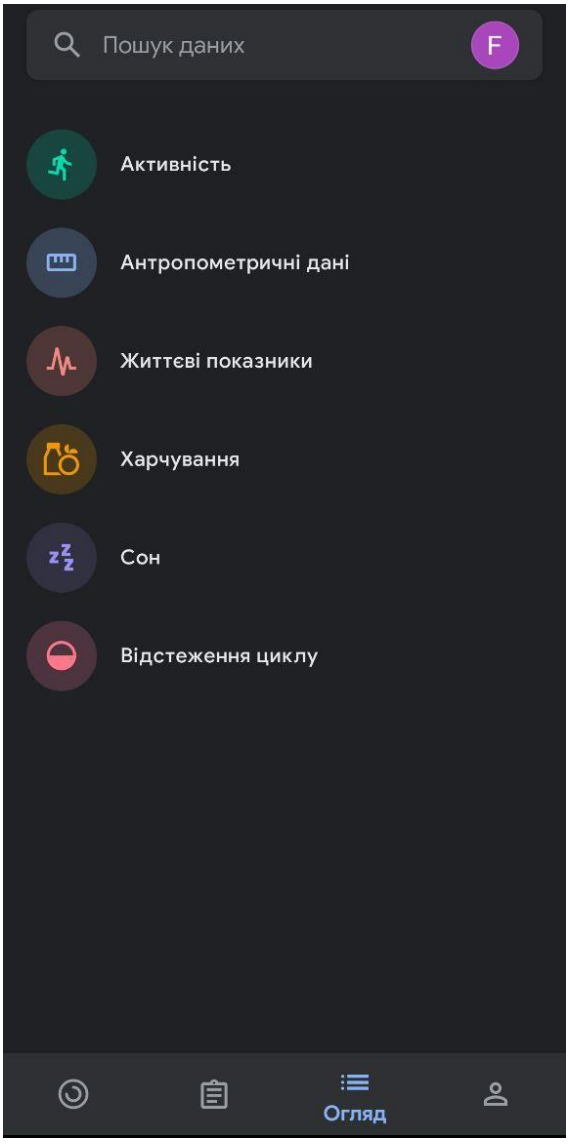


Рисунок 1.1 – Вигляд інтерфейсу «Google Fit»

Zepp Life

Застосунок є своєрідним аналогом-конкурентом Google Fit, але добре працює лише із лінійкою девайсів Mi (браслети, смарт-годинники) від компанії Xiaomi. Має можливість інтеграції із іншими застосунками, однак набагато гіршу API-документацію (табл. 1.2) [12].

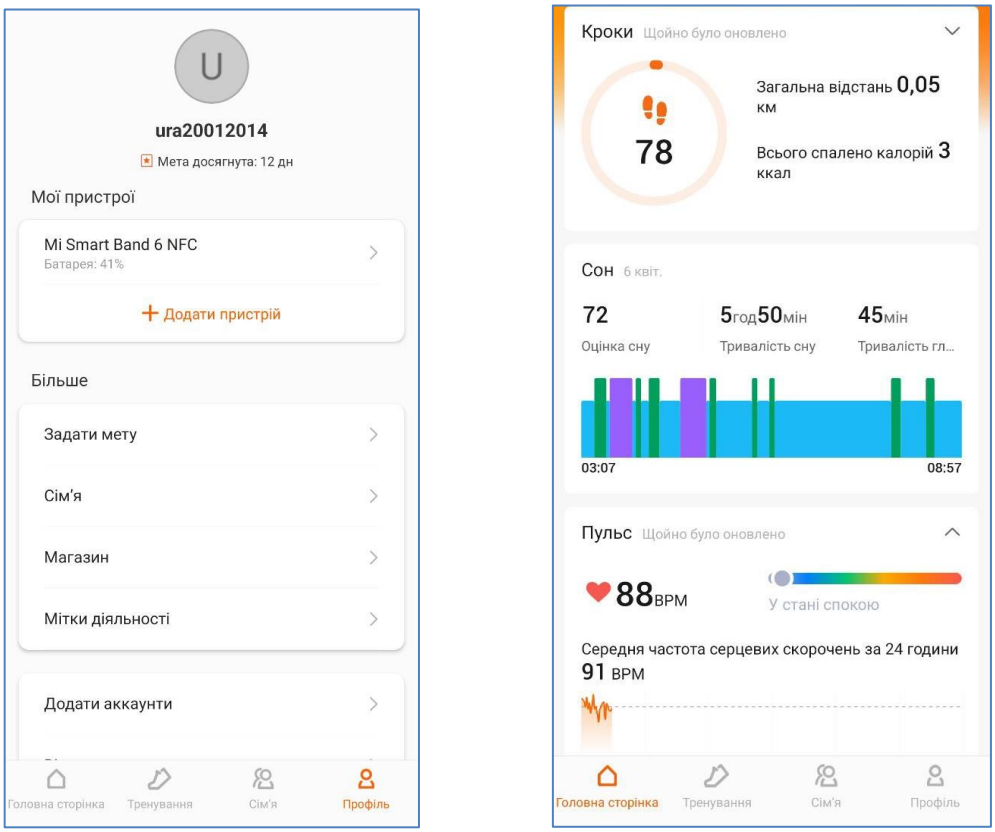
Хоч цей застосунок і має кращий функціонал від конкурента, але має той же недолік – відсутність початкової інтерактивної зацікавленості для звичайного користувача та орієнтацію на мотивованих споживачів (спортсмени, люди, що займаються фітнесом, тощо). Вигляд інтерфейсу застосунку наведено на рис. 1.2.

Таблиця 1.2 – Опис застосунку «Zepp Life»

Назва	**Zepp Life**
Виробник:	Huami Inc.
Архітектура	Мобільний застосунок для Android
Мова реалізації	Java, Kotlin
Функції	1. Моніторинг фізичної активності та кроків. 2. Вимірювання пульсу та кисню в крові. 3. Аналіз якості сну. 4. Збір інформації про рівень стресу. 5. Інтеграція зі смарт-годинниками та іншими пристроями для збору даних.
Переваги	1. Точність вимірювань, 2. Широкий спектр функцій для моніторингу здоров'я, 3. Елегантний дизайн.

Кінець таблиці 1.2

Недоліки	1. Деякі користувачі відмічають проблеми зі синхронізацією даних. 2. Зазвичай потрібна наявність браслета/смарт годинника. 3. Відсутність інтерактивної складової для зацікавлення простих користувачів проблемою моніторингу та підтримання власного здоров'я.
Вебсайт	www.zepp.com/life.html



а) б)

Рисунок 1.2 – Інтерфейс застосунку «Zepp Life»

Walkr

Цей застосунок ігровий, він вирізняється від попередніх аналогів. Застосунок створений для досягнення інтерактивності та привернення уваги широкої цільової аудиторії (ЦА). За допомогою основної характеристики – крокоміру – ігровий застосунок мотивує гравця для здійснення фізичної активності та «отримання» енергії за пройдені кроки для розблокування нових ігрових локацій (табл. 1.3).

Вигляд інтерфейсу застосунку наведено на рис. 1.3.

Таблиця 1.3 – Опис застосунку «Walkr»

Назва	**Walkr**
Виробник:	Fourdesire
Архітектура	Мобільний застосунок для Android та iOS
Мова реалізації	Відсутня інформація, однак ймовірно Java, Kotlin та Swift
Функції	1. Відстеження кроків та дистанції, пройденої користувачем. 2. Можливість створення власних фітнес-цілей та їх відстеження. 3. Мотиваційні елементи, такі як гейміфікація та нагороди за досягнення цілей. 4. Інтеграція із застосунком здоров'я Apple Health та Google Fit.
Переваги	**1.** Легкий у використанні та налаштуванні. інтуїтивний інтерфейс. **2.** Гейміфікація дозволяє зробити фітнес-процес більш цікавим та мотивуючим.

Кінець таблиці 1.3

Недоліки	1. Деякі функції доступні тільки у преміум-версії. 2. Не завжди точно відстежує відстань та кількість кроків. 3. Не містить багатьох додаткових функцій, що присутні у конкурентних застосунках.
Вебсайт	www.fourdesire.com/walkr

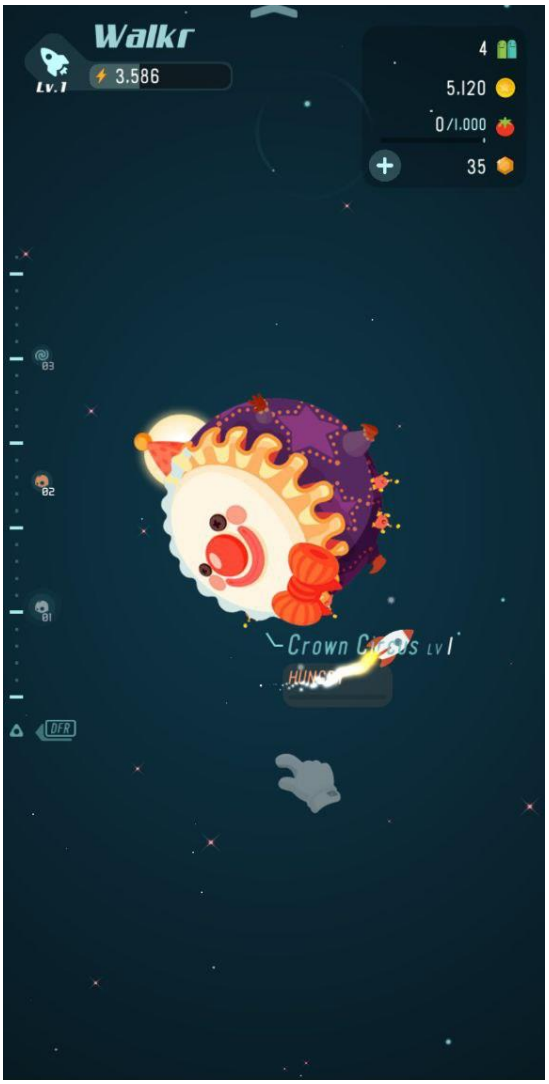


Рисунок 1.3 – Інтерфейс ігрового застосунку «Walker»

1.2 Аналіз застосунку, що розробляється

Створення застосунку моніторингу здоров’я у вигляді казуальної гри призначено для мотивування гравців до фізичної активності та цікавості до власного здоров’я шляхом проходження інтерактивних квестів, прогресії шляхом покращення, відкриття нових локацій та використанням біомедичних даних з фітнес-браслета як «ігрової» валюти. Гравець одночасно отримує задоволення від розваги та проводить час із користю для себе, виконуючи фізичну активність (ходьбу) та отримуючи інформацію про пульс та кількість пройдених кроків і спалених калорій.

Застосунок має підтримувати однокористувацький (клієнтський) режим роботи та бути сумісний із смартфонами, що використовують операційну систему Android (вище версії 6.0) та підтримують технологію Bluetooth Low Energy, що використовується для зв’язку і передачі даних з фітнес-браслета у застосунок (табл. 1.4).

Таблиця 1.4 – Опис застосунку, що розробляється

Функції	1) підтримка BLE-зв’язку із фітнес-браслетом; 2) відстеження кроків; 3) моніторинг пульсу; 4) моніторинг спалених калорій; 5) мотиваційні елементи, такі як гейміфікація та нагороди за досягнення квест-цілей; 6) ігровий рівень; 7) ігровий магазин; 8) прогресія гравця; 9) квестова система; 10) обмін пройдених кроків на ігрову валюту;

Кінець таблиці 1.4

Функції	11) обмін спалених калорій на ігрову валюту; 12) можливість грати у гру навіть без фітнес-браслета (як звичайна казуальна гра)
Користувачі	1) користувач-гравець
Сценарії роботи	1. Користувач встановлює застосунок з метою отримати ігровий досвід та задоволення, йому пропонується надати доступ до фізичних даних та Bluetooth. 2. Користувач запускає гру та опиняється в ігровому меню. 3. Користувач натискає «почати гру», де йому пропонується знайти власний фітнес-браслет. 4. Користувач знаходить та підключає браслет до гри, користувачу показується ігровий інтерфейс. 5. Користувач розпочинає гру та проходить квест-завдання, отримуючи ресурси (монети та енергію). 6. Користувач здійснює фізичну активність, кількість кроків на візуальному лічильнику збільшується, кількість спалених калорій збільшується, гравець може обміняти кроки на енергію у магазині для відкриття (прискорення) нових рівнів.

1.3 Специфікація вимог до ПЗ

Не менш важливими завданнями виступають складення специфікації вимог, функцій та опис основних варіантів використання системи, що проєктується, у вигляді use-case діаграми. Створення діаграм класів, діаграм використання та розгортання також є ключовими завданнями для якісної розробки програмного забезпечення. Нижче наведено специфікацію, діаграми включені до другого та третього розділів КРБ.

Призначення застосунку, для якого розробляється програмне забезпечення

Призначенням застосунку є гейміфікація основних фізичних характеристик (даних) фітнес-браслета завдяки розробці ПЗ мобільного застосунку моніторингу здоров’я у вигляді казуальної гри.

Погодження, що ухвалені в програмній документації

Було погоджено, що для створення ПЗ та його роботи буде використано такі технології: платформа Unity у якості ігрового рушія та відповідно мова програмування C# у якості скриптової мови рушія, бездротова технології Bluetooth Low Energy.

ЗАГАЛЬНИЙ ОПИС

Сфера застосування

Дане ПЗ не має суттєвих обмежень у сфері споживання, можна застосовувати у звичайному житті, сфері спорту, розваг, навчання, для мотивації та моніторингу фізичної активності.

Характеристика користувачів (гравців)

Основні характеристики гравців: наявність смартфону з підтримкою бездротової технології BLE, наявність фітнес-браслету з підтримкою BLE (опціонально, однак без браслету основні функції моніторингу здоров’я не будуть працювати).

Загальна структура і склад системи

Основна структура: ігровий застосунок

Загальні обмеження

Обмеження для роботи: наявність бездротової технології Bluetooth Low Energy.

ОСНОВНІ ФУНКЦІЇ ІГРОВОГО ЗАСТОСУНКУ МОНІТОРИНГУ ЗДОРОВ'Я

Підтримка BLE-зв'язку із фітнес-браслетом

Опис функції

Ця функція дає можливість отримувати фізичні дані (пульс, кроки, калорії, дистанцію) з фітнес-браслета і використовувати її в застосунку.

Вхідна і вихідна інформація

Вхідна – дані про навколишні BLE-девайси (браслети); підключення Bluetooth та геолокації; дозволи на використання Bluetooth та фізичних даних.

Вихідна – регулярно оновлювані дані з обраного браслета.

Функціональні вимоги

Знаходження найближчих девайсів через технологію Bluetooth Low Energy; підключення до девайсів; запит і отримання даних з девайсів з використанням ATT протоколу.

Функція мотиваційних елементів, нагород, гейміфікації фізичних характеристик

Опис функції

Функція надає інтерактивну складову ігровому застосунку, затримує увагу гравця, мотивує до здійснення фізичної активності (через завдання, нагороди).

Вхідна і вихідна інформація

Вхідна – фізичні дані з фітнес-браслета, дані для завдань.

Вихідна – дані про нагороду.

Функціональні вимоги

Використання даних з фітнес-браслета для відображення на екрані; використання даних (кроків, калорій, дистанції) для обміну на енергію; показ завдань, квестів, нагород гравцю; отримання нагороди за виконання завдань.

ВИМОГИ ДО ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ

Джерела і зміст вхідної інформації (даних)

В даному ПЗ джерелами вхідної інформації є користувач (надання дозволів, підключення нативних сервісів – Bluetooth, геолокація (для знаходження найближчих девайсів)) та фітнес-браслет (за наявності, дає доступ до основних даних – пульс, кроки, калорії, дистанція).

Нормативно-довідкова інформація

Вимоги відсутні.

Вимоги до способів організації, збереження та ведення інформації

Отримання даних з фітнес-браслета відбувається через протокол ATT (GATT-профіль) бездротової технології BLE.

ВИМОГИ ДО ТЕХНІЧНОГО ЗАБЕЗПЕЧЕННЯ

Вимоги до технічного забезпечення мають певні обмеження у вигляді підтримки смартфоном технології BLE та відповідного рівня Android API (вище версії 6.0). Наявність фітнес-браслету із підтримкою BLE та сумісність (версії) зі смартфоном (дослідження проведено на основі фітнес-браслету Xiaomi Mi Band 6).

ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Архітектура програмної системи

Архітектура включає питомі для ігрового застосунку частини.

Системне програмне забезпечення

Застосунок має бути побудований з використанням платформи Unity в якості ігрового рушія. Процес отримання даних з фітнес-браслету має відбуватись через протокол ATT технології BLE. Для зв’язку із нативним плагіном для роботи з BLE використовується вбудована в Unity бібліотека JNI

(Java Native Interface). Для під'єднання та отримання даних від фітнес-браслета використовується open-source бібліотека Unity Android Bluetooth Low Energy.

Мова і технологія розробки ПЗ

ПЗ має розроблюватись за допомогою мови C# на платформі Unity.

ВИМОГИ ДО ЗОВНІШНІХ ІНТЕРФЕЙСІВ

Інтерфейс має задовольняти вимоги UI/UX-дизайну для легкого розуміння користувачем ігрового процесу застосунку. Ігровий інтерфейс має містити лічильники основних ресурсів (кроків, енергії, грошей/монет, калорій, пройденої дистанції), кнопку виходу до пауз-меню. Адаптивні та зручні панелі налаштувань, меню та ігрового магазину.

Апаратний інтерфейс

Мобільний девайс користувача (смартфон) на базі ОС Android та сумісний фітнес-браслет.

Програмний інтерфейс

У якості програмного інтерфейсу використовується вбудована у платформу Unity UI Canvas компоненти для створення адаптивного користувацького інтерфейсу, Event System для обробки користувацького вводу. Для розробки ПЗ використовується IDE Rider та Unity Editor, що забезпечують легкість та швидкість процесу розробки.

Комунікаційний протокол

Застосунок передбачає використання протоколу ATT – протоколу для спілкування між BLE-пристроями.

ВЛАСТИВОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Доступність

ПЗ є доступним для користувачів, що грають у казуальні ігри та бажають отримати ігровий досвід від гейміфікації моніторингу основних фізичних характеристик за наявністю апаратного інтерфейсу і підтримки ним BLE.

Супроводженість

Застосунок не потребує супроводженості.

Переносимість

ПЗ може працювати на ОС Android. За потреби перенесення ПЗ на іншу систему (Windows, iOS тощо) доведеться використовувати додаткове апаратне забезпечення (у випадку Windows зовнішні BLE-модулі) та інші нативні бібліотеки/плагіни для зв’язку із фітнес-браслетами (для iOS на Swift або Objective-C). Загалом Unity дозволяє кросплатформність застосунку, однак, для кожного виду платформи є свої особливості та нюанси.

Продуктивність

Продуктивність ПЗ залежить від швидкості оновлення даних браслету та продуктивності смартфону.

Надійність

Персональні дані користувача та дані з фітнес-браслету не зберігаються на сервер та не відправляються в мережу, однак зберігаються локально для збереження прогресу користувача (ресурси у вигляді загальної кількості пройдених кроків, дистанції, спалених калорій, енергії, монет під час ігрової сесії).

Безпека

Застосунок не взаємодіє з мережею.

Висновки до розділу 1

В першому розділі проведено аналітичний огляд застосунків-аналоги з функціями моніторингу здоров’я, покращено навички аналізу функціоналу аналогічних систем у сфері моніторингу здоров’я. Виокремлено переваги та недоліки аналізованих систем, проведено детальний аналіз застосунку, що розробляється, визначено функції, сценарії використання, складено специфікацію вимог (ТЗ) до застосунку.

2 МОДЕЛЮВАННЯ ЗАСТОСУНКУ

2.1 Діаграма сценаріїв використання (use case)

Use Case або сценарій використання – перелік дій або ж сценарій, згідно з яким користувач взаємодіє із проєктованим застосунком, виконуючи будь-яку дію, що дозволена функціоналом застосунку.

Даний ігровий застосунок розробляється у гіперказульному жанрі, що є надзвичайно популярним. Мобільні гіперказуальні ігри мають широку ЦА та інтуїтивні механіки керування, що дає можливість гравцям легко та швидко розібратись із функціоналом застосунку.

Основна задумка гри, що проєктується, – розбудова невеликого містечка (будування певних локацій) за допомогою енергії та ігрової валюти (монет). Енергія є відновлювальним ресурсом (відновлюється з певним часом), монети можна отримати за проходження квестів чи щоденний дохід із побудованих закладів/будівель. Важливою складовою є використання фітнес-браслету, а саме його даним, які будуть використовуватись як ще один варіант отримання (обміну) на енергію та монети.

Короткий Use Case

Користувач встановлює ігровий застосунок на мобільний телефон. Мобільний пристрій запускає застосунок та запрошує у користувача доступ до даних про фізичну активність користувача. Користувач підтверджує надання застосунку права доступу та переходить у вітальне меню гри та починає ігровий процес.

Поверхневий Use Case

Головний сценарій (успішний)

Користувач встановлює ігровий застосунок на мобільний телефон на системі Android. Мобільний пристрій запускає застосунок та запрошує у користувача доступ до даних про фізичну активність користувача. Користувач підтверджує надання застосунку права доступу та переходить у

вітальне меню гри. Меню вітає користувача та пропонує пройти невелике ознайомлення із основним функціоналом гри. Гравець погоджується пройти ознайомлення. Гравцю пропонується знайти і підключити фітнес-браслет. Гравець успішно знаходить та підключає браслет. Опісля демонструється основний екран із наявними ресурсами (енергія, монети), головним персонажем та завданнями і функціоналом застосунку, де пояснюється функція моніторингу кроків, пульсу та калорій. По закінченню невеликого туторіалу гравцю пропонується почати виконання завдань. Гравець починає виконання завдання, таким чином отримуючи ігровий досвід користування функціями моніторингу.

Успішно закінчивши перше завдання гравець отримує винагороду у вигляді монет. Отримані монети та енергію користувач може використати на відкриття нової локації. Гравець відкриває нову локацію, запускається таймер до закінчення відкриття. Для пришвидшення відкриття нового рівня користувачу пропонується обміняти пройдені кроки у першому завданні на енергію.

Use Case діаграма наведена у додатку А.

Альтернативні сценарії:

1) гравець відмовляється надати доступ до даних про фізичну активність. Застосунок повідомляє, що без доступу гра буде діяти в режимі звичайної казуальної гри без моніторингу біомедичних даних з можливістю увімкнути режим моніторингу в налаштуваннях. Гравець переходить до головного меню;

2) гравець надав доступ до даних про фізичну активність. Після переходу у головне меню гравець відмовляється проходити ознайомлення з основним функціоналом, йому одразу пропонується пройти завдання. Гравець починає проходити завдання;

3) гравець не дав доступ до даних про фізичну активність. Після переходу у головне меню гравець відмовляється проходити ознайомлення з

основним функціоналом та не починає проходити завдання. Гравець заходить у налаштування та вмикає моніторинг активності;

4) гравець надавши доступ до даних про фізичну активність переходить до головного меню, відмовляється пройти ознайомлення та виходить із гри.

2.2 Огляд технологій

Для створення ігрового застосунку обрано такий стек технологій:

– мови програмування C# (основна клієнтська мова платформи Unity) та Java (додаткова, нативна мова Android-платформи);

– ігровий рушій (платформа) Unity;

– Bluetooth Low Energy як технологія для зв'язку із фітнес-браслетом;

– open-source плагіни для підключення BLE-технології до застосунку та отримання даних (рис. 2.2).

Рисунок 2.1 – Основний стек технологій

Рушій Unity є однозначним флагманом для створення ігор. Unity Technologies невпинно працюють над вдосконаленням власного продукту та пропонують передові технології для повного циклу розробки ігрових застосунків під різні платформи. Попри це саме розробка мобільних ігор стала найбільшим джерелом попиту рушія. Більш ніж 70 % популярних ігор на мобільній платформі були створені саме на його базі. Вільний доступ, кросплатформність, ґрунтовна та глибока документація, наявний функціонал, аналітика, монетизація, величезна спільнота розробників та готових інструментів дають Unity почесне лідерство серед своїх конкурентів [1].

Через такі переваги і було обрано саме цей рушій. Мова програмування C# за замовчування є скриптовою мовою Unity, тому теж йде в обраний стек.

Java як нативна мова Android входить у стек як додаткова для написання нативного функціоналу до якого Unity не має доступу, але може «спілкуватись» у C# коді через JNI (Java Native Interface), що за замовченням включена в Unity (рис. 2.2).

```
{
    AndroidJavaClass librarySingleton = new AndroidJavaClass("com.velorexe.unityandroidble.UnityAndroidBLE");
    _bleLibrary = librarySingleton.CallStatic<AndroidJavaObject>(methodName: "getInstance");
}
```

Рисунок 2.2 – Виклик Java-методу “getInstance” із C# коду

Оскільки застосунок, що конструюється використовуватиме дані з фітнес-браслету, необхідно визначити технологію, обґрунтувати її використання та впровадити її як частину програмного забезпечення.

У роботі було кілька разів згадано про технологію BLE, але детально не пояснено, яким чином вона використовується для передачі даних, тому нижче наведено теоретична інформація, як саме ця технологія працює та її основні поняття.

Bluetooth Low Energy (BLE) – бездротова технологія, яка дозволяє обмінюватись даними між різними пристроями з низьким рівнем енергоспоживання. Ця технологія широко використовується в різних сферах, таких як IoT, здоров'я та фітнес, автомобільна промисловість та багато інших.

Generic Attribute Profile (GATT) – це протокол, який використовується для передачі даних між BLE-пристроями. GATT визначає ієрархію даних, які передаються між пристроями, включаючи Services та Characteristics.

Services – це набір атрибутів, які описують функціональні можливості BLE-пристрою. Кожен сервіс містить декілька характеристик, які можуть бути доступні для зчитування або запису.

У більшості сучасних пристроїв для фітнесу та моніторингу здоров’я за замовченням вбудований *Heart Rate Service* із характеристикою частоти ударів серця (пульс).

Characteristics – це конкретні атрибути в межах сервісу, які містять деяку інформацію. Кожна характеристика містить у собі декілька полів, включаючи значення, дозволи на зчитування або запис, та рівень доступності. Наприклад, одна з характеристик браслета Xiaomi Mi Smart Band 6 містить такі дані:

– кількість пройдених кроків за добу;
– кількість спалених калорій за добу;
– пройдена кількість метрів за добу.

Взаємодія між BLE-пристроями відбувається через обмін повідомленнями між характеристиками пристроїв. Кожен BLE-пристрій може мати кілька сервісів, і кожен сервіс може мати декілька характеристик. Така ієрархія дозволяє розширити можливості обміну даними між пристроями і виконати широкий спектр завдань в різних сферах, що використовують BLE. Нижче наведена схема такого обміну на прикладі смартфону, фітнес-браслета та встановленого застосунку (рис. 2.3).

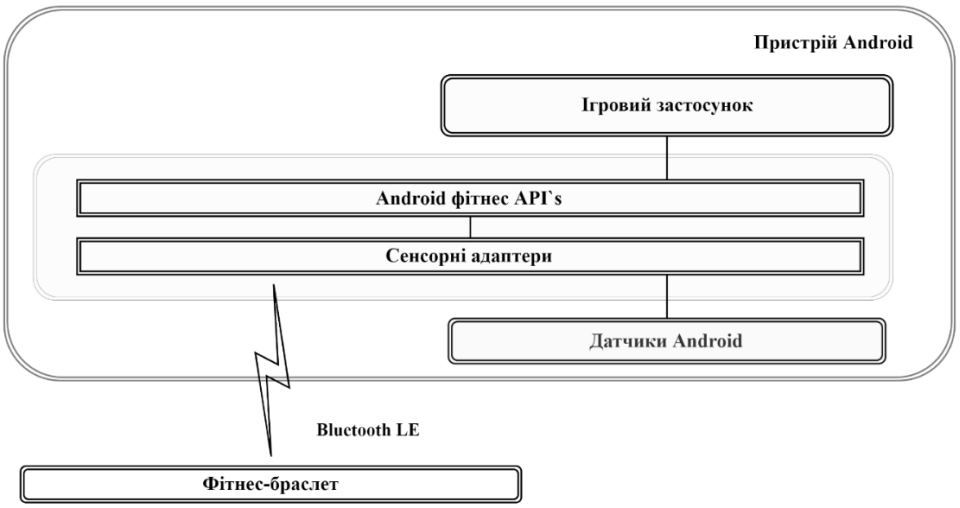


Рисунок 2.3 – Схема обміну даних між застосунками та пристроями

Для функціонування такої передачі даних було знайдено open-source бібліотеку, яка дозволяє налаштувати зв’язок за допомогою BLE у Unity, класи наведено у діаграмах.

2.3 Діаграми діяльності

Ігровий застосунок із функцією моніторингу здоров’я повинен мати зручний і зрозумілий інтерфейс, оскільки ним можуть користуватись гравці різного віку. Щоб відображувати основні дії, які може здійснити гравець зазвичай використовують діаграми діяльності, що є аналогом блок-схем і візуально описує логічну послідовність дій користувача та можливих розгалужень відповіді системи. Створено кілька діаграм діяльності, що наведені нижче:

1) *Покупка енергії за отримані ресурси* (рис. 2.4). Дана діаграма відображає діяльність «**Покупка енергії**». Коли гравець заходить у магазин йому відображається панель із можливими покупками (в даному випадку енергія для гри). Гравець може придбати додаткову енергію за ігрову валюту (гроші), пройдені кроки або спалені калорії, що зчитуються з фітнес-браслета (якщо він під’єднаний до гри). Якщо достатньо ресурсів для придбання – оновлюється кількість наявних ресурсів, додається куплена енергія та оновлюється UI (лічильники ресурсів). Після цього можна повторити цикл покупки, аж доки не закінчаться валюта/ресурси або гравець не захоче вийти;

2) *Пошук Bluetooth LE пристрою (браслета)* (додаток Б). Ця діаграма відображає діяльність, коли гравець надав дозвіл на використання застосунком BLE та фізичними даними, то відкривається панель для пошуку найближчих пристроїв для підключення. Якщо такі були знайдені, то вони з’являться на екрані; у іншому випадку будуть виведені повідомлення про незнаходження чи помилку.

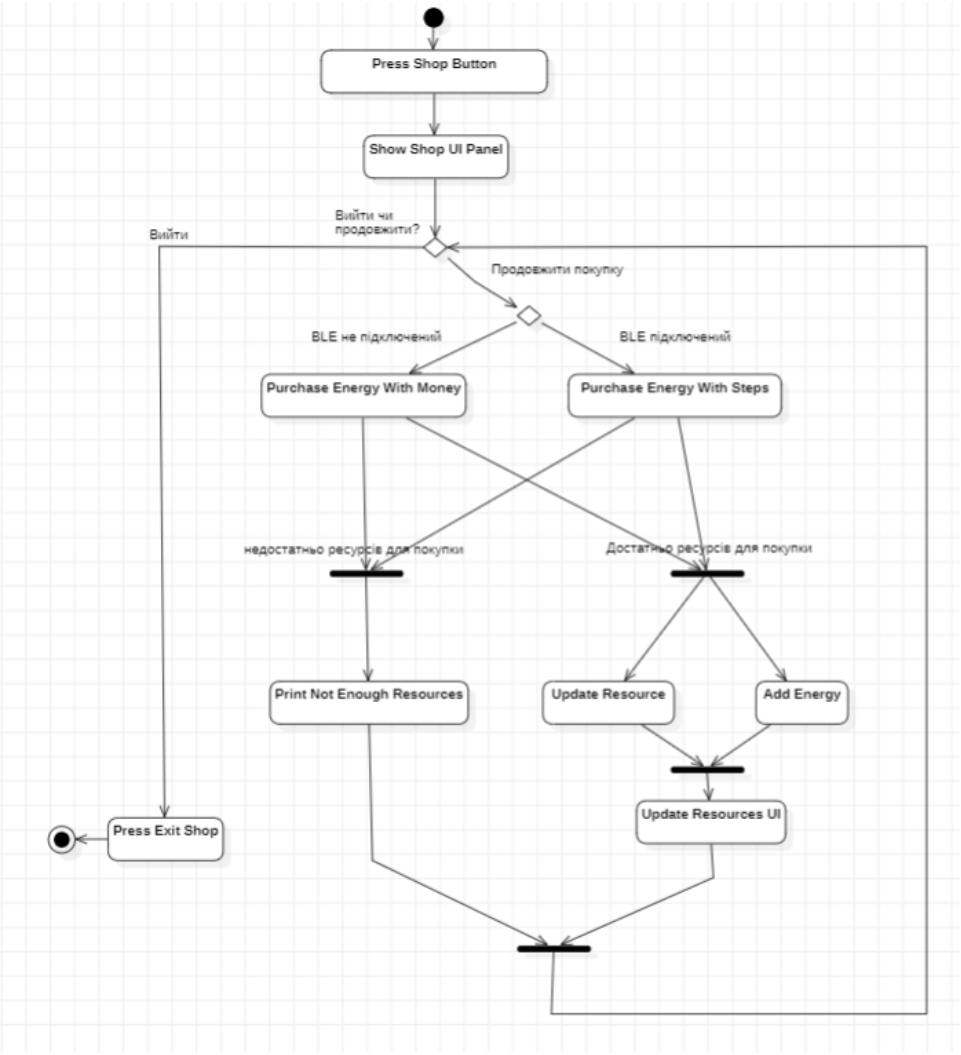


Рисунок 2.4 – Діаграма діяльності покупки енергії в магазині за кроки або ігрову валюту

Діаграми було створені за допомогою спеціалізованого програмного забезпечення моделювання та конструювання Star UML для демонстрації деяких процесів дії користувача застосунку.

2.4 Діаграми послідовності та комунікацій

Для детальнішого розуміння яким чином гравець взаємодії із застосунком і як саме система реагує на дії користувача було розроблено кілька діаграм взаємодії та послідовності (рис. 2.5).

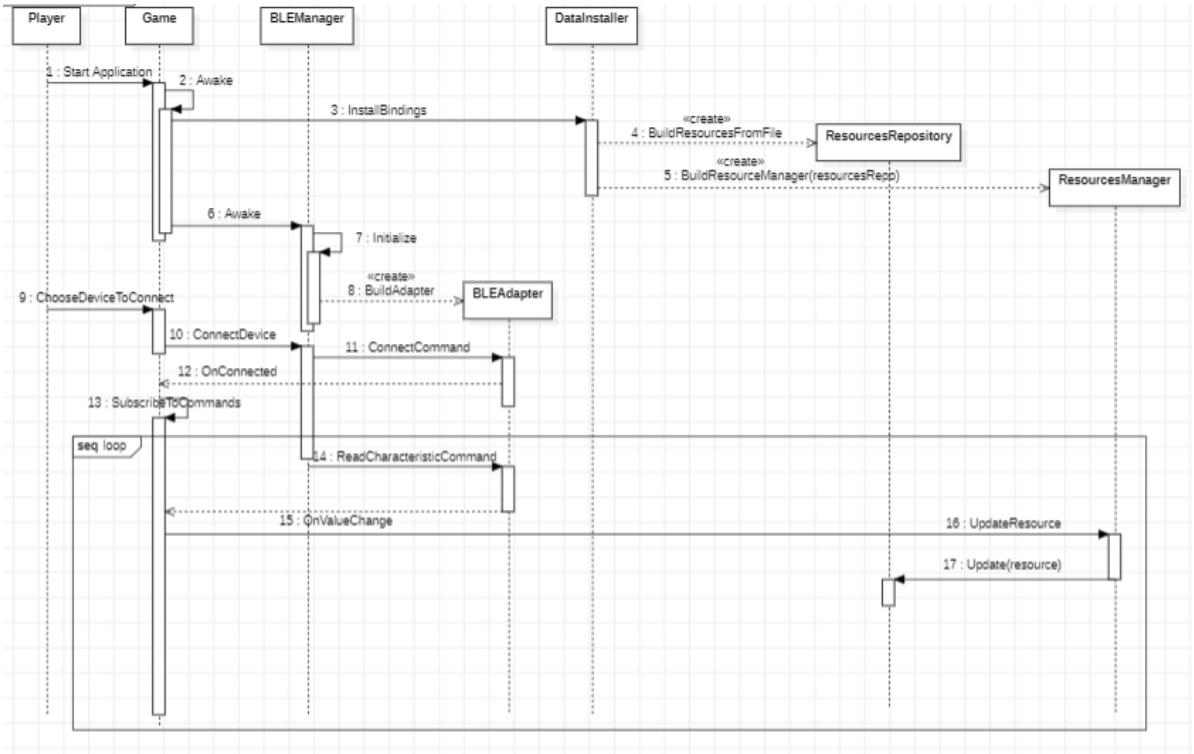


Рисунок 2.5 – Діаграма послідовності основного ігрового циклу застосунку

Це поведінкові (англ. behavior) діаграми, що показують основні класи, події та спілкування між ними під чай дій користувача, це спрощує створення написання кодової бази програмного забезпечення та показує конкретну дію користувача.

Після ініціалізації застосунку (відкритті гри гравцем), гра проводить ініціалізацію усіх потрібних збережених даних у класі DataInstaller, який в свою чергу ініціалізує репозиторій із ресурсів та ресурсний менеджер.

Далі відбувається ініціалізація Bluetooth Low Energy Manager, що відповідає за зв’язок із Bluetooth-пристроями через BLE Adapter. Коли користувач, надавши доступ до необхідних доступів, шукає фітнес-браслет, то BLE Manager надсилає команду на пошук, і повертає користувачу знайдені пристрої поблизу, після чого гравець підключається до пристрою і відбувається підписка на отримання даних з браслета при їх зміні. При зміні цих даних оновлюється UI, де гравець бачить свій поточний пульс, кількість пройдених кроків, «спалених» калорій та інші ресурси.

На рис. 2.6 показана діаграма комунікації, що частково включає діаграму послідовності, але додано зв’язок із магазином, ресурсним менеджером та відображенням ресурсів.

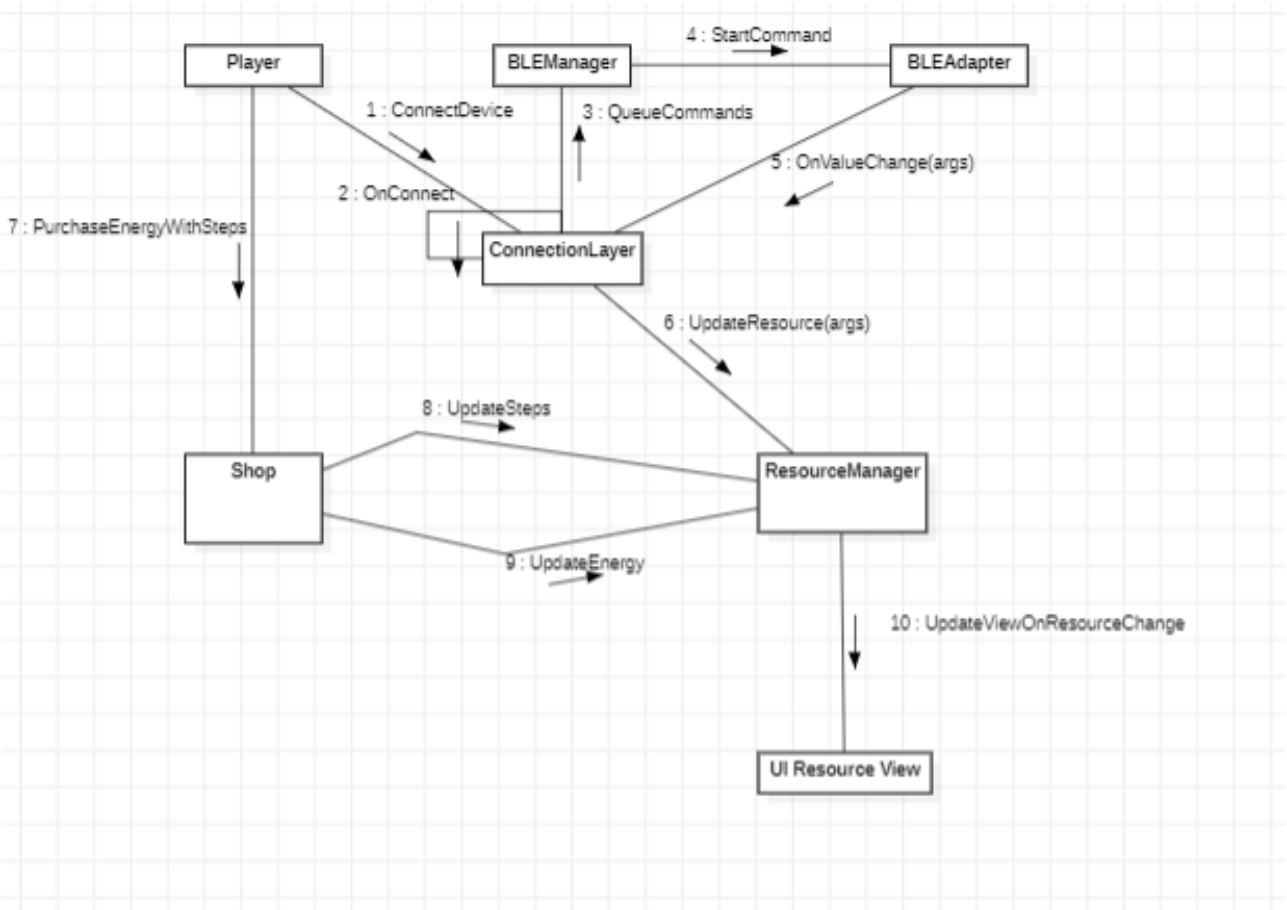


Рисунок 2.6 – Діаграма комунікації

2.5 Діаграма станів та переходів

Діаграми станів та переходів це ще один вид UML-діаграм, які допомагають полегшити розуміння, які стани має застосунок та які переходи між ними існують. Нижче показану узагальнену діаграму для усього застосунку із описом (рис. 2.7).

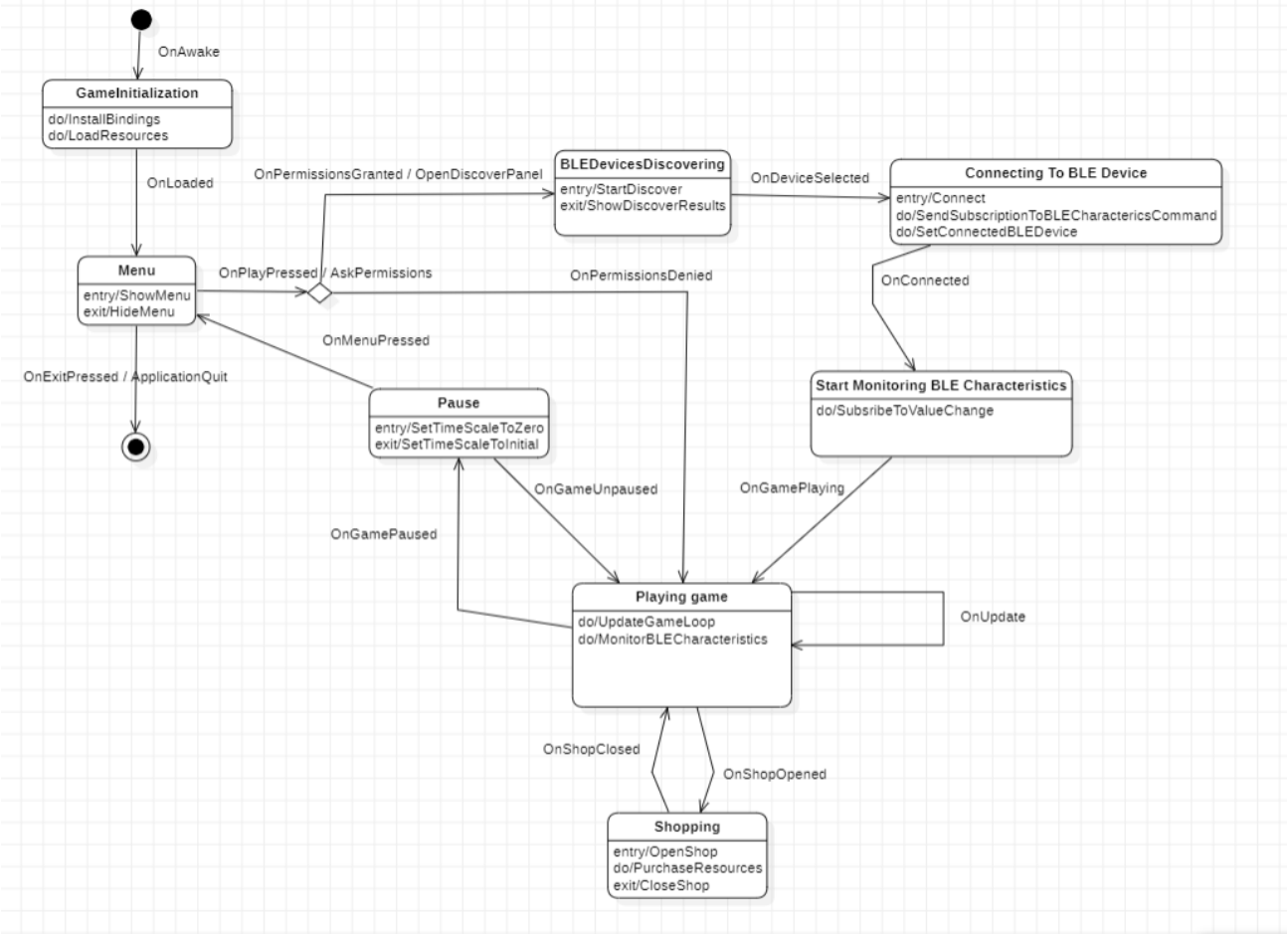


Рисунок 2.7 – Діаграма станів та переходів застосунку

Початковий стан гри запускають event-методи Awake, у яких відбувається основна ініціалізація необхідних класів та логіки. У стані *GameInitialization* проводиться інсталювання прив’язок необхідних залежностей та завантаження ресурсів, після чого відбувається перехід у *Menu*.

В головному меню знаходяться кнопки, які пропонують почати гру чи вийти зі гри. При натисненні на кнопку *Play* відбувається перехід до розгалуження, яке визначає стан через надання/ненадання дозволів на

використання Bluetooth та фізичних даних з фітнес-браслета. При наданні дозволів відкривається панель із пошуком найближчих BLE-девайсів. При відмові відбувається перехід до гри.

У разі надання дозволів та знаходження девайсу, гравець обирає потрібний та підключається до нього, відбувається перехід до *Connecting To BLE Device***,** який надсилає команду на підписку про зміни необхідних для гри характеристик (кроки, пульс, калорії, дистанція). Після успішного підключення перехід до *Start Monitoring BLE Characteristics*.

Після переходу у стан *Playing Game* також можливі переходи у магазин та пауз-меню, через яке можна повернутись до головного меню та вийти зі гри повністю.

Висновки до розділу 2

В другому розділі було описано основні етапи, сценарії використання та алгоритми роботи ігрового застосунку. Розроблено UML-діаграми, а саме діаграму використання, діаграми станів та переходів, діаграми діяльності та інші, завдяки чому закріплено навички описувати алгоритми та архітектуру застосунку за допомогою UML.

Обґрунтовано використання стеку технологій, а саме ігрового рушія Unity, мов програмування C# та Java, технології Bluetooth Low Energy.

3 ПРОЄКТУВАННЯ ІГРОВОГО ЗАСТОСУНКУ

3.1 UML-діаграми

UML (Unified Modeling Language) – це мова моделювання, яка використовується для візуалізації, проєктування та документування програмних систем [5]. Для проєктування мобільних ігрових застосунків також використовують UML-діаграми, частину з них вже було описано і використано у попередньому розділі у якості процесу моделювання застосунку. Нижче наведено більш детальне пояснення основних видів UML-діаграм, що використовуються при моделювання та проєктуванні систем.

Діаграми варіантів використання (англ. Use Case Diagrams) – ці діаграми візуалізовують функції системи з точки зору її взаємодії з зовнішніми акторами (користувачами).

Діаграми класів (англ. Class Diagrams) відображають структуру системи шляхом описування класів, їх взаємозв'язків, атрибутів та методів.

Діаграми послідовності (англ. Sequence Diagrams) використовуються для візуалізації взаємодії об'єктів та послідовності викликів методів між ними.

Діаграми діяльності (англ. Activity Diagrams) дозволяють моделювати послідовність дій або процесів, що відбуваються в системі.

Діаграми станів та переходів (англ. State Diagrams) використовуються для моделювання поведінки об'єкта або системи в залежності від їх поточного стану.

Діаграми компонентів (англ. Component Diagrams) показують організацію та взаємозв'язки компонентів програмної системи.

Діаграми розгортання (англ. Deployment Diagrams) використовуються для моделювання фізичної інфраструктури системи, такої як сервери, комп'ютери та мережі.

Діаграми пакетів (англ. Package Diagrams) допомагають організувати компоненти моделі в пакети для кращого управління складними моделями.

Діаграми комунікації (англ. Communication Diagrams) подібні до діаграм послідовності, але зображують взаємодію об'єктів на більш високому рівні абстракції.

Під час моделювання у другому розділі було показано використання діаграм сценаріїв використання, станів та переходів, діяльності, послідовності та комунікацій, оскільки вони спрощують визначення основних вимог для розробки застосунку. У третьому розділі використано діаграму класів та показано таблицю з основними класами, їх атрибутами та методами.

3.2 Діаграма класів

Діаграма класів (рис. В.1) є важливим шаблоном для переведення моделей у програмний код, є невід’ємною складовою для проєктування проєкту, що використовує об’єктно-орієнтований підхід програмування. Кожен із показаних класів має власну назву, методи та параметри, основні з них описані у табл. 3.1.

Таблиця 3.1 – Класи

Клас	**Параметри**	**Методи**	**Призначення класу**
BLEManager	Instance	Initialize()	Клас, що надсилає та приймає команди/повідомлення на Bluetooth-пристрій
	IsInitialized	Deinitialize()	
	commandsQueue	QueueCommand()	
		OnBLEMessageReceived()	
		SendCommand()	

Кінець таблиці 3.1

BLECommand		Start()	Абстрактний клас Bluetooth Low Energy команди
		End()	
		CommandReceived()	
BLEObject	device	GetByteMessage()	Клас, що використовується як DTO з даними, що приходять з Bluetooth-пристрою
	service		
	characteristic		
	hasError	ToString()	
	errorMessage		
	Base64Message		
ResourcesManager	repository	GetResource()	Клас, що оновлює ігрові ресурси (енергію, кроки тощо)
		UpdateResource()	
		GetMoney()	
		GetEnergy()	
		GetTotalSteps()	
		GetTotalCalories()	
LevelManager	resourcesManager	TryOpenNewLevel()	Клас, що відповідає за відкриття і обробку рівнів
		SetTimer()	
		ValidateResources()	
Shop	resourcesManager	PurchaseEnergy()	Клас, що відповідає за покупку/обмін ресурсів

На діаграмі класів на рис. В.1 та в табл. 3.1 відображені лише основні класи, що використовуються для ігрових процесів; інші класи (UI-представлення, скрипти керування гравця (камери), меню налаштувань тощо) не були включені, оскільки є досить тривіальними.

3.3 Особливості розробки мобільних ігрових застосунків

Мобільні ігри – це програмні застосунки, які розробляються спеціально для мобільних пристроїв, таких як смартфони та планшети, і призначені для розваг. Такі застосунки створюються беручи за основу різноманітні жанри, зокрема казуальні, аркади, головоломки, стратегії, рольові ігри, спортивні ігри, гонки та багато інших. Створюватись можуть як самостійні ігри або доповнюватись в рамках вже створених франшиз або серій ігор.

Мобільні ігрові застосунки використовують різноманітні функції та можливості мобільних пристроїв, такі як сенсорний екран, акселерометр, геолокація, камера та інші сенсори. Як саме відбувається передачі даних між від сенсорів до застосунку показано на рис. 2.4 у попередньому розділі. Як і решта застосунків, вони можуть використовувати Інтернет-з'єднання для мультиплеєрного режиму гри, синхронізації даних або отримання оновлень, Bluetooth-технологію для обміну даними з аксесуарами чи іншими пристроями тощо. Фактично ігри це ті ж самі повноцінні програми, лише використовують наявну апаратну складову пристроїв для ігрових потреб.

Розробка мобільних ігрових застосунків має свої особливості, пов'язані з характеристиками мобільних платформ та особливостями геймдизайну. Деякі з них наведені нижче:

1) оптимізація для різних пристроїв – мобільні пристрої мають різні характеристики: розмір екрану, процесор, оперативна пам'ять та графічні можливості, завжди потрібно враховувати ці різниці та оптимізувати гру таким чином (зокрема і програмним кодом), щоб вона працювала на різних пристроях з максимальною продуктивністю;

2) сенсорне керування, оскільки переважна більшість мобільних ігор використовує сенсорний екран для керування, що є очевидною вимогою для

розробки. Необхідно створювати інтуїтивні та зручні інтерфейси, що відповідають можливостям сенсорного керування;

3) кросплатформність – в залежності від цільової аудиторії, може виникати потреба в розробці мобільної гри для різних платформ, таких як iOS та Android. Для цього можуть бути використані фреймворки та інструменти, які дозволяють спроєктувати гру, сумісну з різними платформами;

4) геймдизайн для коротких сесій – мобільні ігрові сесії часто бувають короткими, оскільки користувачі грають невеликими проміжками у вільний час. Тому важливо розробляти геймплей, який може бути приємним, навіть у короткій ігровій сесії.

3.4 Використання Unity для розробки мобільної гри на Android

Враховуючи наведені у попередньому підрозділі особливості, недарма було обрано саме рушій Unity для розробки застосунку, так як він покриває більшість вимог до розробки ігрових застосунків, зокрема і мобільних [13]. Цільовою ОС було обрано Android, оскільки переважна більшість мобільних ігор створюється саме під неї [20], на додачу Unity пропонує легкий і простий спосіб «збирання» проєкту якраз під цю ОС.

Рисунок 3.1 – Збирання «Unity» проєкту як APK на ОС Android

Варто зазначити, що Unity дозволяє працювати з однією кодовою базою, написаною на мові C# [9] та «білдити» проєкт під різні платформи, однак це потребує врахування певних специфічних нюансів для кожної з платформ чи ОС (рис. 3.2).

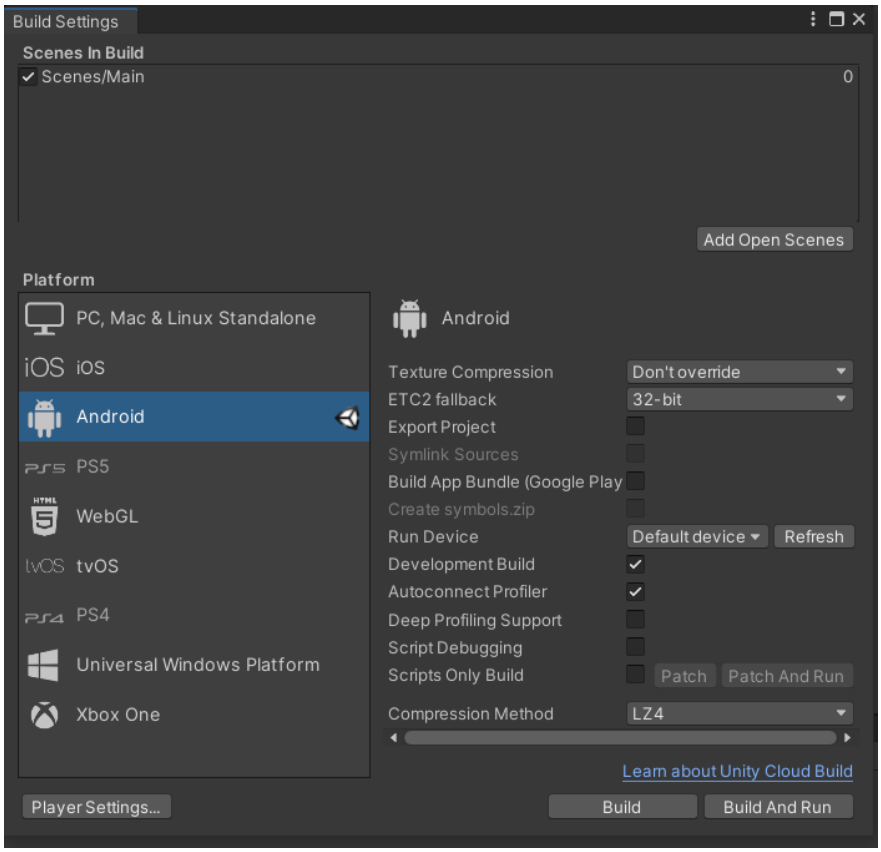


Рисунок 3.2 – Вигляд вкладки для «збирання» проєкту на ОС Android у самому редакторі Unity

На рис. 3.2 показано панель Unity-редактора для «збирання» проєкту під різні платформи, в тому числі і Android, що є очевидною перевагою при проєктуванні застосунку під різні пристрої, оскільки не потрібно створювати різні проєкти використовуючи безліч технологій та інструментів, враховуючи їх специфічні особливості, які фактично будуть дублювати усю логіку застосунку лише різними засобами та збільшувати складність підтримки та розвитку

продукту. Unity дбає про час розробки, бюджет і уніфікування – гра створюється один раз – а за потребою збирається та тестується під різні платформи.

3.5 Створення та організація проєкту Unity

Для створення проєкту на рушії Unity необхідно спершу завантажити спеціальне ПЗ – Unity Hub [14]. Це зручний та універсальний репозиторій із усіма доступними версіями рушія та розширеннями (модулями), які можна додатково встановлювати (рис. 3.3).

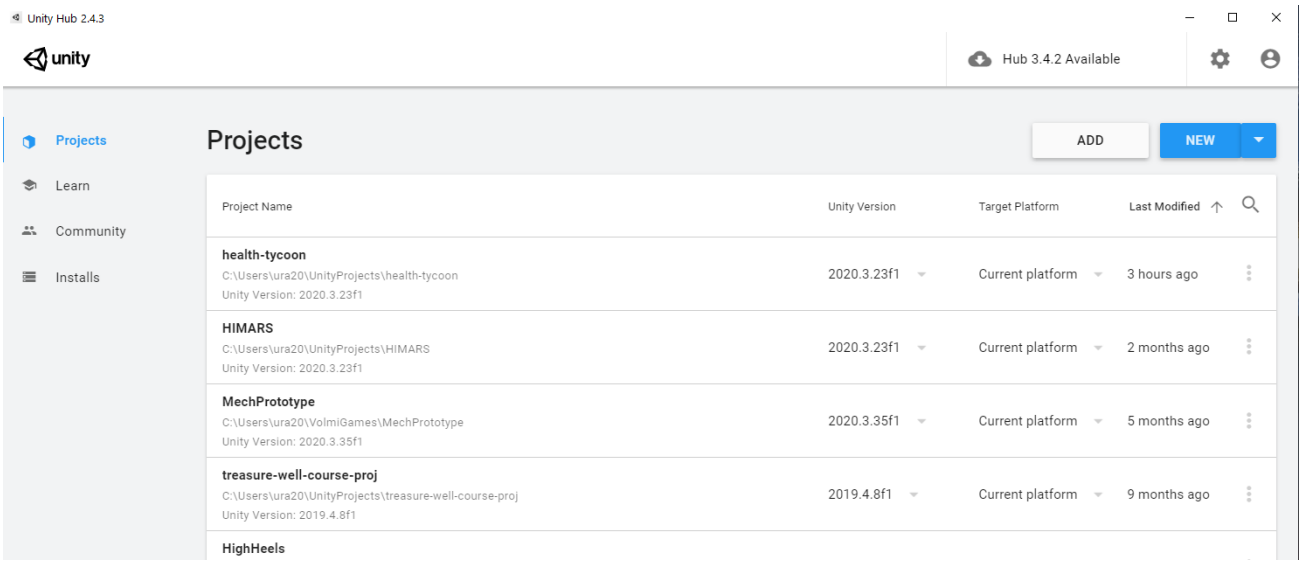


Рисунок 3.3 – Інтерфейс із доступними проєктами Unity Hub

Для створення проєкту треба натиснути New та обрати шаблон проєкту; пропонуються стандартні варіанти: 2D, 3D та шаблони для різних видів Renderer Pipeline (рис. 3.4).

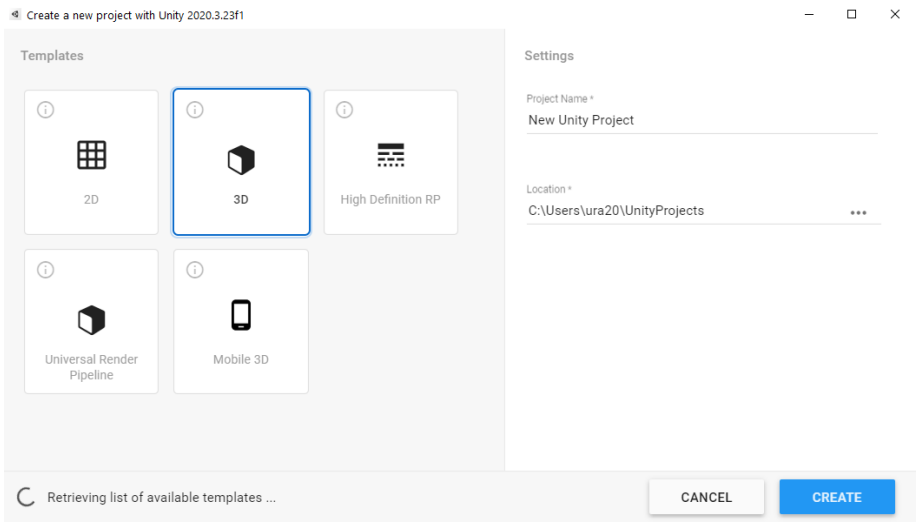


Рисунок 3.4 – Створення нового проєкту

Було обрано стандартний 3D-шаблон та ще встановлено модуль на обрану версію Unity для підтримки «збирання» проєкту під ОС Android.

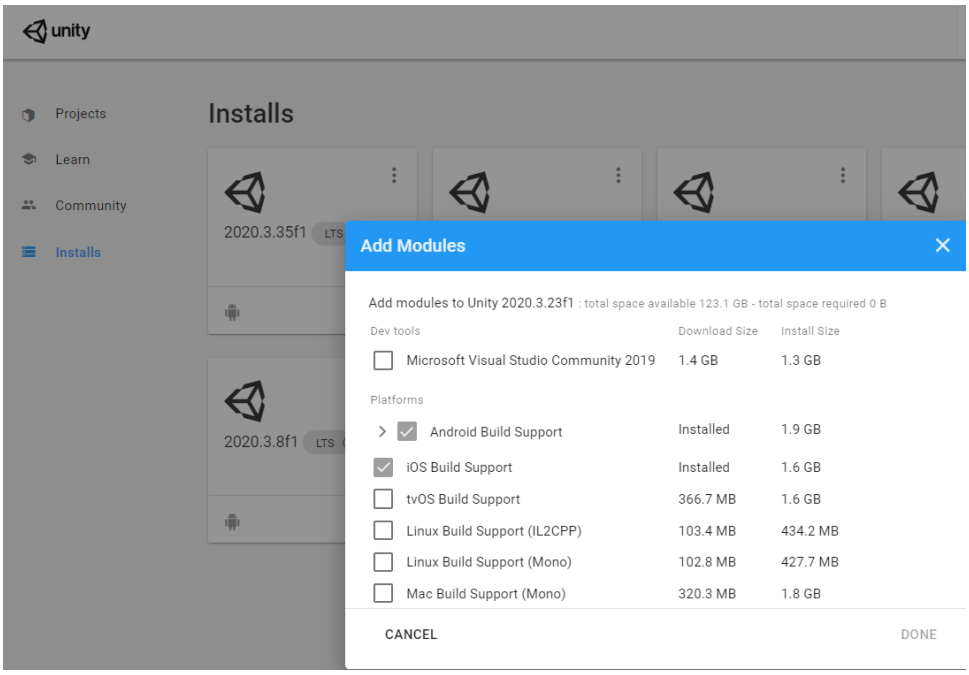


Рисунок 3.5 – Додавання модуля Android Build Supportдо обраної версії Unity

Після успішного встановлення розпочато організацію проєкту по необхідним папкам для ресурсів та коду.

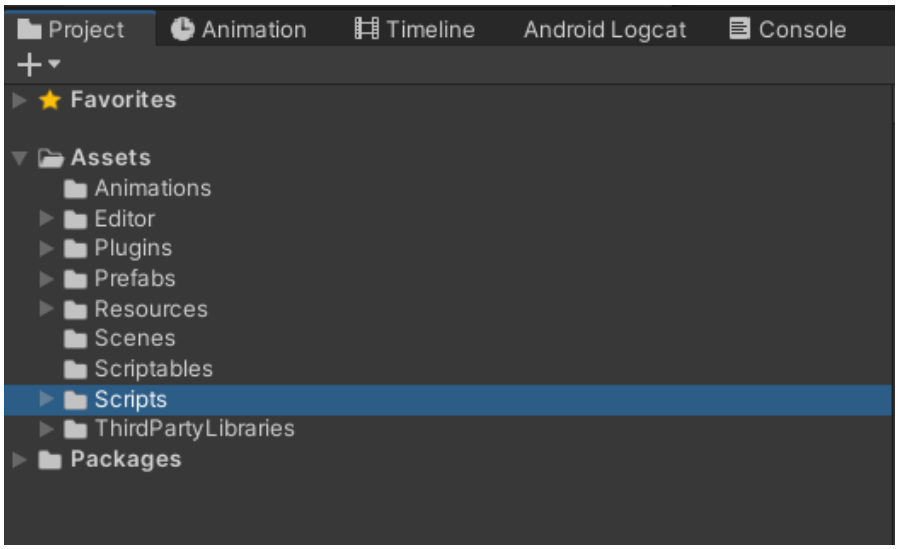


Рисунок 3.6 – Створення папок і організація проєкту

Для проєкту створено ряд папок, які будуть використовуватись в подальшій розробці, яка буде детальніше описана в останньому розділі КРБ. Нижче наведено короткий опис мети кожної з папок:

1) Animations – папка для анімації та аніматорів, які будуть використані у грі;

2) Editor – папка, в якій знаходяться ресурси, що будуть використані лише у редакторі;

3) Plugins – папка, де зберігаються плагіни (бібліотеки) для різних платформ (Android, iOS тощо), там будуть присутні плагіни для зв'язку із сенсорами Android та фітнес-браслета;

4) Prefabs – папка для ігрових об'єктів (префабів), які готові до використання на ігровій сцені;

5) Resources – папка для збереження важливих ресурсів (файлів, текстур, спрайтів, моделей тощо);

6) Scenes – папка для збереження сцен;

7) Scriptables – папка для збереження спец. Файлів Unity, які допомагають створювати власні налаштування/дані для різних ігрових механік;

8) Scripts – папка зі скриптами (файли з програмним кодом, які містять ігрову логіку);

9) ThirdPartyLibraries – папка для ресурсів або сторонніх бібліотек, які можуть використовуватись для розробки.

Розділення проєкту на окремі папки та їх організація є важливою складовою під час проєктування застосунку, адже це впливає на розуміння розробником призначення тих чи інших директорій та швидкості навігації по проєкту.

3.6 Огляд додаткових технологій

Будь-який проєкт завжди використовує сторонні бібліотеки, фреймворки, інструменти для зручної та швидкої розробки. Безкоштовні технології надзвичайно корисні при розробці власних проєктів, щоб заповнити або замінити частину «прогалин» у ПЗ на деякий час або й на весь час життя продукту. У цьому проєкту було також вирішено знайти технології які допоможуть та спростять розробку.

3.6.1 Zenject

Гарним інструментом для роботи із залежностями у Unity (і .NET застосунками в цілому) є Dependency Injection фреймворк – Zenject (Extenject) [15]. Це зручний та добре документований фреймворк, який дозволяє позбутись від проблеми залежностей, особливо, коли патерн проєктування Singleton різко стає антипатерном та ускладнює взаємодію із залежностями у проєкті.

Рисунок 3.7 – Логотип Zenject

В останньому розділі ще буде наведено приклади використання цього фреймворку під час кодування логіки застосунку. Встановлення даного фреймворку доволі легке – потрібно скопіювати посилання на репозиторій та додати його до залежностей Package Manager`а в Unity.

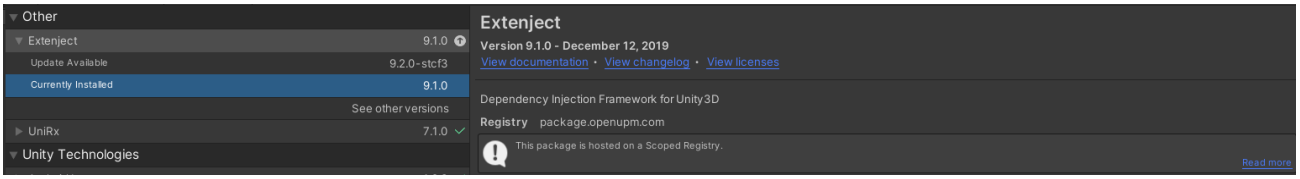


Рисунок 3.8 – Додавання фреймворку як пакета

3.6.2 DOTween

Анімування тексту, лічильників, переміщень об’єктів, адаптивні анімації панелей, кнопок – загалом UI через код – важлива складова UX будь-якої гри, гравець має відчувати задоволення від плавного переходу UI, а не різкою зміною вікон та анімацій. Все це можна досягнути використовуючи open-source бібліотеку – DOTween (HOTween) [16]. Це надзвичайно легка та зручна бібліотека із широким функціоналом, який дозволяє анімувати UI без зайвих проблем.

На сайті розробника присутні також і порівняльна таблиця аналогів-конкурентів, які пропонують подібний функціонал, однак DOTween є найбільш оптимізованим із усіх на зараз наявних бібліотек для анімування.

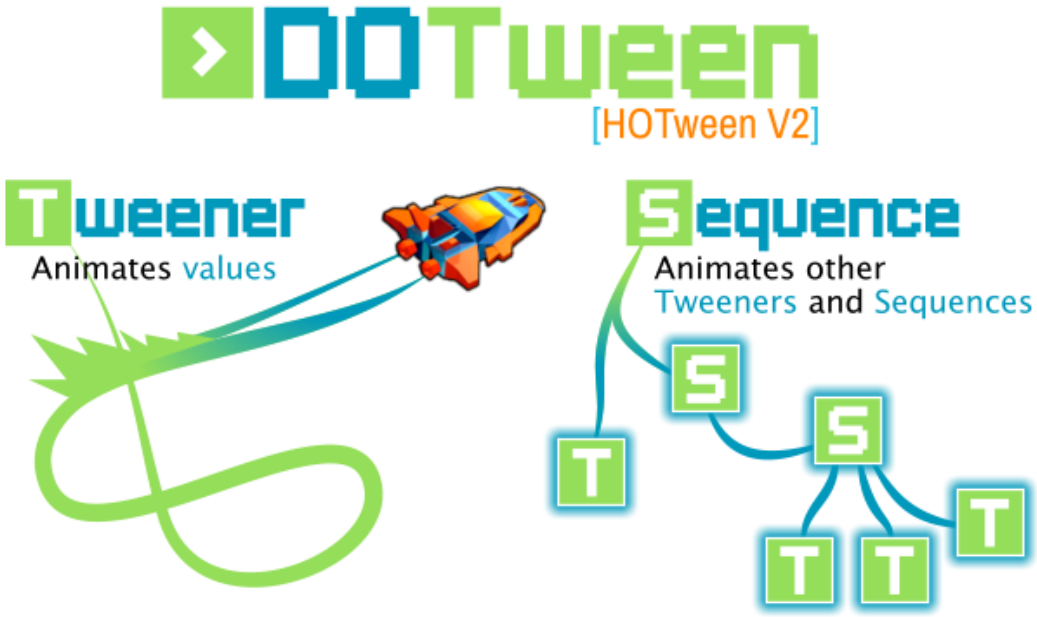


Рисунок 3.9 – Прев'ю бібліотеки

3.6.3 Unity Android BLE

В ході тривалого пошуку інформації про те, яким чином можна зв'язати Unity-застосунок із пристроями, які використовують BLE-технологію зв'язку було знайдено безкоштовний open-source плагін, який якраз і забезпечує стабільний та зручний зв'язок між ігровим застосунком та BLE-пристроєм використовуючи JNI [18]. Даний плагін створює BLEAdapter для підтримання зв'язку, використовує патерн Command для надсилання команд на BLE-пристрій для отримання/надсилання даних з/у ігровий застосунок.

Цей плагін дозволив отримати досвід у роботі із GATT характеристиками та сервісами і краще зрозуміти, як узагалі працює передача даних та у якому

вигляді. В останньому розділі більш детально описано процес «розбиття» отриманого BLE-повідомлення з кроками, пульсом, кількістю пройдених метрів та спалених калорій на окремі дані.

Рисунок 3.11 – Прев’ю плагіна

Висновки до розділу 3

В третьому розділі у таблиці було описано основні класи, зображені на діаграмі класів, визначено особливості розробки мобільних ігрових застосунків, зокрема на рушії Unity. Показано кроки для створення Unity-проєкту та додавання Android-модуля для подальшого використання при «збиранні» проєкту під ОС Android. Створено та описано організаційну структуру папок у проєкті.

В останньому підрозділі було розглянуто додаткові технології, а саме фреймворк Zenject, бібліотеку для анімування DOTween та open-source плагін Unity Android BLE для зв’язку із BLE-пристроями.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ЗАСТОСУНКУ ТА ТЕСТУВАННЯ

4.1 Створення дизайну ігрового застосунку

UI-дизайн (інтерфейс користувача) в ігровому застосунку є дуже важливим елементом, оскільки він визначає спосіб, яким користувачі взаємодіють з грою. Вірно спроектований дизайн може суттєво покращити враження гравців від гри та забезпечити їм комфортну і просту взаємодію із геймплеєм.

Ігровий застосунок містить деякі важливі елементи інтерфейсу, такі як: головне меню, меню налаштувань, меню паузи, меню вибору BLE-пристрою, ігровий інтерфейс (лічильники ресурсів, кнопка налаштувань, покажчик прогресу активних квестів). Для створення такого інтерфейсу було використано стандартні інструменти Unity, а саме UI Canvas-компоненти, що включають різноманітні елементи – кнопки, текстові поля, панелі тощо.

На рис. 4.2, а показано головне меню, яке включає три основні кнопки: Play, Settings та Exit. По натисненню на кнопку Play гравець перейде у меню пошуку BLE-пристрою для підключення його до гри з метою моніторингу фізичних даних. Користувачу перед тим також буде показано панель (рис. 4.1) із інформацією про використання деяких прав та потребу в увімкненні BLE сервісу під час гри для коректної роботи функцій моніторингу.

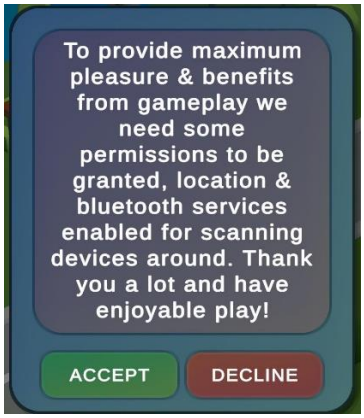
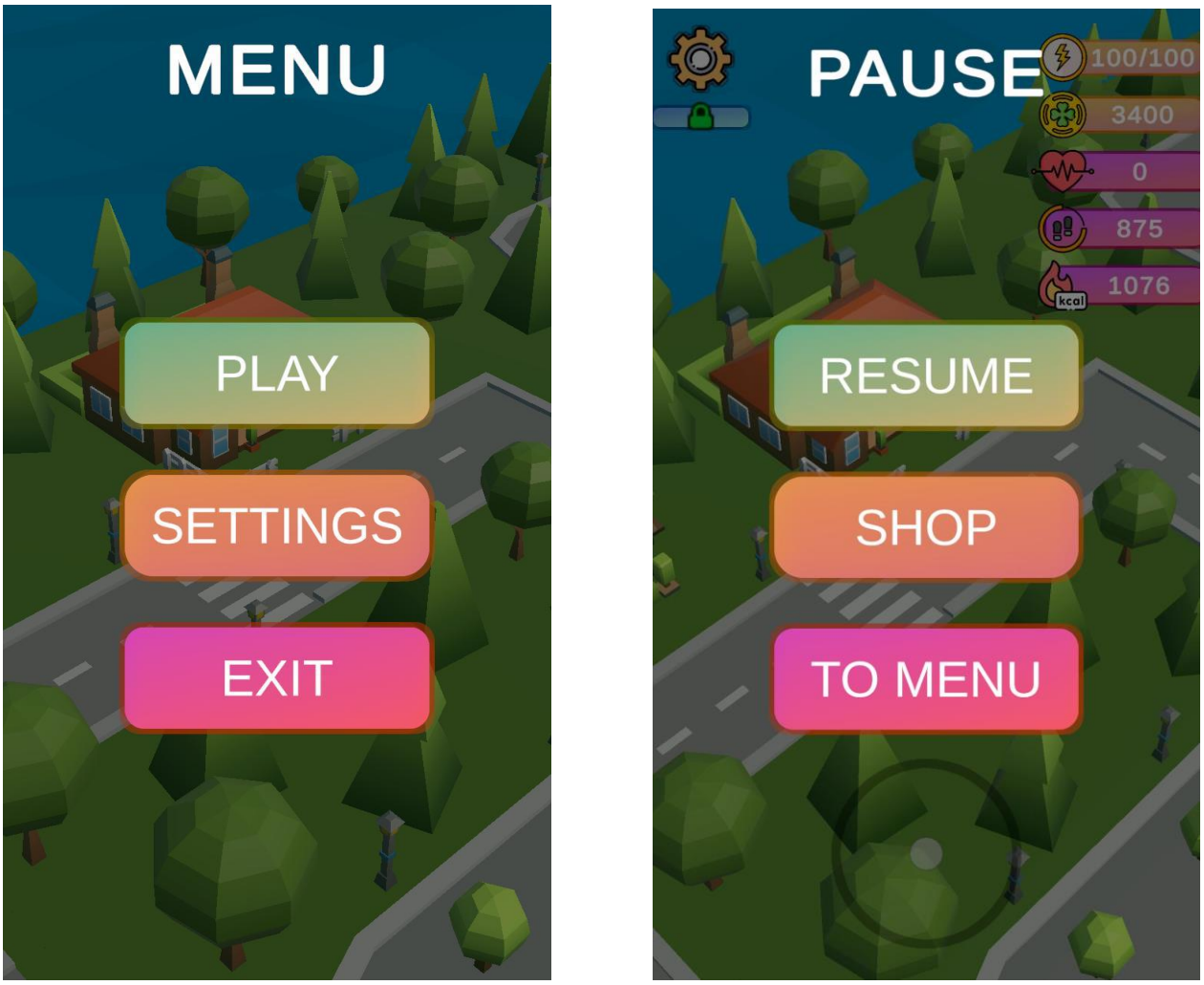


Рисунок 4.1 – Панель із описом



а) б)

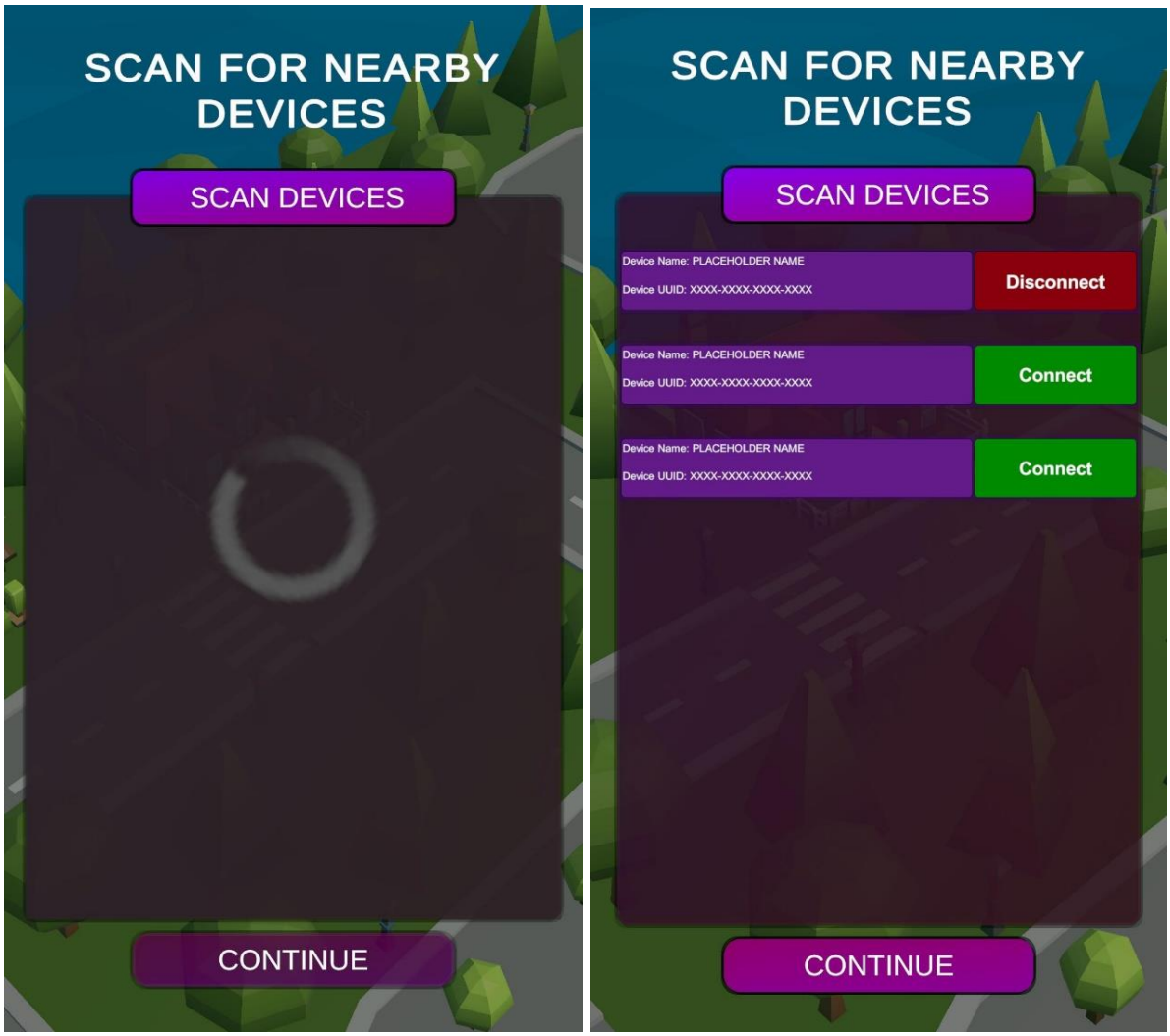
Рисунок 4.2 – Меню застосунку: а – головне меню, б – меню паузи

При натисненні на кнопку Settings користувачу відкриється панель із налаштуваннями, а при натисненні Exit – гра закриється.

На рис. 4.2, б показано меню паузи, яке дозволяє продовжити гру, перейти до магазину або у головне меню. Детальніше про магазин буде описано у наступних підрозділах.

Як вже було описано вище, після натиснення на кнопку Play, користувачу буде показано меню для пошуку найближчих пристроїв, гравець має натиснути Scan Devices щоб почати сканування на наявність пристроїв, які підтримують BLE, якщо було знайдено пристрої, то гравець має обрати свій фітнес-браслет та

натиснути Connect для підключення, тут же користувач може відключити браслет та підключити інший, якщо такий наявний (рис 4.3).



а) б)

Рисунок 4.3 – Мокап меню пошуку BLE-пристроїв: а – після натиснення на кнопку Scan Devices; б – підключено перший пристрій

Гравцю також дається можливість пропустити обрання BLE-пристрою, просто натиснувши кнопку Continue без вибору зі списку наявних пристроїв та без натиснення на Scan Devices, однак тоді у грі буде відключена функція моніторингу та обміну даних з браслета на валюту у грі.

Після успішно обраного фітнес-браслета, гравець має натиснути Continue, де йому буде відображене ігровий інтерфейс із ресурсами (енергія та монети) та фізичними даними (пульс, кількість пройдених кроків, кількість спалених калорій) (рис. 4.4). Зліва згори присутня кнопка для відкриття пауз-меню, що вже було описано вище і відображено на рис. 4.2. Також там же присутня висувна панель із активними квестами, де буде відображено прогрес по завдання (в залежності від режиму завдання будуть відрізнятись, при увімкненому BLE будуть доступні завдання на проходження кроків, спалення калорій, при вимкненому – на витрату/отримання монет та енергії).

а) б)

Рисунок 4.4 – Ігровий інтерфейс із лічильниками: а – лічильники ресурсів та фізичних даних; б – прогрес квестів та налаштування

4.2 Реалізація підключення BLE-пристроїв до застосунку

У третьому розділі було зазначено використання open-source плагіну, який спрощує зв'язок із BLE-пристроями. З використанням цього плагіну було налаштовано зв'язок ігрового застосунку із фітнес-браслетом. На рис. 4.5 показано реалізацію метода Initialize класу BLEManager, який створює BLEAdapter для зв'язку із BLE-пристроями (фітнес-браслетом). Ініціалізація цього менеджера відбувається при виклику Awake методу, тобто коли компонент Monobehavior (Unity компонент) ініціалізується.

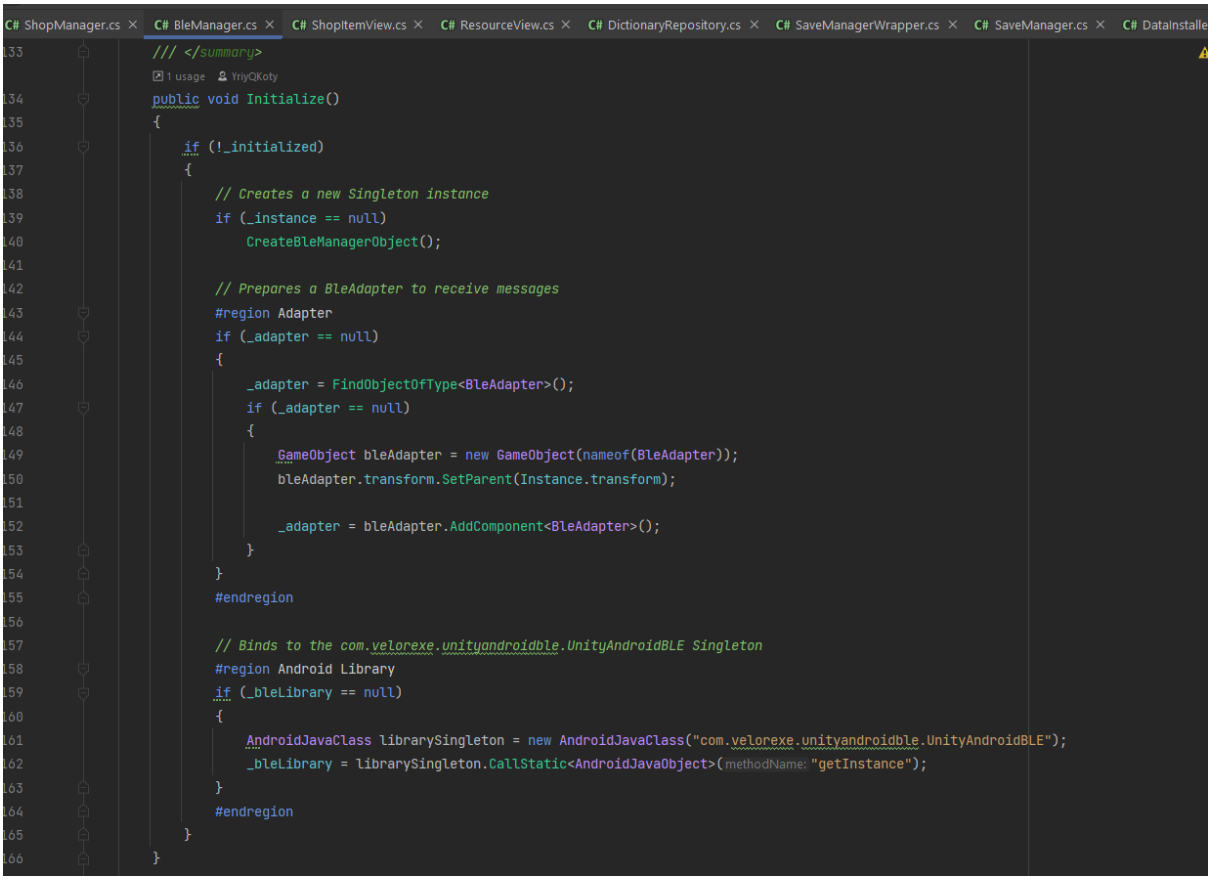


Рисунок 4.5 – Метод ініціалізації BLEManager`а

Також під час ініціалізації відбувається підписка подій на отримання повідомлення від пристрою та отримання помилки (рис. 4.6).

```
private void Awake()
{
    _instance = this;

    if (InitializeOnAwake)
        Initialize();

    _adapter.OnMessageReceived += OnBleMessageReceived;
    _adapter.OnErrorReceived += OnErrorReceived;

    DontDestroyOnLoad(target: this);
}
```

Рисунок 4.6 – Awake-метод BLE Manager, який окрім ініціалізації підписує адаптер на події отримання повідомлення та помилок

Для зв’язку із BLE-пристроєм необхідно спершу відсканувати наявні пристрої, Unity Android BLE плагін допомагає з цим, надаючи API команд для сканування найближчих пристроїв за допомогою компоненту BLE Interactor (рис. 4.7). Він включає в себе підписку на кнопку сканування (виклик методу сканування) та відображення найближчих доступних знайдених пристроїв у вигляді списку.

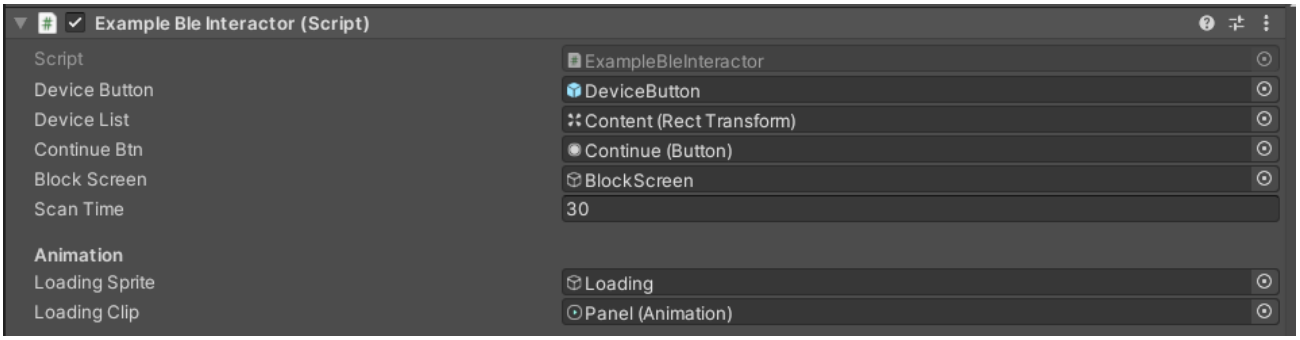


Рисунок 4.7 – Monobehavior-компонент, який дозволяє сканувати та відображати BLE-пристрої

```
1 usage  YriyQKoty
private void Scan()
{
    _isScanning = true;

    _loadingSprite.SetActive(true);
    _blockScreen.SetActive(true);
    _loadingClip.Play();

    BleManager.Instance.QueueCommand(new DiscoverDevices(OnDeviceFound, discoverTime: _scanTime * 1000));
}
```

Рисунок 4.8 – Метод Scan запускає скан-команду протягом _scanTime

При натисненні на кнопку Scan Devices (рис. 4.3, а) запускається даний метод, який додає команду DiscoverDevices у чергу, реєструючи callback OnDeviceFound, тобто при кожному знайденому пристрої буде виконано callback-метод. Цей метод (рис. 4.9) створює елемент UI у адаптивному списку із назвою пристрою та його UUID. Цей елемент містить в собі кнопку Connect, яка дозволяє підключитись до обраного BLE-пристрою. При натисненні на цю кнопку, її колір та назва змінюється на Disconnect, якщо потрібно обрати інший пристрій.

```
1 usage  YriyQKoty
private void OnDeviceFound(string name, string device)
{
    DeviceButton button = Instantiate(_deviceButton, _deviceList).GetComponent<DeviceButton>();
    button.Show(uuid: name, name: device);
}
```

Рисунок 4.9 – Метод, що виконується при знаходженні BLE-пристрою

В момент, коли гравець натискає на кнопку Connect (рис. 4.3, б) виконується наступний код (рис. 4.10). При виклику цього методу створюється ConnectCommand із обробниками подій OnConnected та OnDisconnected, які виконуватимуться відповідно до стану цього пристрою та введення користувача.

```
1 asset usage  YriyQKoty
public void Connect()
{
    if (!_isConnected)
    {
        _connectCommand = new ConnectToDevice(_deviceUuid, OnConnected, OnDisconnected);
        BleManager.Instance.QueueCommand(_connectCommand);

        PlayerPrefs.SetString("ConnectedUUID", _deviceUuid);
    }
    else
    {
        _connectCommand.Disconnect();

        PlayerPrefs.DeleteKey("ConnectedUUID");
    }
}
```

Рисунок 4.10 – Метод підключення до визначеного BLE-пристрою

При під’єднанні до пристрою було додано метод-підписку, яка виконується при оновленні даних у фітнес-браслеті. Таким чином, при зміні пульсу, кількість кроків, кількості спалених калорій чи кількості пройдених метрів буде відправлено подію та виконано метод-обробник (рис. 4.11).

Метод SubscribeToHeartrateMonitoring додає обробник при оновленні даних про пульс та виведення цих даних до консолі (рис. 4.12). В цьому ж методі відбувається оновлення даних у CharacteristicsRepository, який зберігає усі актуальні дані, які потім використовуються під час гри.

Метод SubcribeToActivityMonitoring включає в себе обробник на оновлення даних про фізичну активність, яка включає в себе: кількість пройдених кроків, пройдені метри та спалені калорії у вигляді base64 повідомлення, яке необхідно розшифрувати та визначити конкретні байти, які відповідають тим чи іншим даним.

```
1 usage  YriyQKoty  ••• More
private async void OnConnected(string deviceUuid)
{
    _previousColor = _deviceButtonImage.color;
    _deviceButtonImage.color = _onConnectedColor;

    _isConnected = true;
    _deviceButtonText.text = "Disconnect";

    SubscribeToHeartRateMonitoring();

    SubscribeToActivityMonitoring();

    BleManager.Instance.QueueCommand(SubscribeToHeartRateCharacteristic);

    await Task.Delay(50);

    BleManager.Instance.QueueCommand(SubscribeToActivityCharacteristic);
}
```

Рисунок 4.11 – Метод OnConnected, який включає в себе підписку на зміну фізичних даних (пульс, кроки, дистанція, калорії)

```
1 usage  YriyQKoty
private void SubscribeToHeartRateMonitoring()
{
    SubscribeToHeartRateCharacteristic = new SubscribeToCharacteristic( deviceAddress: _deviceUuid,
        UUIDsConstants.HEART_RATE_SERVICE_UUID_SHORT, characteristic: UUIDsConstants.HEART_RATE_MEASUREMENT_CHAR_UUID_SHORT, onDataFound: value :byte[] =>
        {
            Debug.Log( message: Encoding.UTF8.GetString(value));

            characteristicDataRepository.Update(CharacteristicDataConstants.HEART_RATE, value[1]);
        });
}
```

Рисунок 4.12 – Метод SubscribeToHeartMonitoring, який містить обробник оновлення даних характеристики пульсу

З допомогою reverse engineering'у було визначено [19], що характеристика активності для фітнес-браслета Xiaomi Mi Smart Band 6 (рис. 4.14) має такий порядок байтів та їх призначення: 2-й, 3-й, 4-й, 5-й байти відповідають за

значення пройдених кроків; 6-й, 7-й, 8-й, 9-й байти за значення метрів; а 10-й, 11-й, 12-й, 13-й – за кількість спалених калорій.

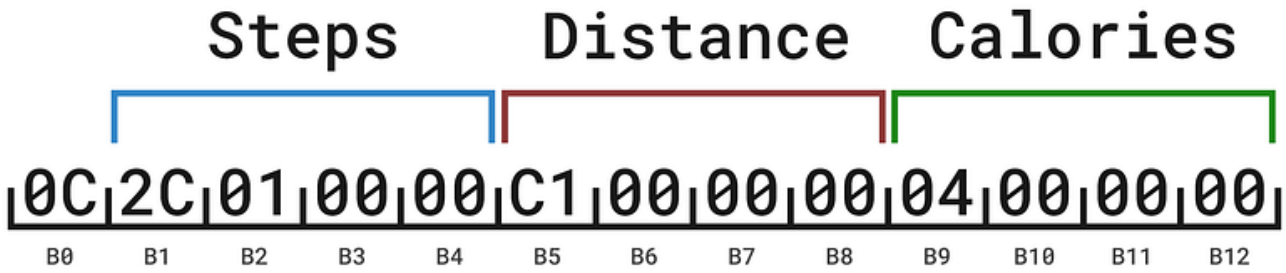


Рисунок 4.13 – Схематичне представлення отриманого повідомлення з масивом байтів у характеристиці активності
для фітнес-браслета Xiaomi Mi Smart Band 6

```
1 usage  YriyQKoty
private void SubscribeToActivityMonitoring()
{
    SubscribeToActivityCharacteristic = new SubscribeToCharacteristic( deviceAddress: _deviceUuid,
        UUIDsConstants.MI_STEPS_SERVICE_UUID_LONG,
        characteristic: UUIDsConstants.MI_STEPS_CHAR_UUID_LONG,  onDataFound: value :byte[]  =>
        {
            var steps = BitConverter.ToInt32( value: new[] { value[1], value[2], value[3], value[4] },  startIndex: 0);
            var meters = BitConverter.ToInt32( value: new[] { value[5], value[6], value[7], value[8] },  startIndex: 0);
            var calories = BitConverter.ToInt32( value: new[] { value[9], value[10], value[11], value[12] },  startIndex: 0);

            Debug.Log( message: $"Steps: {steps}");
            Debug.Log( message: $"Meters: {meters}");
            Debug.Log( message: $"Calories: {calories}");
            Debug.Log( message: Encoding.UTF8.GetString(value));

            characteristicDataRepository.Update(CharacteristicDataConstants.STEPS_PER_DAY, steps);
            _readOnlyResourceManager.GetRawResource(EResourceType.STEPS).CountPerDay = steps;

            characteristicDataRepository.Update(CharacteristicDataConstants.METERS_PER_DAY, meters);

            characteristicDataRepository.Update(CharacteristicDataConstants.CALORIES_PER_DAY, calories);
            _readOnlyResourceManager.GetRawResource(EResourceType.CALLORIES).CountPerDay = calories;
        },  customGatt: true);
}
```

Рисунок 4.14 – Метод SubscribeToActivityMonitoring

В цьому методі відбувається конвертування даних з масиву байтів до конкретних значень, а саме кроків, метрів та калорій, ці дані також виводять в консоль для спрощення тестування правильності підключення фітнес-браслету та коректності даних.

Після конвертування даних вони записуються у два окремих сховища – у CharacteristicsRepository («сирі» дані) та ResourcesManager (ігрові дані). Метою такого розділення є «відносність» таких даних, оскільки вони виступають у ролі ігрових ресурсів для гри, тобто їх можна «витратити» на покупку реальних ресурсів (енергії, монет), однак оновлення цих даних приходить із фітнес-браслета у реальному часі із вирахуванням пройдених кроків, калорій, дистанції тощо, і потрібно якимось чином розділяти «сирі» дані від «ігрових», які можуть суттєво відрізнятись від тих, які отримує застосунок від фітнес-браслета. Наприклад гравець може витрати 1000 пройдених кроків на *n* енергії, і це зменшить кількість його пройдених кроків у вигляді ігрового ресурсу, однак ці дані не мають ніяким чином змінюватись у фітнес-браслеті – вони абсолютно незалежні від тих, які використовуються у грі, але при збільшенні кількості пройдених кроків, калорій, дистанції вони все ж впливають на кількість ігрової валюти. Детальна логіка обробки таких ресурсів описана у підрозділі, присвяченому ігровій механіці обрахування ресурсів та їх обміну.

Отже, під’єднання до фітнес-браслету відбувається через кнопку на інтерфейсі доступного пристрою у списку, під час під’єднання відбувається підписка ряду обробників на певні BLE характеристики (в даному випадку пульс, кроки, калорії, дистанція). В момент оновлення ці дані обов’язково записуються у два різних сховища з метою розділення їх призначення та відокремлення використання для власних потреб. Ці ж дані згодом використовуються для відображення у лічильниказ ігрового інтерфейсу, що буде згадати в наступних підрозділах.

4.3 Реалізація ігрової механіки обміну фізичних даних на ресурси

Основна задумка ігрового застосунку з функцією моніторингу здоров'я – у зв'язку фітнес-браслета із грою та гейміфікацією даних, які передаються з браслета у вигляді «ігрової» валюти, тобто обміну пройдених кроків, спалених калорій на енергію та монети, які будуть використовуватись для продовження гри (відкриття нових локацій, пришвидення тощо). У попередньому підрозділі було згадано ResourceManager та CharacteristicsRepository, перший клас використовує також ResourcesRepository, який в свою чергу зберігає усі актуальні дані з фітнес-браслета у рантаймі та у файлі, щоб гравець міг потім продовжити гру із тими даними, з якими закінчив останню гру. На рис. 4.15 показано два методи, які використовуються для відображення актуальної кількості ресурсів на ігровому інтерфейсі.

```
Frequently called  6+1 usages  YriyQKoty
public int GetTotalResource(EResourceType resourceType)
{
    if (resourceType == EResourceType.MONEY || resourceType == EResourceType.ENERGY)
    {
        return _repository.Get(resourceType).TotalCount;
    }
    else
    {
        var res :Resource = _repository.Get(resourceType);
        return res.TotalCount + res.CountPerDay - res.SpentPerDay;
    }
}

0+2 usages  YriyQKoty
public Resource GetRawResource(EResourceType resourceType)
{
    return _repository.Get(resourceType);
}
```

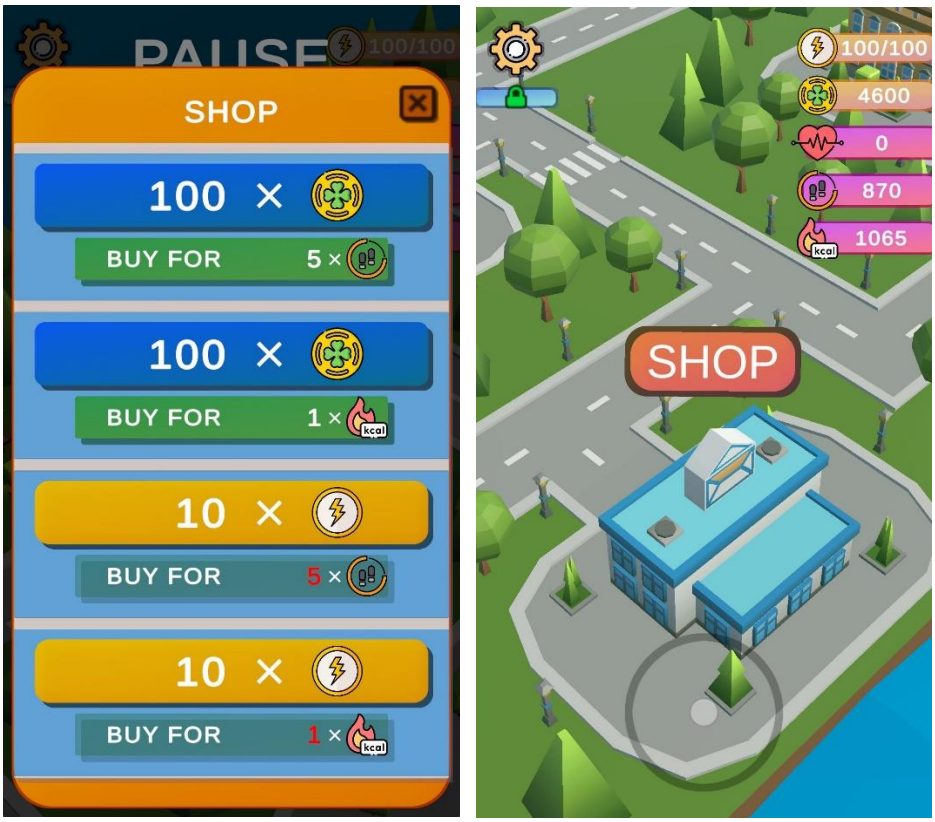
Рисунок 4.15 – Методи GetTotalResource та GetRawResource, які дозволяють отримати актуальні дані з репозиторію

Було розроблено універсальне відображення (view) для лічильників ресурсів, які отримують дані через використання ResourceManager'а та типу ресурсів, який необхідно відображати (рис. 4.16).

```
4 asset usages  YriyQKoty  ··· More
public class ResourceView : MonoBehaviour
{
    [Header("Text field")][Space( height: 2)]
    [SerializeField] private TextMeshProUGUI valueField;  Changed in 1 asset

    [Header("Resource type")] [Space( height: 2)]
    [SerializeField] private EResourceType resourceType;  Changed in 1 asset

    [Inject] private IReadOnlyResourceManager resourceManager;

    Event function  YriyQKoty
    private void Update()
    {
        valueField.text = resourceManager.GetTotalResource(resourceType).ToString();
    }

}
```

Рисунок 4.16 – Компонент, який щокадру оновлює дані у вигляді тексту в лічильнику

Наявність ігрових ресурсів передбачає певний їх обмін на якісь інші валюти або для спрощення ігрового процесу. Для цього було створено магазин, в якому можна «обміняти» кроки та калорії на монети або енергію (рис. 4.17). Для цього також було створено спеціальне відображення ShopItemView, яке представляє тип ресурсу, який можна купити, тип ресурсу який можна обміняти, їх кількість та, власне, метод купівлі цих ресурсів. Дане відображення також має валідацію, тобто, якщо потрібних ресурсів для обміну не вистачає, або енергія повністю заряджена (оскільки енергія – ресурс, який поповняється із часом), то кнопка покупки стає недоступною (рис. 4.18).



а) б)

Рисунок 4.17 – Вікно магазину із тестовими даними для обміну на енергію або монети: а – вікно магазину; б – присутність магазину на ігровій мапі

```
36          }
37
            1 usage  YriyQKoty
38          private void PurchaseItem()
39          {
40              _shopManager.PurchaseItem(itemId);
41          }
42
            Event function  YriyQKoty
43          private void Update()
44          {
45              if (!_shopManager.EnoughResourceToPurchase(itemId))
46              {
47                  purchaseBtn.interactable = false;
48                  resourcePrice.color = notEnoughResourceColor;
49              }
50              else
51              {
52                  purchaseBtn.interactable = true;
53                  resourcePrice.color = Color.white;
54              }
55          }
```

Рисунок 4.18 – Метод купівлі та валідація ресурсів для купівлі

Купівля ресурсу відбувається через клас ShopManager, а саме метод PurchaseItem (рис. 4.19), під час купівлі знімається відповідна кількість ресурсів, яка була вказана у налаштуваннях обміну, які виділені у окремий файл налаштувань.

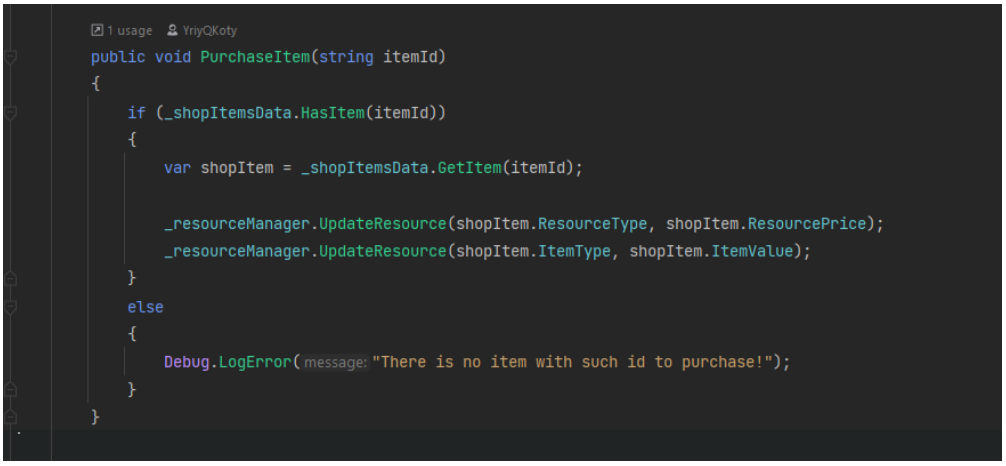


Рисунок 4.19 – Метод купівлі (обміну) ресурсів

Як видно, у методі використовуються двічі метод UpdateResource класу ResourceManager, перший виклик відповідає за оновлення кількості «ігрових» ресурсів (кроків, калорій), другий відповідає за оновлення куплених ресурсів (монет, енергії). Детальніше код цього методу наведено нижче (рис. 4.21).

Обмін даних вимагає збереження, саме тому було додано спеціальний клас SaveManager, який займається збереженням (серіалізацією) таких даних та їх завантаження (десеріалізацією) при повторному запуску гри.

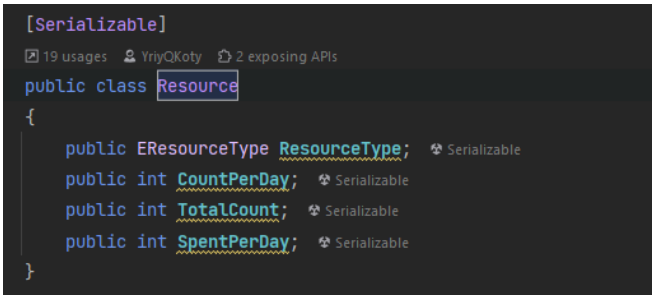
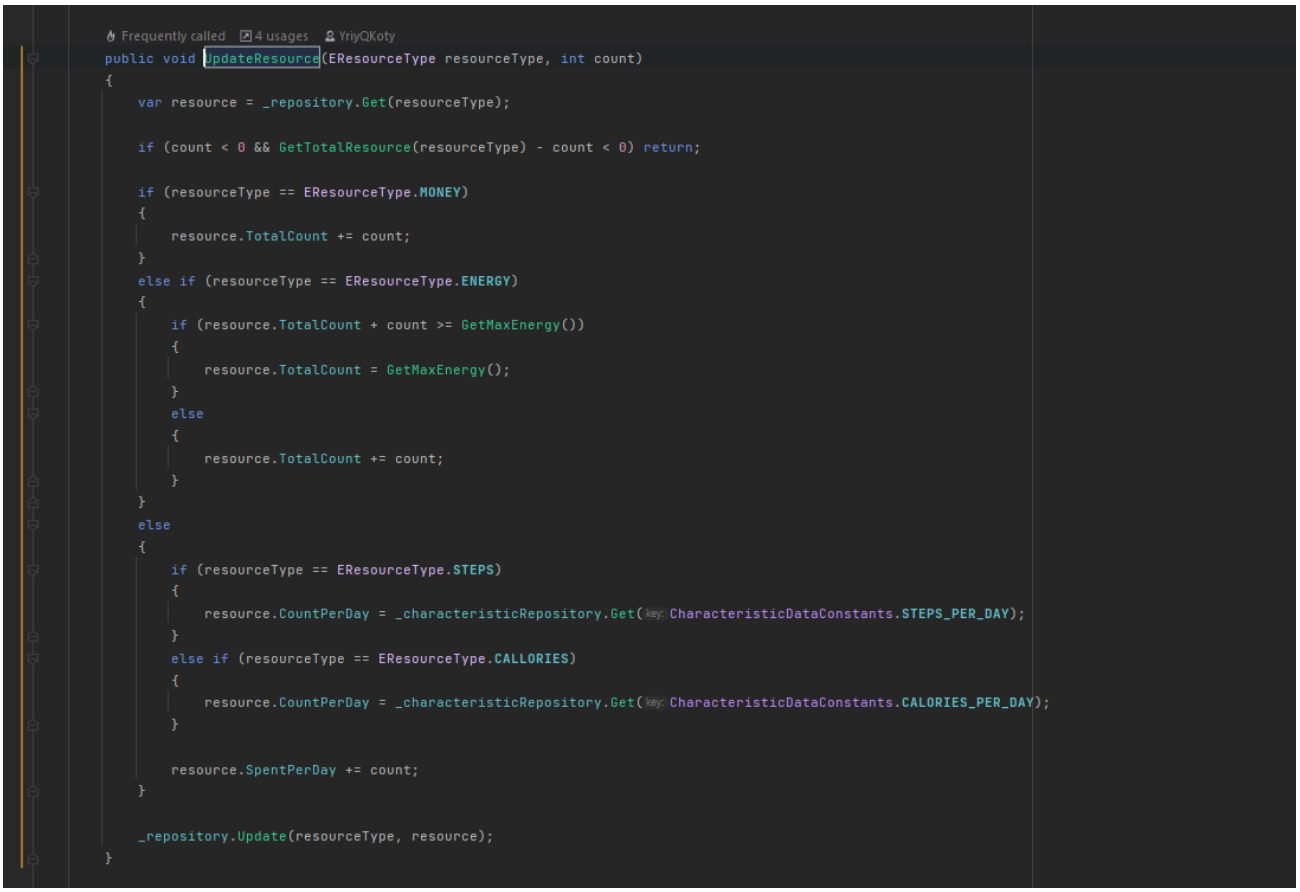


Рисунок 4.20 – Клас Resource із основними даними, які зберігаються



```
Frequently called  4 usages  YriyQKoty
public void UpdateResource(EResourceType resourceType, int count)
{
    var resource = _repository.Get(resourceType);

    if (count < 0 && GetTotalResource(resourceType) - count < 0) return;

    if (resourceType == EResourceType.MONEY)
    {
        resource.TotalCount += count;
    }
    else if (resourceType == EResourceType.ENERGY)
    {
        if (resource.TotalCount + count >= GetMaxEnergy())
        {
            resource.TotalCount = GetMaxEnergy();
        }
        else
        {
            resource.TotalCount += count;
        }
    }
    else
    {
        if (resourceType == EResourceType.STEPS)
        {
            resource.CountPerDay = _characteristicRepository.Get(key: CharacteristicDataConstants.STEPS_PER_DAY);
        }
        else if (resourceType == EResourceType.CALLORIES)
        {
            resource.CountPerDay = _characteristicRepository.Get(key: CharacteristicDataConstants.CALORIES_PER_DAY);
        }

        resource.SpentPerDay += count;
    }

    _repository.Update(resourceType, resource);
}
```

Рисунок 4.21 – Алгоритм оновлення ресурсів

Отже, алгоритм обміну ресурсів дуже простий та наочний і не потребує додаткових пояснень, окрім обміну ресурсів також було додано спеціальний клас-таймер, який алокує відновлювальні ресурси (енергію) з певною частотою (кілька хвилин), оскільки це важлива особливість, адже гравець може і не використовувати BLE-пристрої для гри, а енергію якимось чином треба відновлювати.

4.4 Тестування ігрового застосунку

Будь-які застосунки вимагають тестування. Нижче описано результати кількох успішних сценаріїв та їх розширень (табл. 4.1–4.3).

Таблиця 4.1 – Підключення гравцем BLE пристрою

Актори	Гравець
Мета	Зайти до гри, знайти свій фітнес-браслет та підключити його.
Передумова	Користувач має увімкнути сервіс BLE та геолокацію для пошуку пристроїв, користувач має надати доступ застосунку для використання BLE та локації.
Успішний сценарій: 1. Гравець увімкнув BLE та геолокацію, надав доступи застосунку. 2. Гравець заходить до гри. 3. Гравець натискає кнопку Play та переходить до пошуку пристроїв. 4. Гравець натискає кнопку Scan Devices. 5. Застосунок починає пошук найближчих пристроїв. 6. Гравець візуально бачить нові пристрої у списку. 7. Застосунок відкриває можливість обрати пристрій. 8. Гравець обирає свій пристрій та натискає Connect. 9. Пристрій підключено.	
Сценарій пройдено успішно, гравець підключив пристрій	
Розширення. Успішно виконані	
1а	Гравець проігнорував вибір пристрою та продовжив гру. Результат: продовження гри без функції моніторингу
2а	Гравець не бачить пристроїв. Результат: повідомлення по помилку (перевірити підключення BLE або пристрої).

Таблиця 4.2 – Обмін гравцем ігрової валюти

Актори	Гравець
Мета	Обміняти ігрові ресурси (кроки, калорії) на монети або енергію
Передумова	Користувач має зайти до ігрового магазину
Успішний сценарій: 1. Гравець зайшов до пауз меню та натиснув на кнопку Shop або кнопку на ігровому рівні на будівлі магазину. 2. Гравець бачить панель із товарами та цінами обміну. 3. Гравець натискає кнопку «купити» енергію за кроки. 4. Гравець отримує енергію, застосунок віднімає заявлену ціну ресурсів (кроків) із лічильника гравця.	
Сценарій пройдено успішно, гравець обміняв ресурси	

Розширення. Успішно виконані	
1а	Гравець не має достатньої кількості ресурсів. Результат: неможливість придбати ресурси в магазині, «червоний» текст ціни при обміні та «неактивні» кнопки для обміну
2а	Гравець вже має максимальний ліміт енергії. Результат: «неактивні» кнопки обміну енергії, можливість обміну монет

Таблиця 4.3 – Збереження даних застосунком

Актори	Гравець, застосунок
Мета	Перевірка збереження даних (кроків, калорій)
Передумова	Користувач має зайти в магазин, витратити ресурси, вийти зі гри, повернутись у гру
Успішний сценарій: 1. Гравець зайшов до пауз меню та натиснув на кнопку Shop або кнопку на ігровому рівні на будівлі магазину. 2. Гравець бачить панель із товарами та цінами обміну. 3. Гравець купує будь-які ресурси (енергію, монети) за кроки чи калорії. 4. Гравець отримує ресурс, застосунок віднімає заявлену ціну ресурсів (кроків, калорій) із лічильника гравця. 5. Гравець виходить зі гри через меню чи згортаючи застосунок. 6. Гравець повертається у гру. 7. К-ть ресурсів лишилась такою, як була після покупки.	
Сценарій пройдено успішно, гравець обміняв ресурси	
Розширення. Успішно виконані	
1а	Гравець вийшов зі гри о 23:59, дані фітнес-браслета обнулюються о 00:00. Результат: Дані збереглись на момент 23:59, ігрові дані не зникли та не обнулились

Висновки до розділу 4

У четвертому розділі було показано розробку ігрового дизайну та основних елементів користувацького інтерфейсу (головне меню, меню паузи, меню пошуку пристроїв, ігровий інтерфейс). Описано роботу по програмній реалізації ігрових механік обміну фізичних даних як ігрової валюти на ресурси (енергія, монети), що забезпечує гейміфікацію даних з фітнес-браслета. Наведено

реалізацію підключення BLE-пристроїв (на основі фітнес-браслета Xiaomi Mi Smart Band 6) у коді. Описано практичне застосування технології BLE та передачі даних з її допомогою.

Проведено тестування ігрового застосунку, описано успішні сценарії. Завдяки проведеному аналізу знайдено деякі незначні баги (помилки) в логіці механік та виправлено їх у коді.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи бакалавра було отримано досвід з конструювання та проєктування ПЗ, а саме ігрового застосунку із підключенням до зовнішніх пристроїв через бездротову технологію Bluetooth Low Energy. Завдяки проведеному аналізу аналогів було визначено актуальність та доцільність створення мобільного ігрового застосунку моніторингу здоров’я. За результатами дослідження аналогів сформовано вимоги, визначено основні функції та специфікацію вимог до ігрового застосунку.

Реалізовано поставлене ТЗ по створенню ігрового застосунку, базуючись на специфікації вимог до ПЗ. Для досягнення визначеної мети КРБ було виконано поставлені завдання:

– здійснено аналіз предметної області та застосунків-аналогів;

– визначено основні функції, створено специфікацію вимог до ПЗ;

– створено діаграму сценаріїв використання, діаграми діяльності, послідовності, класів, взаємодії та комунікацій, описано основні та альтернативні use cases;

– обґрунтовано використання технології Bluetooth Low Energy для зв’язку з фітнес-браслетом;

– створено мобільний ігровий застосунок на мові програмування C# та в ігровому рушії Unity з підтримкою функції моніторингу здоров’я.

Згідно з технічним завданням було обрано найбільш зручні інструменти та технології розробки, а саме: ігровий рушій Unity, мови програмування C#, Java та open-source плагіни для зв’язку мобільного застосунку із фітнес-браслетом. Результатом КРБ є прототип мобільного застосунку у вигляді казуальної гри із підтримкою функції моніторингу біомедичних показників стану здоров’я користувача.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Афонін Ю. С., Журавська І. М. Використання платформи Unity для розробки казуальної гри з функцією моніторингу біомедичних показників. *Free and Open Source Software (FOSS'2023)* : тези доп. XI Міжнар. наук.-практ. конф. / Харків. нац. економ. ун-т ім. Семена Кузнеця. Харків, 14–16 лютого 2023 р. Харків : ХНЕУ ім. Семена Кузнеця, 2023. С. 46–47. URL: http://repository.hneu.edu.ua/bitstream/123456789/29041/1/foss-2023-theses.pdf (дата звернення: 25.04.2023).

2. Афонін Ю. С., Журавська І. М. Мобільний застосунок для моніторингу здоров'я у вигляді казуальної гри на платформі Unity. *Інформаційні технології та інженерія* : тези доп. Всеукр. наук.-практ. конф. Миколаїв, 07–10 лют. 2023 р. Миколаїв : Чорном. нац. ун-т ім. Петра Могили, 2023. С. 90–92.

3. A study of fitness trackers and wearables. URL: https://www.hfe.co.uk/blog/a-study-of-fitness-trackers-and-wearables/ (дата звернення: 25.04.2023).

4. Baek K., Ha E. Mobile-based digital healthcare hlatforms: Smart wellness. *Archives of Design Research*. Vol. 34, 2021. P. 101–113. DOI: 10.15187/adr.2021.02.34.1.101.

5. More P., Phalnikar R. Generating UML diagrams from natural language specifications. *International journal of applied information systems.* 2012. Т. 1, № 8. P. 19–23. DOI: 10.5120/ijais12-450222.

6. Desai V., Gupta A., Andersen L., et al. Stress-reducing effects of playing a casual video game among undergraduate students. *Trends in Psychol.* 2021. Vol. 29. P. 563–579. DOI: 10.1007/s43076-021-00062-6.

7. Edwards E. A., Lumsden J., Rivas C., et al. Gamification for health promotion: systematic review of behaviour change techniques in smartphone apps. *BMJ Open*. 2016. Vol. 6, Is. 10. DOI: 10.1136/BMJOPEN-2016-012447.

8. Unity Documentation : вебсайт. URL: https://docs.unity3d.com/ScriptReference/ (дата звернення: 05.05.2023).

9. Гончар В. М. Універсальність мови програмування C# для її застосування в різних предметних галузях. *Прикладні аспекти сучасних міждисциплінарних досліджень* : матеріали I Міжнар. наук.-практ. конф. Вінниця, 18 листоп. 2022 р. Вінниця : ДонНУ імені Василя Стуса, 2022. С. 106–108. URL: https://jpasmd.donnu.edu.ua/article/view/12930 (дата звернення: 05.05.2021).

10. Саган О. В. Гейміфікація як сучасний освітній тренд. *Collection of Research Papers Pedagogical sciences*. 2022. Is. 100. С. 12–18. DOI: 10.32999/ksu2413-1865/2022-100-2.

11. Фітнес-застосунок Google Fit. URL: https://www.google.com/fit/ (дата звернення: 05.05.2023).

12. Фітнес-застосунок Zepp Life. URL: https://play.google.com/store/apps/details?id=com.xiaomi.hm.health&hl=uk&gl=US (дата звернення: 05.05.2023).

13. Unity User Manual. URL: https://docs.unity3d.com/Manual/ (Last accessed: 07.05.2023).

14. Unity Hub. URL: https://unity.com/download (дата звернення 07.05.2023).

15. Zenject Framework. URL: https://github.com/modesttree/Zenject (Last accessed: 10.05.2023).

16. Бібліотека DOTween. URL: http://dotween.demigiant.com/ (дата звернення: 10.05.2023).

17. Плагін Unity Android BLE. URL: https://github.com/Velorexe/Unity-Android-Bluetooth-Low-Energy (дата звернення: 10.05.2023).

18. Java Native Interface Specification Contents. URL: https://docs.oracle.com/javase/8/docs/technotes/guides/jni/spec/jniTOC.html (Last accessed: 10.05.2023).

19. BLE Reverse Engineering – Mi Band 5. URL: https://medium.com/@_celianvdb/ble-reverse-engineering-mi-band-5-c3deed12c7 (Last accessed: 31.05.2023).

20. Takoordyal K. Beginning Unity Android Game Development. 2020. P. 270. DOI:10.1007/978-1-4842-6002-9.

ДОДАТОК А
Use Case діаграма

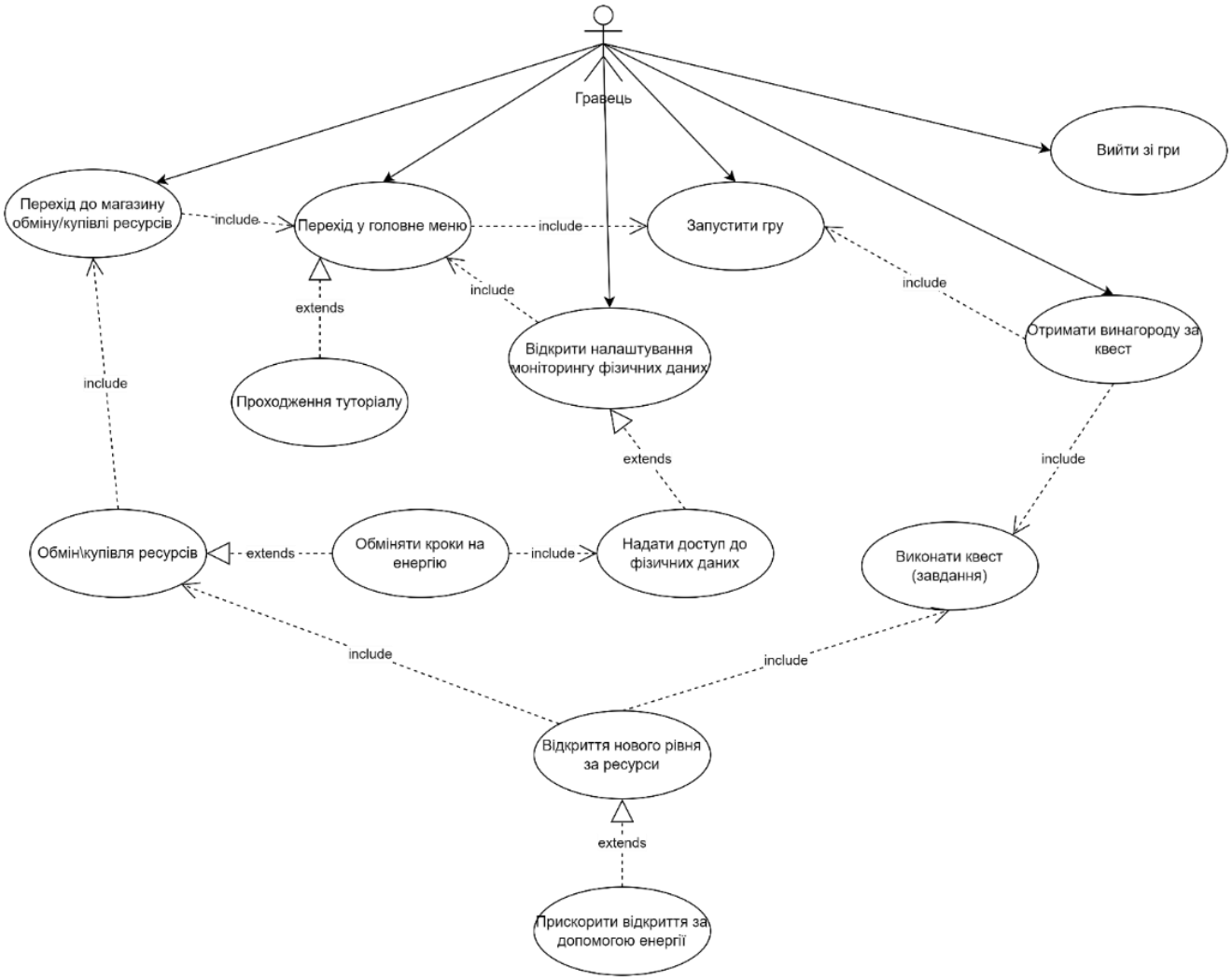


Рисунок А.1

ДОДАТОК Б
Діаграма діяльності для пошуку BLE-пристроїв

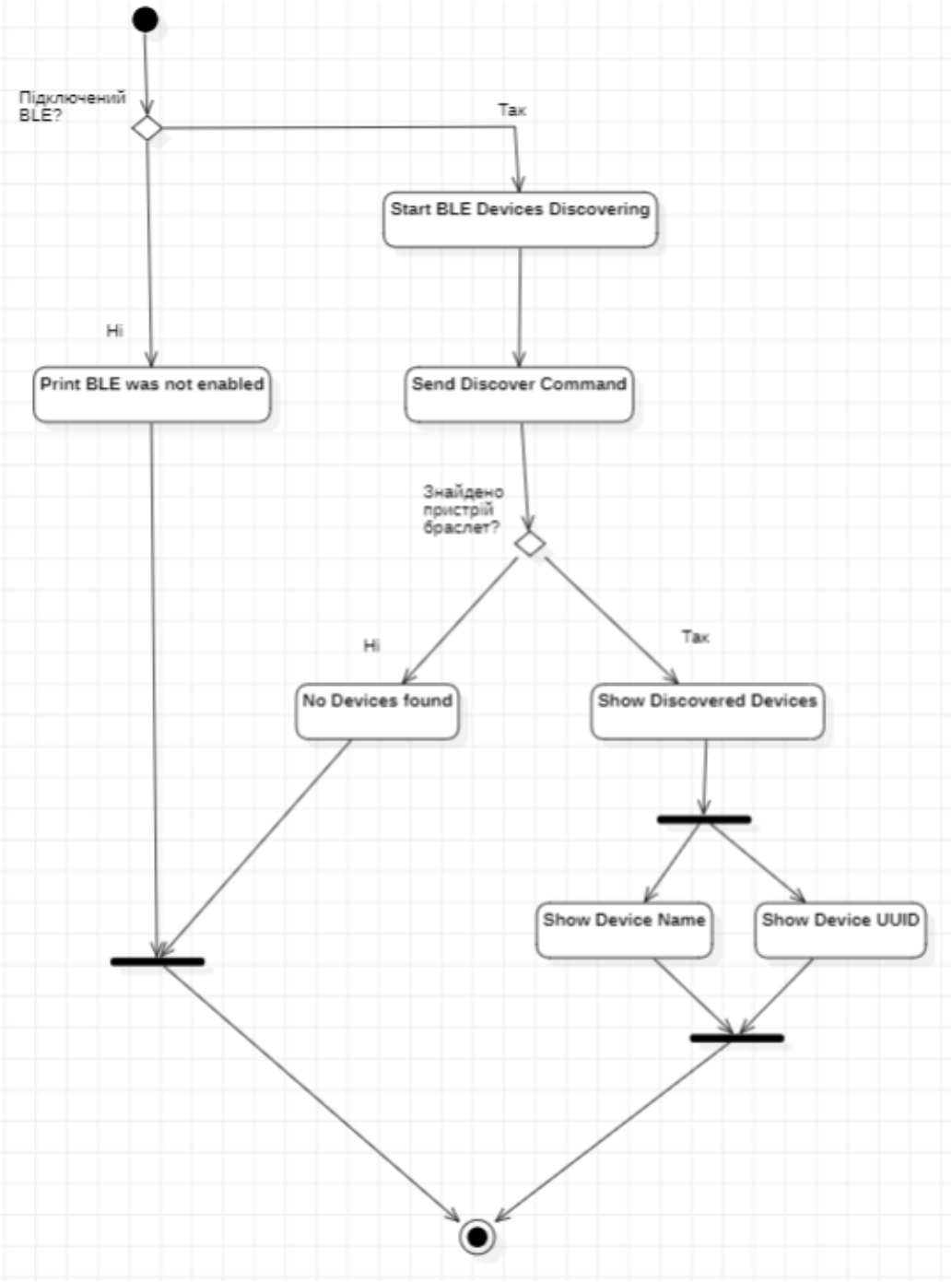


Рисунок Б.1

ДОДАТОК В

Діаграма класів

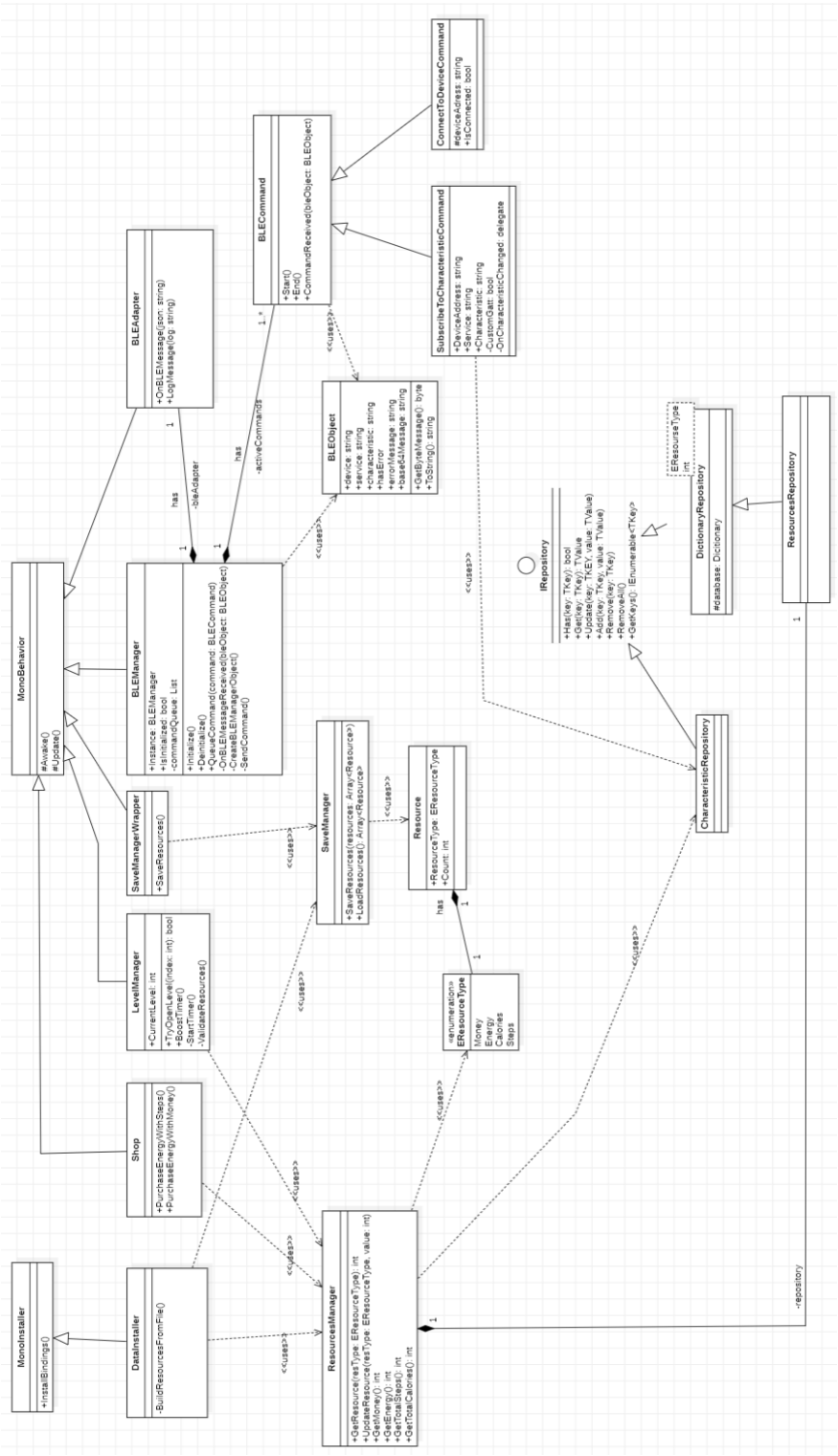


Рисунок В.1