Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра програмного забезпечення

Пояснювальна записка

до магістерської кваліфікаційної роботи

Магістр

ступінь вищої освіти

на тему: «Розробка методу та програмних засобів мобільної системи гейміфікації подій»

Виконав: студент II курсу, групи 1ПІ-18м

спеціальності

121 – Інженерія програмного забезпечення

(шифр і назва напряму підготовки, спеціальності)-

Фролов В.О.

(прізвище та ініціали)

Керівник к.т.н., доц. каф. ПЗ Ракитянська Г.Б.

(прізвище та ініціали)

Рецензент к.т.н., доц. каф. КН Арсенюк І.Р.

(прізвище та ініціали)

Вінниця – 2019 року

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення
Ступінь вищої освіти магістр
Спеціальність 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ_____О.Н. Романюк
“____” _____20___року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Фролову Віктору Олександровичу

1. Тема роботи: Розробка методу та програмних засобів мобільної системи гейміфікації подій
керівник роботи: к.т.н., доцент кафедри ПЗ Ракитянська Г. Б. затверджені наказом вищого навчального закладу від “___”_____20__року №__

2. Строк подання студентом роботи _____

3. Вихідні дані до роботи : Методи геймифікації, технології розробки мобільних додатків, середовище розробки – WebStorm, мова програмування – Javascript, платформа – Android/iOS

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): вступ; аналіз стану питання та постановка задач дослідження; розробка методу та засобів гейміфікації; розробка програмних модулів; висновки; додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень): мета і задачі роботи; аналіз сучасного стану питання; метод гейміфікації подій; модель системи гейміфікації, алгоритми роботи програми, структура інтерфейсу програми, основні результати роботи програми

6. Консультанти розділів магістерської кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада Консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-4	Ракитянська Г. Б., к.т.н., доцент кафедри ПЗ		
5	Бальзан М.В, к.е.н.,доцент кафедри ЕПіВМ		

7. Дата видачі завдання_____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Обґрунтування вибору методу розробки та постановка задачі дослідження		
2	Розробка структури та алгоритмів додатку		
3	Вибір середовища розробки, мови програмування та архітектури додатку		
4	Розробка клієнтської частини		
5	Розробка серверної частини		
6	Тестування роботи додатку		
7	Оформлення матеріалів до захисту МДР		

Студент _____ **Фролов В.О.**
(підпис) (прізвище та ініціали)

Керівник магістерської кваліфікаційної роботи _____ **Ракитянська Г. Б.**
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Магістерська кваліфікаційна робота присвячена розробці системи гейміфікації подій з використанням мобільних засобів комунікації.

Розроблено мобільний додаток клієнт-хмарної архітектури, що дозволяє у зручному вигляді передавати користувачам найновіші пропозиції, акції та відкриває можливість введення ігрової складової в маркетингову стратегію компанії.

Розроблено метод гейміфікації подій та моделі мобільної системи гейміфікації, орієнтованої на використання адаптивних алгоритмів роботи, які забезпечують підбір ігрового контенту з урахуванням активності та можливостей конкретного користувача.

Програма розроблена під операційні платформи Android/iOS з використанням середовища розробки WebStorm та мови програмування Javascript.

ANNOTATION

Master's qualification work is devoted to the development of event gamification system using mobile communication tools.

A mobile application of client-cloud architecture has been developed, which allows to send the latest offers, promotions to the users and opens the possibility of introducing the game component into the company's marketing strategy.

The method of gamification of events and the model of the mobile gamification system, focused on the use of adaptive algorithms that provide the selection of game content taking into account the activity and capabilities of a particular user, have been developed.

The program is designed for Android / iOS operating platforms using the WebStorm development environment and Javascript programming language.

ЗМІСТ

ВСТУП

Обґрунтування вибору теми дослідження. Організація нових каналів маркетингової комунікації з клієнтами відіграє ключову роль в ефективному веденні підприємницької діяльності на внутрішньому та міжнародному ринках[1].

Для сучасного бізнесу маркетингові канали виконують такі функції:

- збір датасетів про потенційних та майбутніх клієнтів, постачальників, реалізаторів, конкурентів тощо;
- стимулювання збуту;
- трансфер поживної цінності;
- забезпечення прав передачі власності на товар.

Канали, що відносяться до дистрибуції та реалізації, також можуть скоротити число контактів на ринку, що зменшує суму витрат на підтримування каналу.

Все більше компаній та брендів шукають нові ефективні канали маркетингового залучення користувачів через ІТ-технології, підключаючи такі технології як: нейронні-мережі, «біг-дата», персоналізацію [2]. Відносно новим напрямком у маркетинговій взаємодії з користувачем є гейміфікація.

Гейміфікація – це перенесення ігрових механізмів, механік, прийомів на неігрові процесів для залучення користувачів до вирішення поставлених завдань[3]. Гейміфікація широко використовується у таких категоріях:

- клієнтська взаємодія;
- повернення грошових інвестицій;
- отримання якісних даних;
- навчання;
- спорт.

Проведені дослідження [1-3] показують позитивну динаміку активності користувачів після введення гейміфікації подій. Тому розробка мобільної системи гейміфікації подій є актуальною задачею, затребуваною відділами маркетингу сучасних фірм.

Мета та завдання дослідження. Метою роботи є розширення маркетингового комунікаційного функціоналу системи через введення гейміфікації подій з використанням мобільної системи, що дозволить зацікавити нових та активізувати наявних користувачів системи у процесі вирішення робочих завдань.

Основними **завданнями** дослідження є:

– удосконалення методу гейміфікації подій шляхом введення адаптивного підходу до вибору складності завдань у процесі реалізації ігрової ситуації;

– розробка моделей та структур системи гейміфікації подій;

– розробка алгоритмів роботи системи гейміфікації подій;

– розробка програмних модулів системи з використанням технологій розробки мобільних додатків під операційні платформи Android/iOS;

– розробка програми клієнт-хмарної архітектури;

– проектування та розробка хмарного сховища даних;

– розробка засобів адміністрування системи;

– тестування розробленого мобільного застосунку.

Об’єкт дослідження – процес гейміфікації подій з використанням технологій розробки мобільних додатків.

Предмет дослідження – програмні засоби реалізації процесу гейміфікації подій з використанням технологій розробки мобільних застосунків під операційні платформи Android/iOS.

Методи дослідження. У процесі досліджень використовувались:

- методи прикладної теорії інформації і теорії алгоритмів для розробки алгоритмів роботи програми;
- методи проектування програмного забезпечення для виконання розробки програмного мобільного додатку;
- комп’ютерне моделювання для аналізу та перевірки отриманих теоретичних положень.

Наукова новизна одержаних результатів:

1. Подальшого розвитку дістав метод гейміфікації подій, у якому, на відміну від існуючих, використовуються адаптивні алгоритми підбору рівня завдань, що дозволяє підвищити активність користувачів шляхом їх зацікавлення процесом проходження ігрових завдань.
2. Подальшого розвитку дістали моделі системи гейміфікації подій, які, на відміну від існуючих, орієнтовані на адаптивний підхід підбору завдань з урахуванням рівня і активності користувача, що дозволяє розширити комунікаційну взаємодію користувачів у процесі розвитку маркетингової стратегії компанії.

Практична цінність отриманих результатів. Практичне значення отриманих результатів полягає в тому, що на основі проведених теоретичних досліджень і отриманих наукових результатів розроблено мобільний додаток для гейміфікації подій, призначений для більш ефективного маркетингового залучення користувачів до цільового продукту.

Зв'язок роботи з науковими програмами, планами, темами. Робота виконувалася згідно плану виконання наукових досліджень на кафедрі програмного забезпечення

Особистий внесок здобувача. В публікації [4] автором розроблено архітектуру мобільного сервісу та алгоритм авторизації користувачів системи гейміфікації подій. Усі наукові результати, викладені у магістерській кваліфікаційній роботі, отримані автором особисто.

Достовірність теоретичних положень підтверджена результатами тестування розробленого мобільного додатку.

Апробація матеріалів магістерської кваліфікаційної роботи. Результати роботи доповідалися на Всеукраїнській науково-практичній Інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи - 2019».

Публікації. Результати роботи опубліковані в тезах доповіді «Розробка автоматизованої мобільної системи гейміфікації подій» [4].

Структура та обсяг роботи. Робота містить вступ, п’ять розділів, висновки, перелік посилань, додатки. У першому розділі визначено загальний стан питання по розробці мобільних додатків для гейміфікації подій та розглянуто аналоги розробки, обґрунтовано вибір засобів програмної реалізації системи гейміфікації. У другому розділі описано метод та моделі системи гейміфікації подій, структури та алгоритми роботи системи. У третьому розділі проведено програмну реалізацію системи за клієнт-хмарною технологією з використанням засобів реалізації мобільних застосунків під операційні платформи Android/iOS. У четвертому розділі проведено тестування роботи системи гейміфікації подій. У п’ятому розділі проведено економічне обґрунтування доцільності та економічної вигоди даного програмного продукту. Перелік посилань містить 20 джерел. У додатках міститься технічне завдання на магістерську кваліфікаційну роботу, лістинг коду та ілюстративний матеріал до захисту роботи.

1 АНАЛІЗ СТАНУ ПИТАННЯ ГЕЙМІФІКАЦІЇ ПОДІЙ ТА ПОСТАНОВКА ЗАДАЧ ДОСЛІДЖЕННЯ

За останні декілька років гейміфікація набула широкого використання у різних галузях діяльності людини. Велика кількість компанії прагне додавати елементи гейміфікації у свої додатки, адже методи гейміфікації впливають на основні природні людські інстинкти, а саме: на конкуренцію, прагнення досягати кращих результатів, підтвердити свій статус, самовиразитись, на альтруїзм, пошук оптимальних методів розв’язання задач тощо. Тому важливим є використання гейміфікації в маркетингових стратегіях сучасних підприємств.

1.1 Аналіз стану питання гейміфікації подій

Основна стратегія гейміфікації – це надання учаснику ігрового процесу деякого завдання, виконавши яке, він отримає приз чи винагороду. В якості винагороди може бути будь-який заохочувальний бонус: віртуальні бали, рівні чи позиції просування в рейтингу, реальні гроші тощо.

Конкуренція – це один з ключових елементів гейміфікації. Він полягає в наданні всім користувачам можливості бачити винагороди інших, формуванні списків переможців або лідерів змагань. Такі елементи інтерфейсу будуть стимулювати користувача до виконання завдань.

Гейміфікація також широко використовується для навчання. Елементарні прояви гейміфікації можна побачити навіть у таких речах як: дошки пошани, змагання між класами у вивченні навчальних дисциплін тощо.

Гейміфікація може використовуватись у випадках:

1. Формування певних навичок, патернів поведінки.
2. Пояснення та візуалізації таких дій, які важко пояснити традиційними методами.

Використання гейміфікації має певні тенденції для різного роду компаній:

- маленькі стартапи зазвичай замовляють гейміфікацію свого продукту – для них цей процес складається зі створення продукту, який максимально

залучає споживачів та приваблює їх до процесу його покупки та споживання в майбутньому;

- середній бізнес зазвичай гейміфікує маркетинг з метою залучити потенційних споживачів із зазначеного сегмента ринку до бренду, продуктів і послуг;

- великі компанії зазвичай фокусуються на гейміфікації робочого процесу з метою розвитку співробітників та культивування почуття солідарності і спільності з усією командою.

Гейміфікацію використовують при створенні програм лояльності, в роботі складних інтерфейсів, навчальних процесах, соціальних мережах, а також в рекламних кампаніях, коли необхідно, щоб користувач добре ознайомився з брендом або контентом. Наприклад, «Google» ввела ігрову валюту для своїх співробітників – «Goobles». «FedEx»та деякі авіалінії використовують ігрові симулятори з рейтингами з метою навчання працівників. Косметична компанія «L'Oreal» створила ігри для нових працівників, в яких вони можуть перевірити свої навички і визначити для себе кращу посаду в компанії. «Nike» запустила додаток для телефонів «Nike+» ще в 2012 році.

В українській компанії «Interpipe» використовується ігрова валюта «їжачки», яку співробітники можуть заробити за виконання завдань, безпосередньо пов'язаних з виробничим процесом. Зароблені «їжачки» можна витратити на матеріальні подарунки. Компанія «Socar» була однією з перших українських паливних операторів, хто застосував технології гейміфікації. В рамках програми лояльності вона запустила додаток «Level Game», який дозволяє постійним клієнтам за кожну десяту покупку отримувати приз – аналогічний товар або знижку. При здійсненні покупок клієнт отримує віртуальні стікери, які потім складаються в купон, що дає право на отримання призу.

Таким чином, можна зробити висновок, що широта та комплексність застосування такого інструменту, як гейміфікація, має потенціал для розбудови потужної стратегії збільшення продажів будь-якого продукту та розвитку іміджу компанії.

1.2 Порівняльний аналіз аналогів

Попри те, що в наш час все більше компаній залучають до своїх додатків гейміфікацію, сьогодні таких додатків на ринку не так багато. Основним недоліком наявних програм є те, що для повного функціоналу потрібно придбати робочу версію додатку або щомісячно вносити плату за його користування. Також часто незрозумілим та незручним для користувачів є інтерфейс відомих додатків.

«Habitica – Gameficate your tasks» – це щоденний помічник, який допомагає додавати гейміфікацію у повсякденне життя. В рамках програми можна спостерігати за своїми звичками, виконувати завдання та піднімати свій власний рівень. Гра допомагає з мотивацією для виконання повсякденних завдань. Інтерфейси «Habitica» зображенно на рис 1.1.

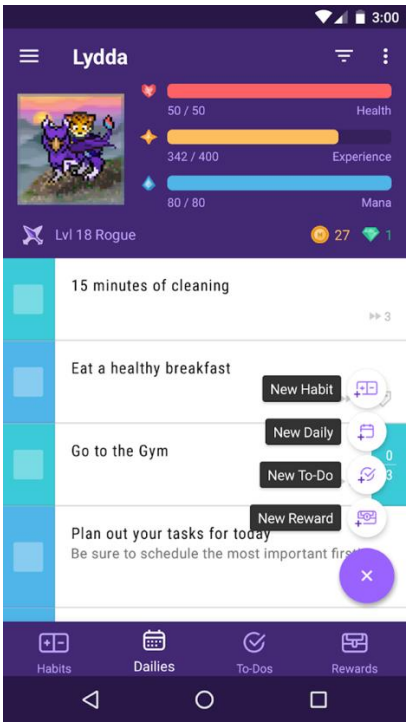


Рисунок 1.1 – Головна сторінка додатку «Habitica»

Інший популярний додаток з елементами гейміфікації – «Gamified».

«Gamified» – це освітня платформа, яка використовує гейміфікацію з метою мотивації участі під час занять. Додаток бере за основу те, що постійне

навчання демотивує студентів на учнів. Але з введенням гейміфікації мотивація до навчання повертається.

Платформа дозволяє отримувати ігрові бали за виконання навчальних завдань та втрачати їх за невиконання завдань або за регулярне невідвідування занять. Також у додатку присутня система рейтингів.

У додатку «Gamified» також можна переглади онлайн-лекції та виконувати домашнє завдання.

Програма «Gamified» рекомендована для використання в школах та університетах штату Пенсильванія США.

Інтерфейс додатку «Gamified» зображений на рис. 1.2.

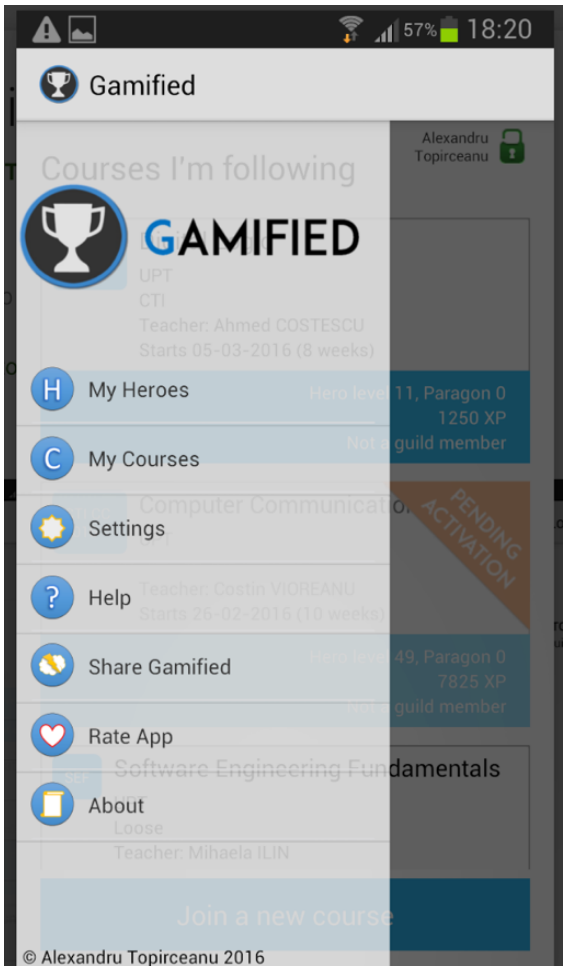


Рисунок 1.2 – Головна сторінка додатку «Gamified»

Додатком, який показує залучення гейміфікації у маркетинг великих компаній, є мобільний додаток «Starbucks».

Крім основних функції по перегляду локації кафе та онлайн-оплати, додаток має підсистему «Rewards», яка дозволяє користувачам збирати «зірочки» за відвідування закладів мережі та обмінювати ці «зірочки» на спеціально підготовленні товари мережі.

Для збирання «зірочок» після відвідування та отримування чеку закладу досить лише завантажити чек через мобільний додаток та отримати «зірочки» відповідно до витраченої суми.

Інтерфейс «Starbucks» зображено на рис 1.3.

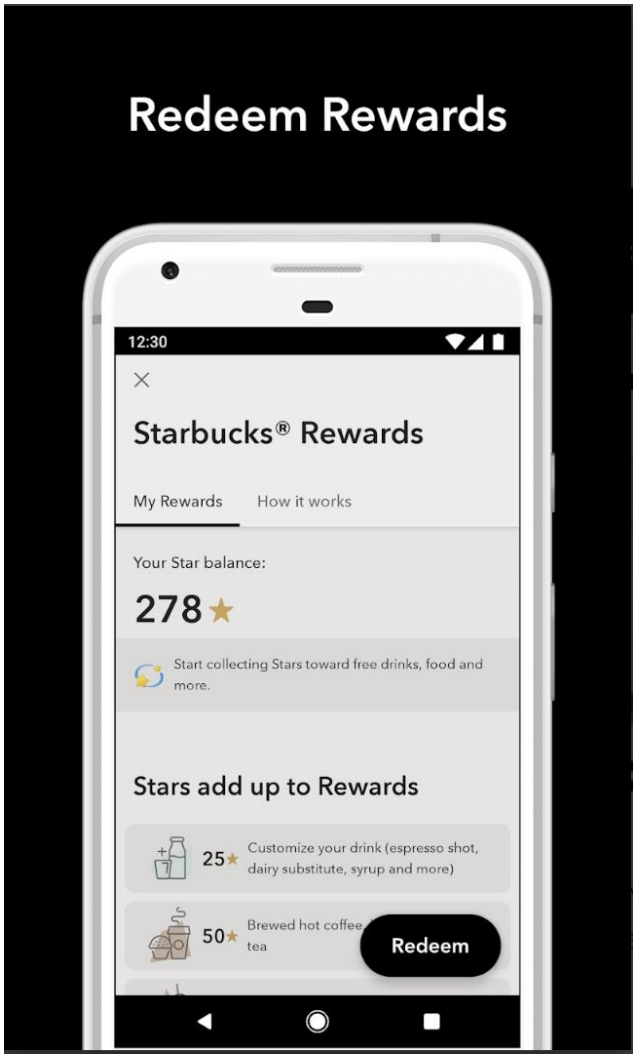


Рисунок 1.3 – Сторінка додатку «Starbucks» з винагородами

Проаналізувавши аналоги, узагальнимо їхні функціональні можливості та недоліки. Результати аналізу, зведені в табл. 1.1, враховуємо при створенні власного програмного додатку з назвою «Qarat Play».

Таблиця 1.1 – Порівняльні характеристики аналогів

Критерій	«Habitica»	«Gamified»	«Starbucks»
Зручний інтерфейс	+	-	+
Не вимагає доступу до карток чи паролів	+	+	-
Наявність системи рейтингів користувачів	+	+	-
Високий рівень безпеки	-	-	+
Повний безкоштовний функціонал	-	-	+
Не віртуальні нагороди	-	-	+

Таблиця порівняльних характеристик (табл. 1.1) показала, що розробка власного додатку з гейміфікацією подій є доцільною. В результаті отримаємо

мобільну розробку, що покриває недоліки існуючих рішень і забезпечує більшу інтерактивність взаємодії з користувачем, а, отже, і отримання задоволення від гри.

1.3 Аналіз середовищ розробки мобільного додатку

Для розробки мобільних додатків розглянемо такі середовища розробки:

- XCode;
- Webstorm.

JetBrains WebStorm – це інтегроване середовище розробки на JavaScript, CSS та HTML від компанії JetBrains, розроблене на базі платформи IntelliJ IDEA.

WebStorm забезпечує автоматичне оновлення, аналіз коду, навігацію за кодом, реконструкцію, інтеграцію системи управління версіями. Важливим вмістом інтегрованої середини розробки WebStorm є робота з проектами. Підтримується множинна вкладеність. Користувачі можуть розширити свої можливості, встановивши плагіни, створені для платформи IntelliJ, а також вони можуть писати власні плагіни.

Особлиовсті WebStorm:

- Модифікація файлів .css, .html, .js з одночасним переглядом результатів (англ. Live Edit, в деяких джерелах ця функціональність називається «редагування файлів на льоту» або «в реальному часі» або «без перезавантаження сторінки»).
- Інтеграція з системами управління версіями: Subversion, Git, GitHub, Perforce, Mercurial, CVS підтримуються з коробки з можливістю створення списків змін і відкладених змін.
- Інтеграція з системами стеження за помилками та системами стеження за службовими повідомленнями.
- Віддалене розгортання за протоколами FTP, SFTP на монтованих мережевих дисках з можливістю автоматичної синхронізації.
- Налагодження коду на JavaScript.

- Можливості Zen Coding і Emmet.
- Підтримка HTML5.
- Підтримка JSDoc.
- WebStorm підтримує налагодження додатків в node.js. Також підтримується повний набір функцій редагування додатків на Javascript, як для виконання на сервері, так і в браузері: автодоповнення, навігація по коду, рефакторинг і перевірка на помилки.
- Інтеграція з системами відстеження випусків: Atlassian JIRA, JetBrains YouTrack, Маяк, Pivotal Tracker, GitHub, Redmine, Trac.
- Мови стилів LESS, Sass, SCSS і Stylus, які розширюють можливості описів стилів в CSS

Xcode – це інтегроване середовище розробки (IDE) для macOS, що містить набір інструментів розробки програмного забезпечення, розроблених Apple для розробки програмного забезпечення для macOS, iOS, iPadOS, watchOS та tvOS. Вперше випущений у 2003 році, останній стабільний реліз - версія 11.0 і доступний через Mac App Store безкоштовно для користувачів macOS Mojave [7]. Зареєстровані розробники можуть завантажувати попередні випуски та попередні версії пакету через веб-сайт Apple Developer.

Особливості XCode. Xcode підтримує вихідний код для мов програмування C, C ++, Objective-C, Objective-C ++, Java, AppleScript, Python, Ruby, ResEdit (Rez) та Swift з різними моделями програмування, включаючи, але не обмежуючись ними, Карбон та Ява. Треті сторони додали підтримку GNU Pascal,Free Pascal, Ada, C #, Perl, та D.

Xcode може створювати жирові бінарні файли, що містять код для декількох архітектур з виконуваним форматом Mach-O. Вони називаються універсальними бінарними файлами, які дозволяють запускати програмне забезпечення як на платформах PowerPC, так і на базі Intel (x86) і які можуть включати як 32-бітний, так і 64-бітний код для обох архітектур. Використовуючи SDK iOS, Xcode також може використовуватися для компіляції та налагодження програм для iOS, які працюють на процесорах архітектури ARM.

Xcode включає інструмент GUI Instruments, який працює над динамічною рамкою трасування, DTrace, створеною Sun Microsystems і випущеною у складі OpenSolaris.

1.4 Постановка задач роботи

На основі проведеного аналізу стану питання перспектив розробки додатків з елементами гейміфікації на основі мобільної системи у магістерській кваліфікаційній роботі потрібно виконати такі задачі:

- удосконалити метод гейміфікації подій шляхом введення адаптивного підходу до вибору складності завдань у процесі реалізації ігрової ситуації;
- розробити моделі та структури системи гейміфікації подій;
- розробити алгоритми роботи системи гейміфікації подій;
- розробити програмні модулі системи з використанням технологій розробки мобільних додатків під операційні платформи Android/iOS;
- розробити програму клієнт-хмарної архітектури;
- спроектувати та розробити хмарне сховище даних;
- проаналізувати особливості та розробити структуру бази даних для мобільного додатку;
- розробити базу даних для показу рейтингу учасників системи;
- розробити засоби адміністрування системи;
- протестувати розроблений мобільний застосунок.

1.5 Висновки

У першому розділі розглянуто особливості процесу гейміфікації подій. Проведено порівняльний аналіз функціоналу сучасних додатків з елементами гейміфікації: «Habitica», «Gamified», «Starbucks». Проаналізовано перспективність створення власного додатку. Розглянуто особливості середовищ розробки мобільного програмного продукту. З урахуванням проведеного аналізу було сформульовано задачі магістерської кваліфікаційної роботи.

2 РОЗРОБКА МЕТОДУ ТА ЗАСОБІВ ГЕЙМІФІКАЦІЇ ПОДІЙ

З метою підвищення результативності розроблюваного методу гейміфікації подій алгоритми його функціонування будуть орієнтовані на адаптацію до дій користувача, що дасть змогу підлаштовувати складність та часову затратність обраних завдань під активність та спроможність конкретного користувача. Такий підхід дасть змогу максимально зацікавити й активізувати користувачів системи й мотивувати їх на проходження нових рівнів ігрових завдань, долучаючись до просування за маркетинговою стратегією, захоплюючись ігровим контентом процесу гейміфікації.

2.1 Розробка інформаційного забезпечення системи гейміфікації

До вхідних даних системи гейміфікації відносимо інформаційний контент, який зберігається в базі даних:

- завдання для користувача;
- рівні складності завдань;
- систему бонусних заохочень;
- базу рейтингу користувачів.

Вихідними даними є надані користувачем результати виконання завдання та результати оцінювання дій користувача з прийняттям рішення про бонусне заохочення учасника ігрового процесу.

Система оцінки використовує алгоритм, який збільшує винагороду в залежності від важкості виконаного завдання. Всі завдання поділені за рівнями важкості від 1 до 6. Алгоритм підбору завдання враховує активність кожного користувача та якісний рівень результатів виконаних завдань, що дозволяє налаштувати систему під конкретного користувача та забезпечити йому комфортні умови інтерактивної взаємодії з системою гейміфікації подій, що підвищує зацікавлення користувача гейміфікованим процесом та сприяє зменшенню кількості відмов через низький рівень результатів [6].

2.2 Розробка методу та моделей система гейміфкації подій

Метод гейміфікації подій включає таку послідовність етапів:

1. Авторизація користувача в мобільній системі.
2. Створення завдань, ігорового контекту, класифікованих за типом та складністю. Серед типів виділяють завдання: ігрові, інтелектуальні, сюжетні, з елементами гумору. Ці типи завдань також можуть поділятися на класи швидкої реалізації і ті, що потребують певних часових витрат на виконання.
3. Вибір стратегії ведення процесу гейміфікації з використанням адаптивних алгоритмів підбору завдань. При реалізації адаптивних алгоритмів враховуються зацікавленість й активність користувача в процесі проходження завдань, що впливає на автоматичний підбір завдань різного рівня складності та очікуваної тривалості його виконання.
4. Виконання завдань користувачем та викладення результатів у середовищі мобільного додатку системи гейміфікації.
5. Оцінювання результатів виконаної користувачем роботи.
6. Надання бонусів та інших засобів заохочення учасників гейміфікованого процесу маркетингової стратегії розвитку компанії.

Модель процесу гейміфікації подій зображена на рис. 2.1.

Система гейміфікації складається з мобільної програми, орієнтованої на користувача, та веб-додатку для адміністрування. Адміністратор створює/завантажує завдання та перевіряє (надає доступ до перевірки) правильності його виконання. Користувач системи отримує і виконує завдання, завантажує в систему результати виконання поставленої задачі, отримує оцінку результатів і бонуси, які може використати чи матеріалізувати за доступною системою заохочень. Система збереження даних використовує хмарні сховища.

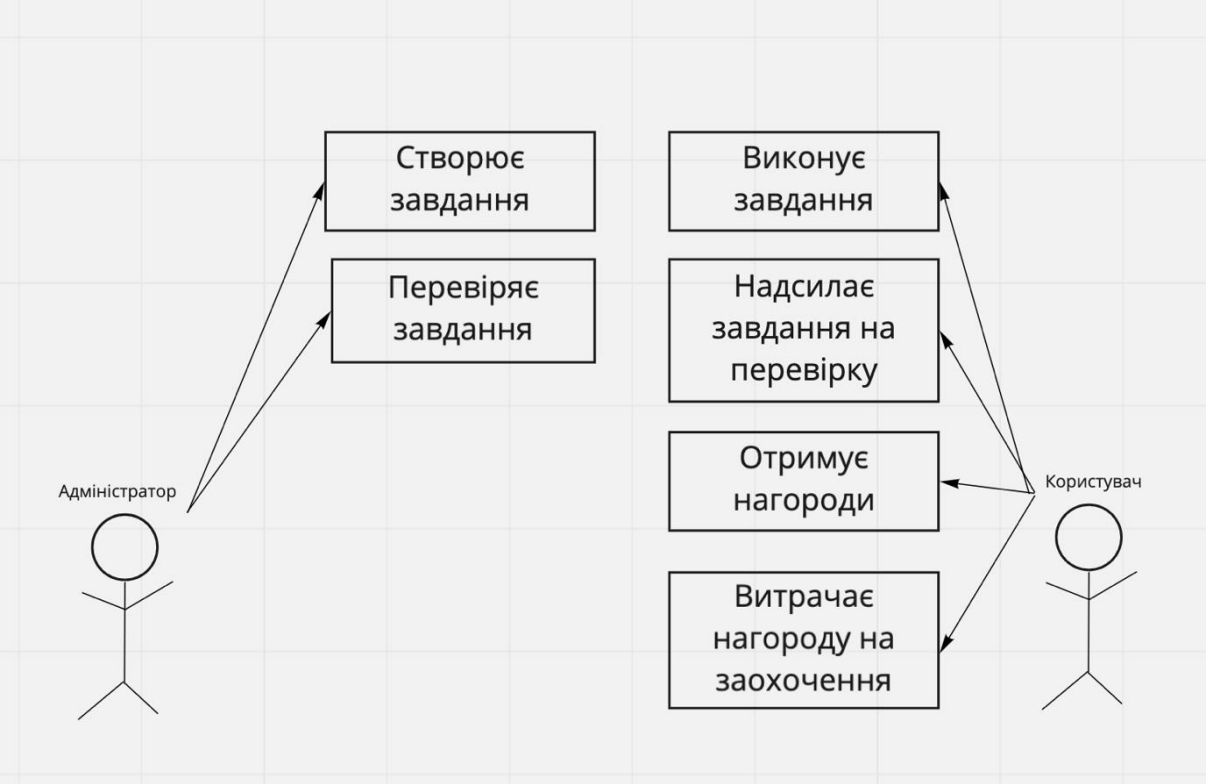


Рисунок 2.1. – Узагальнена модель процесу гейміфікації подій

Ключові елементи моделі системи гейміфікації зображено на рис. 2.2.

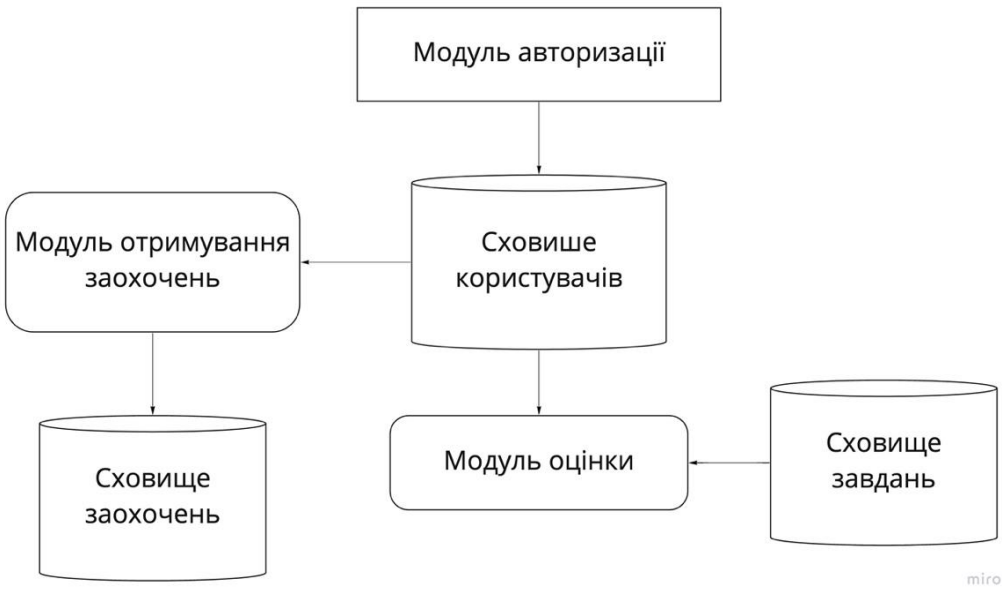


Рисунок 2.2. – Модель системи гейміфікації

Модуль авторизації забезпечує коректне входження користувачів у систему та реалізує засоби захисту даних. Авторизовані користувачі потрапляють на свою сторінку, де отримують чергове завдання зі сховища завдань. Після його виконання користувач завантажує результати в систему, де їх опрацьовує модуль оцінки і видає користувачеві відповідне бонусне заохочення ідентифікованого змісту й форми. Отримані бонуси користувач зможе використати на інших етапах проходження ігрового процесу, що дає змогу мотивувати учасника гри на участь в наступних ігрових контестах.

2.3 Розробка алгоритмів роботи системи гейміфікації

Алгоритм процесу гейміфікації події наведено на рис 2.3.

Рисунок 2.3 – Алгоритм методу гейміфікації

Для підбору чергового завдання використовується адаптивний алгоритм зміни важкості завдання в залежності від успішності виконання попередніх задач [7]. Адаптивний алгоритм роботи системи зображено на рис 2.4.

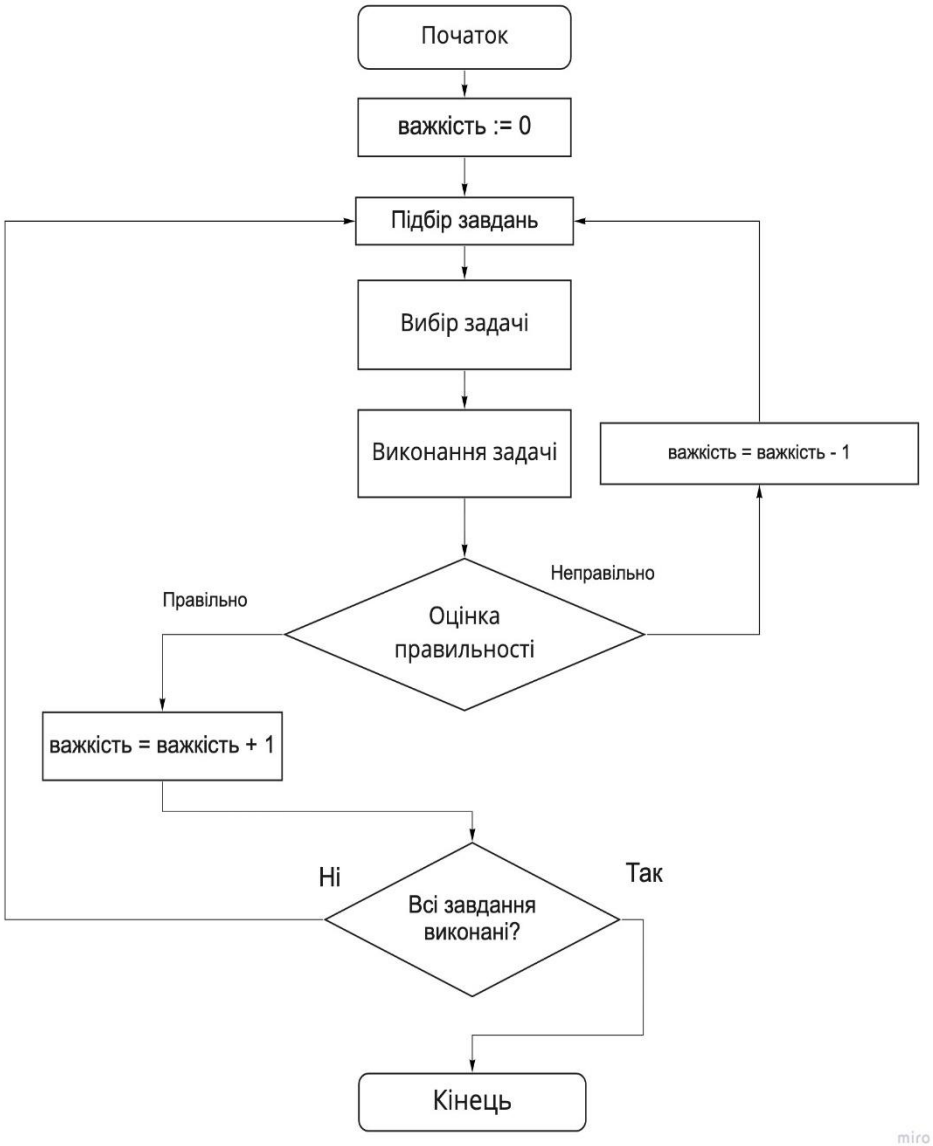


Рисунок 2.4 – Адаптивний алгоритм вибору складності завдання

2.4 Розробка інтерфейсу мобільного застосунку

Розробка інтерфейсів починається з розробки навігаційного меню, яке буде включати основні сторінки для навігації. Структура меню зображена на рис 2.5.

Рисунок 2.5 – Головне навігаційне меню програми

Навігаційний пункт «завдання» буде відображати списки завдань, доступних для виконання, завдань у перспективі та вже виконаних завдань. Інтерфейс доступу до модуля візуалізації завдань зображено на рис 2.6.

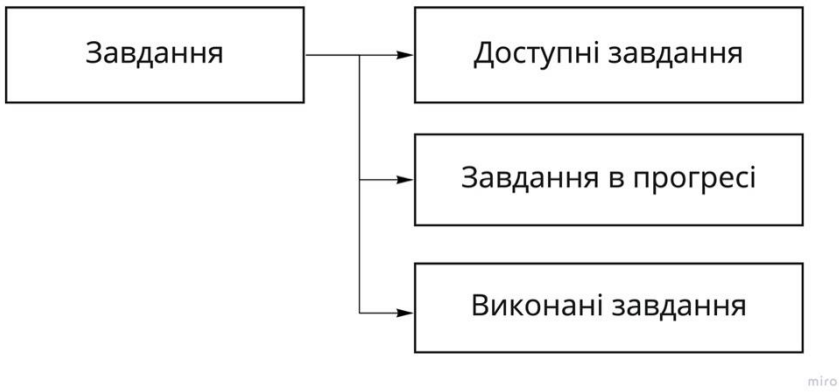


Рисунок 2.6 – Інтерфейс доступу до модуля візуалізації завдань

Навігаційне пункт «Турнір» відображає список користувачів з переліком їх нагород.

Навігаційне пункт «Досягнення» відображає список виконаних та доступних досягнень у додатку. Структуру інтерфейсу модуля «Досягнення» зображено на рис 2.7.

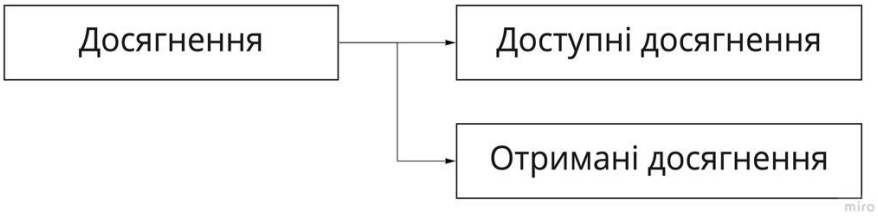


Рисунок 2.7 – Структура інтерфейсу модулю «Досягнення»

Навігаційний пункт «Магазин» буде відображати списки доступних заохочень для купівлі за віртуальну валюту, отриману при виконанні завдань. Структуру інтерфейсу модуля «Магазин» зображено на рис 2.8

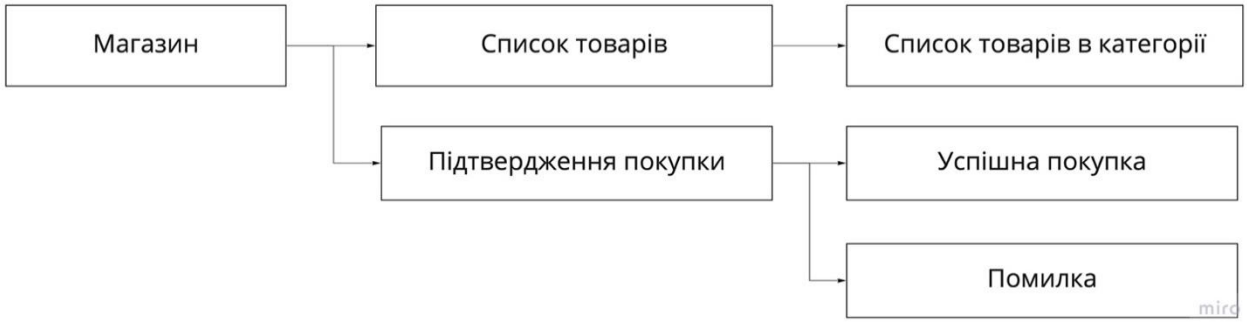


Рисунок 2.8 – Структура інтерфейсу модуля «Магазин»

Навігаційний пункт «Профіль» буде відображати профіль користувача, його персональну інформацію, надавати можливість зміни цієї персональної інформації та можливість перегляду придбаних у магазині заохочень. Структуру інтерфейсу модуля «Профіль» зображено на рис 2.9.

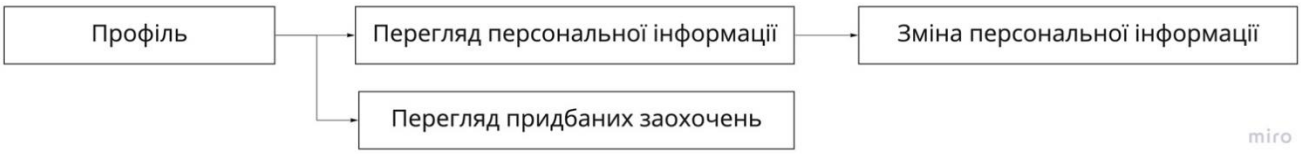


Рисунок 2.9 – Структура інтерфейсу модуля «Профіль»

2.5 Розробка структури бази даних системи гейміфікації подій

База даних (англ. database) – це певна сукупність структурованих даних, організованих відповідно до концепції, яка описує характеристики та властивості цих даних і зв'язки між їх елементами.

У процесі проектування структури бази даних застосовували метод семантичного моделювання. Семантичне моделювання є моделюванням структури, опираючись на зміст самих даних. Як інструмент семантичного моделювання використовувалися різні варіанти діаграм сутність-зв'язок (ER – Entity-Relationship).

Сутність – це вид однотипних об'єктів, інформація про яких повинна бути записана в моделі. Примірник сутності – це конкретний представник даної сутності. Атрибут сутності – це іменована характеристика, що є деякою властивістю сутності. Ключ сутності – це ненадлишковий набір атрибутів, значення яких у сукупності є унікальними для кожного екземпляра сутності. Зв'язок – це деяка асоціація між двома сутностями. Одна сутність може бути пов'язана з іншою сутністю або сама з собою.

Існує декілька типів зв’язку:

- Один-до-одного це означає, що логічна одиниця першої сутності (лівої) пов'язана з логічною одиницю іншої сутності (правої). Зв'язок один-до-одного найчастіше свідчить про те, що ми маємо лише одну сутність, але неправильно розділену на дві.
- Один-до-багатьох це означає, що логічна одиниця першої сутності (лівої) пов'язана з багатьма одиницями іншої сутності (правої). Це найбільш популярний тип зв'язку.
- Багато-до-багатьох означає, що кожна одиниця першої сутності може бути пов'язана з декількома одиницями іншої сутності, і кожен примірник іншої суті може бути пов'язаний з декількома примірниками першої сутності.

Розглянемо структуру бази даних для заданої предметної області (рис. 2.10).

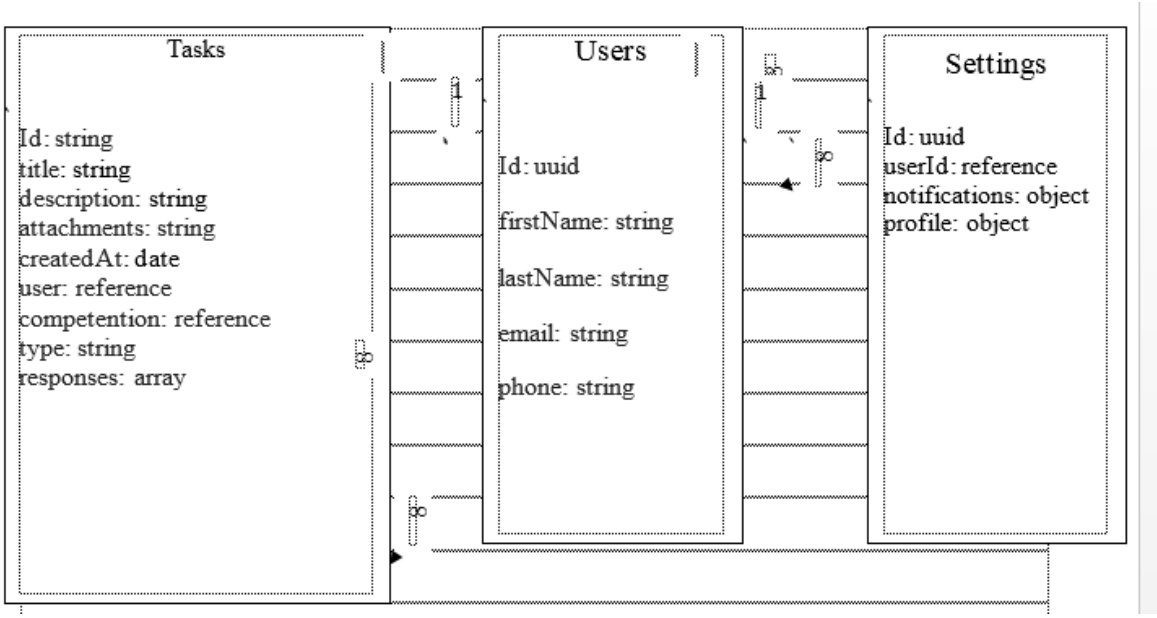


Рисунок 2.10 – Структура бази даних системи гейміфікації подій

Tasks – таблиця, що містить всі необхідні дані про завдання, які додані користувачами на модерацію.

Settings – таблиця, в якій зберігаються налаштування користувачів системи.

Users – таблиця, що зберігає інформацію про всіх користувачів системи.

2.6 Висновки

У другому розділі магістерської кваліфікаційної роботи виконано постановку вимог до системи гейміфікації подій.

Визначено інформаційні об’єкти системи та розглянуто особливості використання системи.

Розроблено метод гейміфікації подій, орієнтований на виконання адаптивних алгоритмів комунікації з користувачами.

Розроблено діаграму бази даних, визначено основні сутності системи та відношення між ними.

3 РОЗРОБКА МОБІЛЬНОЇ ПРОГРАМИ СИСТЕМИ ГЕЙМІФІКАЦІЇ

3.1. Обґрунтування необхідності використання концепції MVC фреймворку React Native в проектуванні системи

Архітектурний шаблон модель–вигляд–контролер (MVC) поділяє програму на три незалежні, але взаємопов'язані частини, розподіляють функції між компонентами додатку. Модель (Model) використовується для зберігання даних і забезпечує інтерфейс до них. Вигляд (View) надає представлення цих даних користувачеві. Контролер (Controller) виконує керування компонентами, отримує події у вигляді змін на дії користувача (зміна положення курсора миші, натискання кнопки, ввід даних в текстове поле) і передає ці дані у модель.

Модель – це центральний компонент шаблону MVC, він відображає поведінку та його функціонал додатку, незалежно від інтерфейсу користувача. Модель стосується прямого управління даними користувача, поведінкою та правилами додатку.

Вигляд -– це будь-яке подання інформації, одержуване на виході з інших частин додатку, наприклад, таблиця чи графік. В один момент часу можуть існувати кілька представлень однієї і тієї ж інформації, наприклад графік та таблиця співробітників

Контролер – це частина логіки, що одержує вхідні дані, оброблює і перетворює на команди для моделі чи вигляду.

Модель інкапсулює ядро даних і функціонал їхньої обробки не залежить від процесу вводу чи виводу даних.

Вигляд може мати декілька взаємопов'язаних областей, наприклад різні таблиці і поля форм, в яких відображаються дані.

У функції контролера входить також і відстеження певних подій, що виникають в результаті взаємодії користувача з інтерфейсом. Контролер дозволяє розробнику структурувати свій код шляхом групування пов'язаних дій в окремий клас.

Події транслюються в різні типи запитів, що потім передаються компонентам або об'єктам, які відповідальним за відображення даних. Відокремлення моделі від вигляду дозволяє використовувати різні компоненти для відображення інформації незалежно. Отже, якщо кінцевий користувач через контролер додасть зміни до моделі даних, то ця інформація, подана одним або декількома компонентами, буде автоматично перебудовано відповідно до тих змін, що відбулися.

Сама очевидна перевага, яку отримуємо від використання концепції MVC – це чіткий поділ логіки подання (інтерфейсу користувача) і логіки програми.

Підтримка різних типів користувачів, які використовують різні типи пристроїв є спільною проблемою наших днів. Наданий інтерфейс повинен відрізнятися, якщо запит приходить з персонального комп'ютера або з мобільного телефону. Модель повертає однакові дані, єдина відмінність полягає в тому, що контролер вибирає різні види для виведення даних.

Крім ізолювання видів від логіки додатки, концепція MVC істотно зменшує складність великих додатків. Код виходить набагато більш структурованим, і, тим самим, полегшується підтримка, тестування і повторне використання рішень.

React дозволяє розробникам створювати великі застосунки, які використовують дані, котрі змінюються з часом, без перезавантаження сторінки. Його мета полягає в тому, щоб бути швидким, простим, масштабованим. React обробляє тільки користувацький інтерфейс у застосунках. Це відповідає видові у шаблоні модель-вид-контролер (MVC), і може бути використане у поєднанні з іншими JavaScript бібліотеками або в великих фреймворках MVC, таких як AngularJS. Він також може бути використаний з React на основі надбудов, щоб піклуватися про частини без користувацького інтерфейсу побудови веб-застосунків. Як бібліотека інтерфейсу користувача React часто використовується разом з іншими бібліотеками, такими як Redux.

В даний час React використовують Khan Academy, Netflix, Yahoo, Airbnb, Sony, Atlassian та інші.

Крім того, всі механізми обробки запитів та інших операцій введення / виведення (I / O) побудовані на подіях. Це означає, що в Node.js немає ніякого способу, щоб заблокувати працюючий в даний момент потік. Кожна операція в Node.js виконується асинхронно. Це є величезною перевагою, особливо якщо ваш код повинен бути побудований на операціях введення-виведення: читання дисків, підключення до бази даних, веб-сервіси і т.д. [13-15]/

В якості фреймворку використовувався React Native – JavaScript бібліотека для створення інтерфейсів від Facebook, яка повинна вирішувати проблеми повного та часткового оновлення вмісту сторінки. Бібліотека також включає в себе набір бібліотек для роботи з запитами – axios, та сховище данних Redux що наслідує архітектуру Flux [10].

3.2 Програмна реалізація структури додатку

При створенні програми на React Native командою react-native init «appName», react native генерується стандартна структура файлів і папок, що являється однією з його переваг, ця структура є загальною для всіх проектів, в ній можна легко зорієнтуватися, дивлячись на чужий код. Структура React Native проекту подана в табл. 3.1.

Оскільки React Native реалізує архітектурий шаблон MVC, для створення повноцінного модуля потрібно створити модель в системі, описати контролер цієї моделі та створити сторінку відображення. Розглянемо приклад створення модуля на основі модуля користувачів системи.

Для генерації додатку для android виконаємо команду cd android && ./gradlew assembleRelease. Данна команда звернеться до основного пакетного менеджеру gradle, підвантажить всі необхідні бібліотеки та запустить завдання на створення виконуваного apk файлу для платформи android.

Процес створення додатку для ios дещо складніший. Перш за все це обумовлено закритістю платформи. Для створення додатків потрібна операційна система зі спеціальною архітектурою – macOS. Та додаток для розробки – Xcode. Відкривши створений додаток у XCode, потрібно авторизуватися в ITunes Connect, створивши набір сертифікатів та ключів. Після цього можна зібрати робочу версію проекту, вибравши відповідний пункт у середовищі розробки. Після проходження всіх необхідних етапів – з'явиться вікно симулятора з робочим додатком [11, 16-19].

Таблиця 3.1 – Структура файлів проекту

Файл/директорія	Призначення
app	Клієнтська частина на JS
api/controllers	Контролери додатку, використовуються для взаємодії з моделлю і обслуговування клієнтських запитів
api/models	Моделі, які описують структури для зберігання даних додатку
api/views	Логіка відповідей на запити
config/	Конфігураційні файли проекту
views/	Відображення додатку, за замовчуванням в якості шаблонізатору використовується JSX
android/	Папка з системними бібліотеками для підтримки android
node_modules/	Зберігаються всі клієнські бібліотеки
Ios/	Папка з системними бібліотеками для підтримки ios

Робота додатку починається з модуля визначення стану користувача, що описується програмним кодом:

```
@codePush(codePushOptions)
export default class Welcome extends Component {
unsubscribe;
async componentDidMount() {
const amplitude = new RNAmplitude();
amplitude.logEvent('session-start', {
        user_id: auth().currentUser && auth().currentUser.uid,
        timestamps: +Date.now(),
        phone: auth().currentUser && auth().currentUser.phoneNumber
 });
 const onboarding = await AsyncStorage.getItem('@onboarding');
       if (!onboarding) {
              await AsyncStorage.setItem('@onboarding', 'true');
               return splashApp();
       }
        this.unsubscribe = auth().onAuthStateChanged(async user => {
        if (!user) {
       return nonloginedApp();
       }
        const userRef = firestore().collection('qi_users').doc(user.uid);
        const userData = await userRef.get();
        const profile = userData.data();
        if (!profile || !profile.nickname) {
       return Navigation.setRoot({
       root: {
       stack: {
       children: [{
        component: {
       name: 'Registration'
       }
```

```
            }]
            } } })
            }
            loginedApp();
            });
            }

        componentWillUnmount(): void {
            this.unsubscribe && this.unsubscribe()
        }

        render() {
            return ( <Loader/> );
        }
}
```

3.3 Розробка модуля реєстрації

Модуль реєстрації дозволяє користувачеві зареєструватись в системі, ввівши номер телефону та підтверджувальне смс-повідомлення. Для реалізації використовується хмарний сервіс Firebase Auth та хмарне сховище користувачів Firebase Firestore. З’єднання з хмарним сервісом проходить через бібліотеку з набору firebase. Лістинг коду для реєстрації:

```
export default class Registration extends Component {
  state = {
    nickname: ''
  };

  static options() {
    return {
      topBar: {
        title: {
```

```
            text: 'Логин'
          }
        }
      };
    }

    async save() {
      try {
        const {nickname} = this.state;

        if (this.loading) {
          return;
        }

        this.loading = true;

        const result = await firestore().collection('qi_users').where('nickname', '==',
nickname).get();

        if (result.empty) {
          const uid = auth().currentUser.uid;
          const user = {
            nickname
          };

        let url = await links().getInitialLink();
        if (url) {
          const ID = this.getParameterFromUrl(url, 'invited');
          user.invitedBy = firestore().doc('qi_users/'+ID);
        }

        await firestore().collection('qi_users').doc(uid).set(user, {merge: true});

          loginedApp();
        } else {
          this.loading = false;
          alert('Этот никнейм - уже занят')
        }
      } catch (e) {
        alert(e);
      }
    }

    render() {
```

```
    return (
      <ImageBackground
        source={require('./images/bg.png')}
        imageStyle={{resizeMode: 'stretch'}}
        style={{width: '100%', height: '100%'}}>
        <KeyboardAvoidingView
          style={styles.wrap}
          behavior={Platform.OS === 'ios' ? 'height' : null}
        >
          <Text style={styles.welcome}>
            Придумайте Ваш логин, после нажмите кнопку далее
          </Text>
          <Input
            rightIcon={<MaterialIcon name={'account-circle'} size={32}
color='#E4E4E4'/>}
            placeholder={'Логин'}
            autofocus={true}
            onChangeText={nickname => this.setState({nickname})}
            value={this.state.nickname}
            onSubmitEditing={() => this.save()}
            returnKeyType={'go'}
          />
          {this.state.errorMessage &&
          <Text style={{color: 'red'}}>
            {this.state.errorMessage}
          </Text>
          }
          <Button
            onPress={() => this.save()}
            title={'Далее'}
          />
        </KeyboardAvoidingView>
      </ImageBackground>
    );
  }
}

const styles = StyleSheet.create({
  wrap: {
    flex: 1,
    padding: 24,
    justifyContent: 'space-around',
    paddingBottom: 48
  },
```

```
  welcome: {
    fontFamily: 'Circe-Regular',
    fontSize: 18,
    color: 'rgba(0, 0, 0, 0.35)',
    textAlign: 'center',
    margin: 10
  }
});
```

3.4 Розробка модуля авторизації

Для авторизації користувач вводить свій номер телефону у міжнародному форматі, система запитує у сховища стан користувача, адже користувач може не існувати або бути заблокованим. Після цього система запитує підтвердження та за умови введення правильних даних, записує данні користувача до локального зашифрованого сховища та здійснює навігацію далі. Лістинг коду:

```
import React, {Component} from 'react';
import {StyleSheet, Text, KeyboardAvoidingView, Platform, ImageBackground,
Dimensions,View} from 'react-native';
import MaterialIcon from 'react-native-vector-icons/MaterialIcons';
import {auth} from './services/firebase';
import {Navigation} from 'react-native-navigation';

import {Button, Input} from './components';

export default class SignUp extends Component {
  state = {
    phoneNumber: '',
  };

  showError(text) {
    let errorMessage = '';

    if (text && text.code === 'auth/invalid-phone-number') {
      errorMessage = 'Неправильный номер'
    }

    this.setState({errorMessage});
```

```
  }

  async login() {
    let {phoneNumber} = this.state;
    phoneNumber = '+' + phoneNumber.replace(/[^0-9]/g, '');

    if (phoneNumber.length !== 12) {
      return this.setState({errorMessage: 'Неверный номер'})
    }

    try {
      const confirmResult = await auth().signInWithPhoneNumber(phoneNumber);

      Navigation.push(this.props.componentId, {
        component: {
          name: 'SignUpCode',
          passProps: {
            phoneNumber,
            confirmResult
          }
        }
      })

    } catch (e) {
      this.showError(e);
    }
  }

  render() {
    return (
      <ImageBackground
        source={require('./images/bg.png')}
        imageStyle={{resizeMode: 'stretch'}}
        style={{width: '100%', height: '100%'}}
        onLayout={e => this.setState({
          keyboardVerticalOffset: Dimensions.get('window').height -
e.nativeEvent.layout.height
        })}
      >
        <KeyboardAvoidingView
          style={styles.wrap}
          keyboardVerticalOffset={this.state.keyboardVerticalOffset || 0}
          behavior={Platform.OS === 'ios' ? 'height' : null}
        >
```

```
            <Text style={styles.welcome}>
              Введите ваш номер телефона, после нажмите кнопку далее
            </Text>
            <View style={styles.inputBox}>
              {/*<Text style={styles.input}>*/}
              {/*    KZ*/}
              {/*</Text>*/}
              <Input
                mask={'+7 (799) 999-99-99'}
                rightIcon={<MaterialIcon name={'phone'} size={32}
color='#E4E4E4'/>}
                placeholder={'Номер телефона'}
                // autofocus={true}
                onChangeText={phoneNumber => this.setState({phoneNumber})}
                value={this.state.phoneNumber}
                keyboardType={'phone-pad'}
                onSubmitEditing={() => this.login()}
                returnKeyType={'go'}
                style={{flex:5}}
              />
            </View>
            {this.state.errorMessage &&
              <Text style={{color: 'red'}}>
                {this.state.errorMessage}
              </Text>
            }
            <Button
              onPress={() => this.login()}
              title={'Далее'}
            />
          </KeyboardAvoidingView>
        </ImageBackground>
      );
    }
}

const styles = StyleSheet.create({
  wrap: {
    flex: 1,
    padding: 24,
    justifyContent: 'space-evenly',
    paddingBottom: 48
  },
  welcome: {
```

```
    fontFamily: 'Circe-Regular',
    fontSize: 18,
    color: 'rgba(0, 0, 0, 0.35)',
    textAlign: 'center',
  },
  inputBox: {
    flexDirection: 'row',
  },
  input: {
    flex:1,
    marginRight: 12,
    backgroundColor: '#FBFAFF',
    borderWidth:2,
    borderColor: 'rgba(0,0,0,0.1)',
    borderRadius: 15,
    textAlign: 'center',
    textAlignVertical: 'center',
    fontSize: 18,
    color: 'rgba(0, 0, 0, 0.65)',
    fontFamily: 'Circe-Regular'
  }
});
```

3.5 Висновки

У третьому розділі проведено розробку базових модулів мобільного додатку з боку клієнта. Описаний функціонал системи гейміфікації подій. Описано використання копцепції MVC та фреймворку React Native у процесі розробки системи гейміфікації.

4. ТЕСТУВАННЯ ДОДАТКУ ТА РОЗРОБКА РЕКОМЕНДАЦІЙ КОРИСТУВАЧУ

4.1 Вибір методів тестування програмного забезпечення

Тестування програмного забезпечення – процес технологічного дослідження, спрямований на виявлення інформації про якість програмного забезпечення [20].

Техніка тестування «білої скриньки» досліджує внутрішню поведінку програми, перевіряє правильність побудови елементів програми та регулярність їх взаємодії.

Тестування «білої скриньки» базується на аналізі структури управління програмою. Вважається, що програма повністю перевірена, якщо всі шляхи її контрольного графа перевірені.

Для тестування сформовані наступні параметри тестування:

- перевірено всі незалежні канали програмування;
- існують гілки True, False для всіх логічних рішень;
- виконуються всі цикли (у рамках їх меж і діапазонів);
- аналізує правильність внутрішніх структур даних.

Переваги тестування «білої скриньки»:

- кількість помилок є мінімальним у «центрі» і найбільшим на «периферії» програми;
- попередні припущення щодо ймовірності потоків керування або даних у програмі часто неправильні, отже, типовим шляхом є модель розрахунку, яка погано обробляється;
- при написанні програмного алгоритму у вигляді тексту обраною мовою програмування, можна ввести типові помилки передачі (синтаксичні та семантичні);
- деякі результати програми не залежать від вихідних даних, а залежать від внутрішніх станів програми.

Недоліки тестування «білої скриньки»:

- кількість незалежних маршрутів може бути дуже високою;

- повна перевірка доріжок не гарантує, що програма відповідає оригінальним вимогам;

- помилки, виникнення яких залежить від даних, не можуть бути виявлені.

Метод тестування «чорної скриньки» досліджує функціональні можливості програмної функції у всій області визначення. Тест не враховує внутрішню логічну структуру функцій.

Методи випробування показують:

- як реалізуються функції програми;

- як надходять вхідні дані;

- як досягти результату;

- як зберігається цілісність зовнішньої інформації.

Тестування «чорної скриньки» дозволяє шукати такі категорії помилок:

- неправильні або відсутні функції;

- помилки інтерфейсу;

- помилки у зовнішніх структурах даних або при доступі до зовнішньої бази даних;

- характеристики помилки (необхідна ємність пам'яті тощо);

- помилки ініціалізації та завершення.

Аналізуючи переваги та недоліки методів тестування «чорної скриньки» та «білої скриньки» відповідно до мети програми, для тестування програмного забезпечення було обрано метод «чорної скриньки»[12].

4.2 Тестування розробленого мобільного додатку

Першим кроком тестування додатку є його запуск. Після запуску додатку очікується побачити завантажене головне меню. Результат запуску додатку наведено на рис. 4.1.

Рисунок 4.1 – Початковий екран програми

При першому запуску додаток видає коротку інструкцію по використанню, далі екран логіну на рисунку 4.2

Рисунок 4.2 – Початковий екран

Для початку треба авторизуватися в мобільному додатку за допомогою номера телефона. Після успішної авторизації з’являється головне меню (рис.4.3).

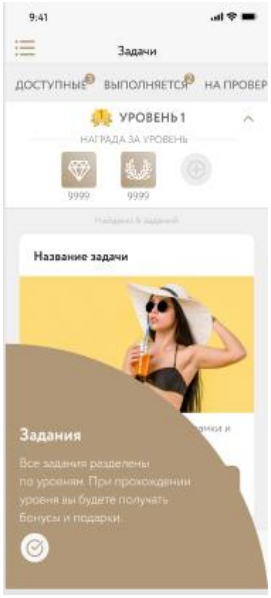


Рисунок 4.3 – Головне меню

На основному екрані відображаються доступні для виконання задачі. Також на основному екрані є нижній бар, для навігації по екранах: «Турнір», «Досягення», «Магазин», «Профіль», Екран «Профіль» зображено на рис. 4.4.

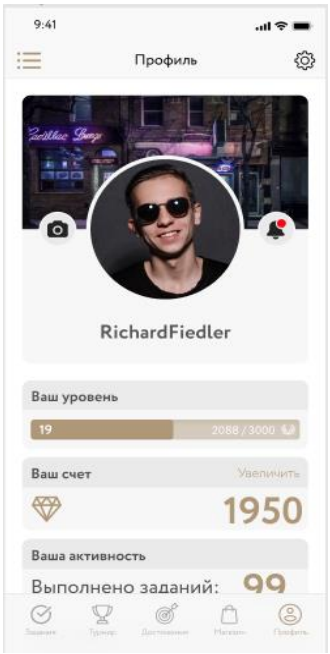


Рисунок 4.4 – Профіль

На екрані «Профіль» можна бачити щось на зразок сторінки в соціальній мережі, там може бути нікнейм користувача, фото профіля, інформація про досвід користувача та інформація про його наявні бонуси. Ці бонуси можна заробити, виконуючи завдання на екрані з задачами. На бонуси, які є у профілі, можна купити справжні речі, які знаходяться у вкладці «Магазин», показаній на рис. 4.5.

Рисунок 4.5 – «Магазин»

Якщо на рахунку достатньо бонусів, можна придбати будь-який товар, який є в наявності в магазині. Доступна велика кількість товарів, які сортовано по категоріях. Вкладка «Турнір» – на рисунку 4.6.

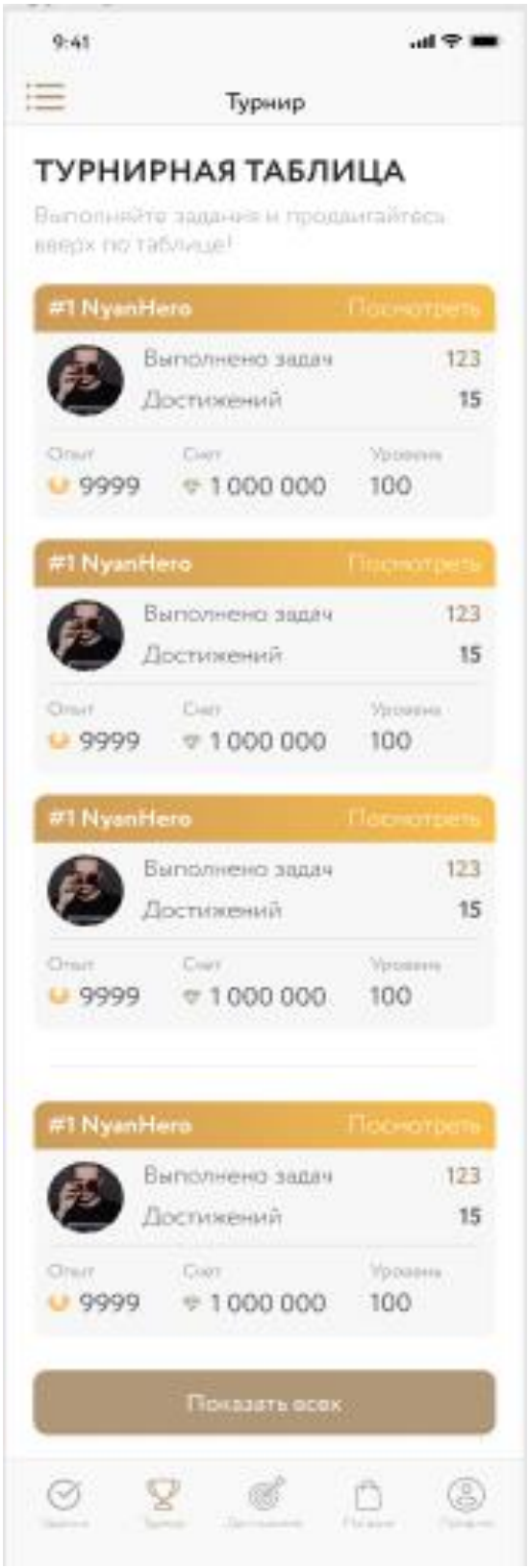


Рисунок 4.6 – Турнір

На екрані «Турнір» можна бачити найкращих гравців, їх досягнення та місце у рейтингу серед усіх гравців. Рейтинг визначають отримані бонуси, рівень та досягнення. Свої досягнення можна побачити у вкладці «Досягнення» на рис. 4.7.

Рисунок 4.7 – «Досягення»

На екрані досягнень виведені досягнення, які можна отримати, та вже отримані користувачем досягнення. При свайпі від лівої грані, або при натисканні кнопки у лівому верхньому куті відкривається бокове меню на рис.4.8.

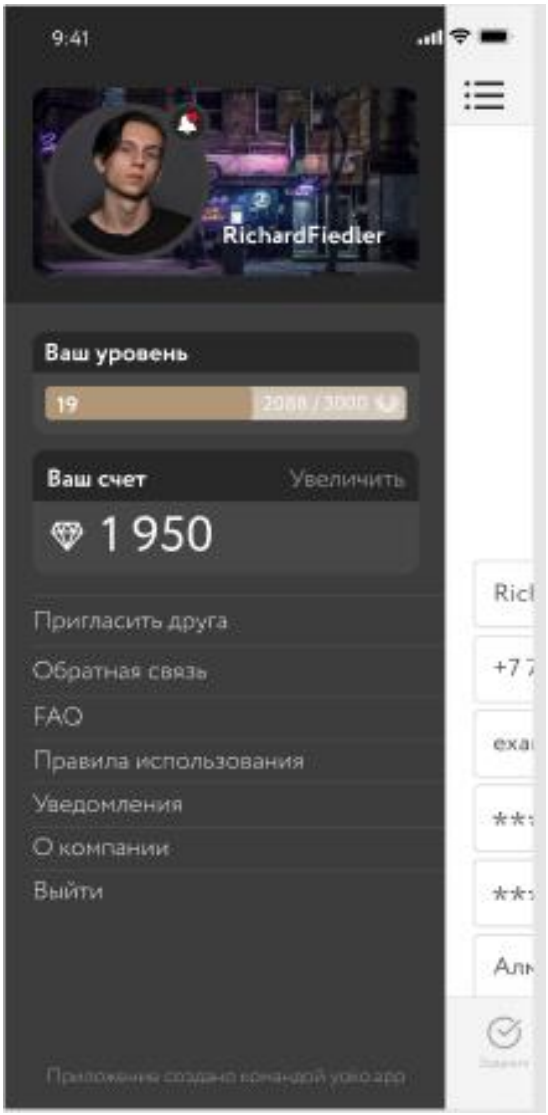


Рисунок 4.8 – Бокове меню

У боковому меню є коротка інформація про аккаунт, фото та досвід. А також тут є меню для взаємодії з додатком, є різна інформація, яка може знадобитися для користувача. Також є опція «Запросити друга», яка також дасть вам можливість отримати бонуси, екран з запрошенням на рис. 4.9.

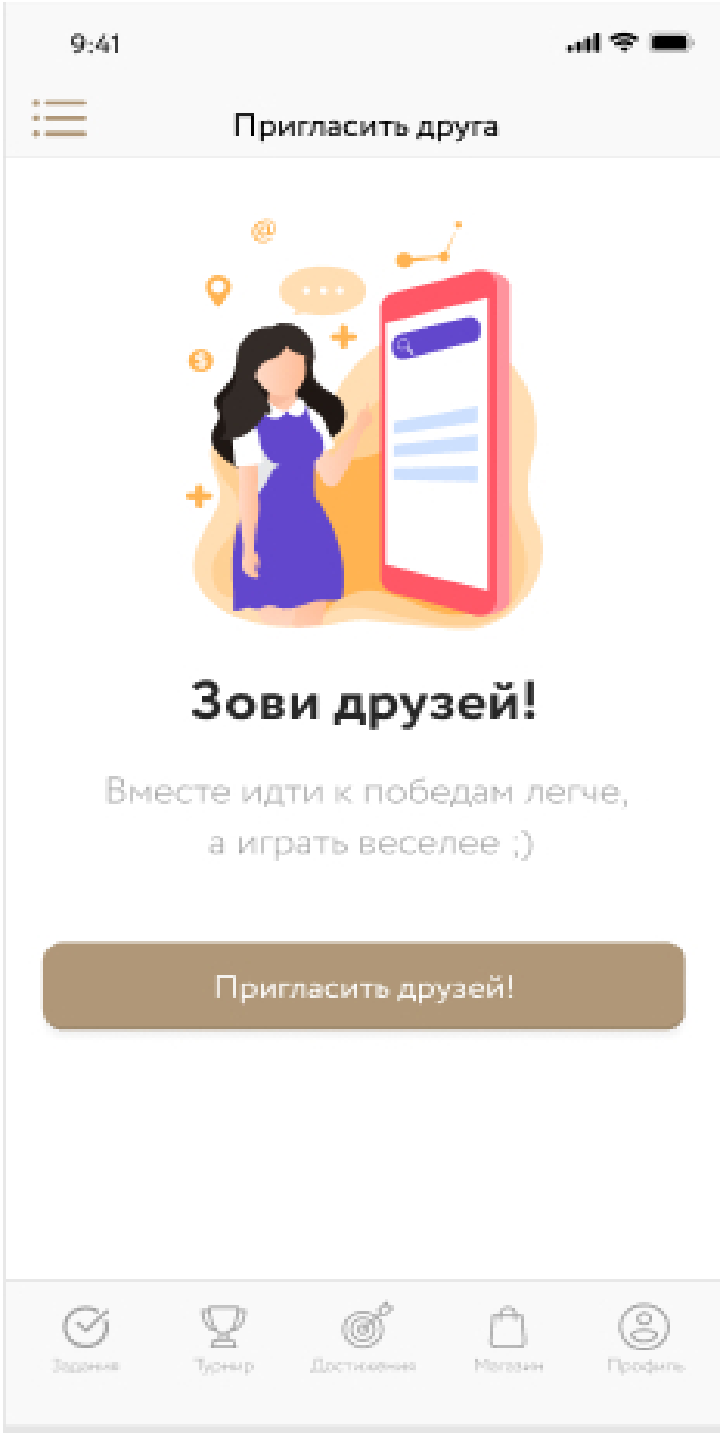


Рисунок 4.9 – екран «Запросити друга»

Також у боковому меню користувач може звернутись до адміністрації. Якщо йому щось не зрозуміло, або виникли проблеми, можна створити заявку, і адміністрація вам відповість. Зворотній зв’язок показано на рисунку 4.10

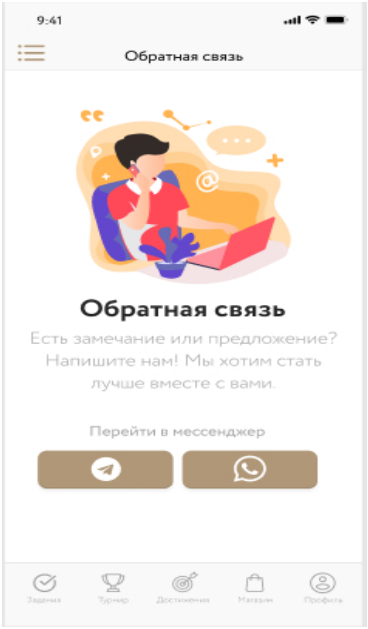


Рисунок 4.10 – «Зворотній зв’язок»

Щоб заробити валюту додатку потрібно виконувати задачі, які сортовано по рівням. На кожному рівні є свої завдання у карточках. На карточці видно обкладинку завдання, короткий опис на кнопку «Приступити» на рисунку 4.11

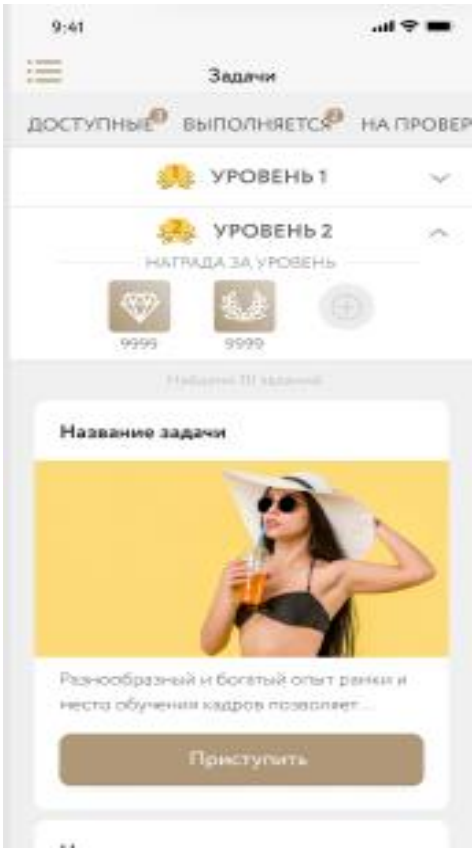


Рисунок 4.11 – Завдання по рівнях

При натисненні кнопки «Приступити» відкривається довгий опис завдання та ще одна така ж кнопка.

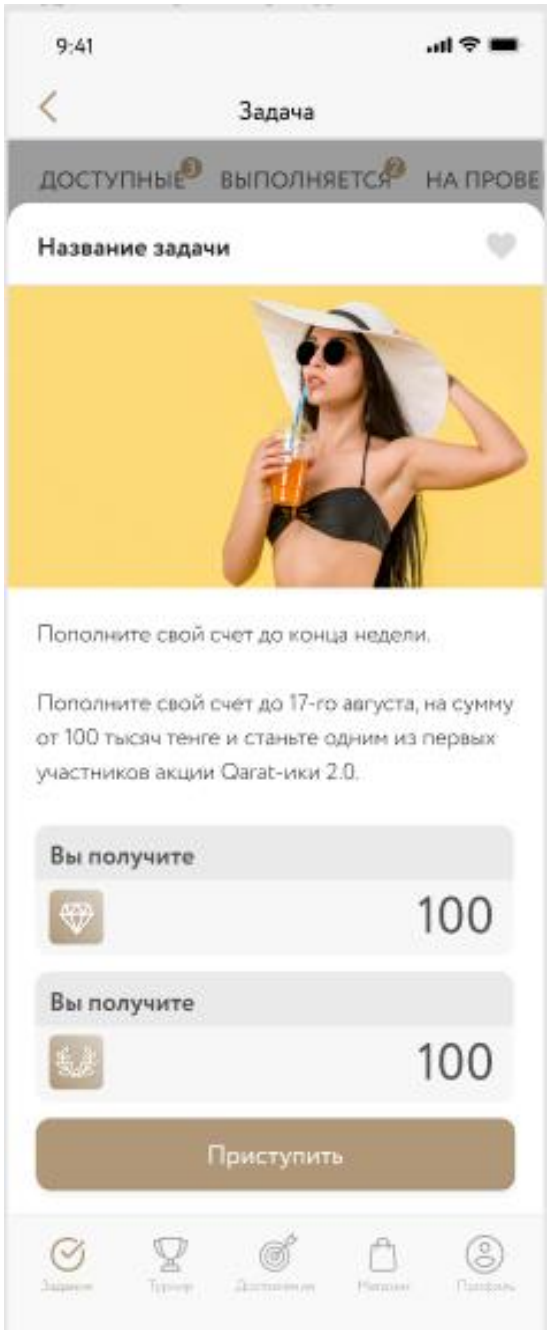
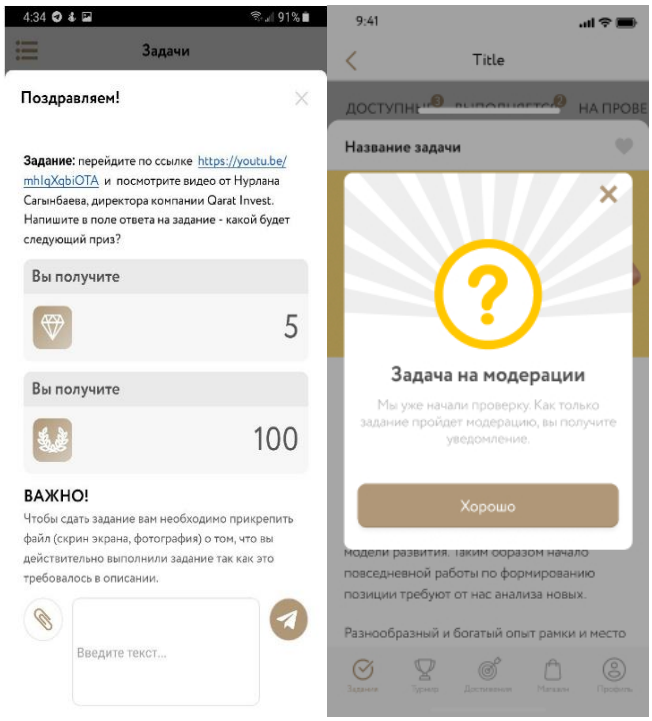


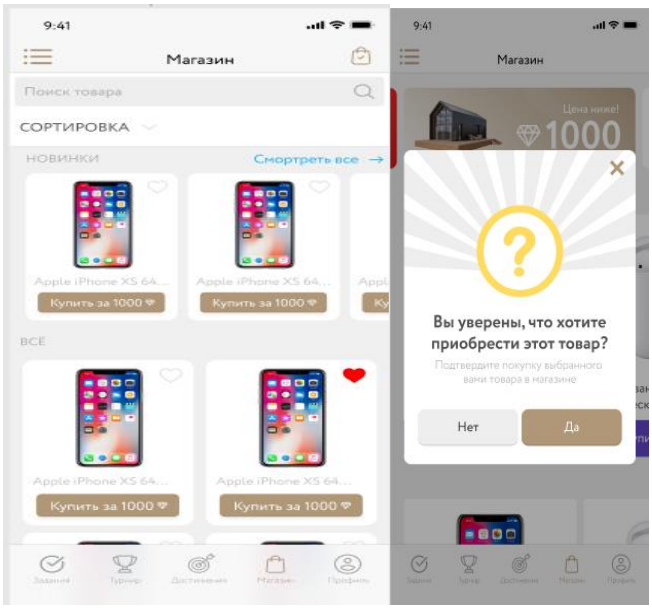
Рисунок 4.12 – Повна картка завдання

При натисненні кнопки «Приступити» ще раз, завдання переходить із вкладки доступні у вкладку «Виконуються», після цього користувач повинен виконати завдання та відкрити його у вкладці «Виконуються» (рис. 4.13).



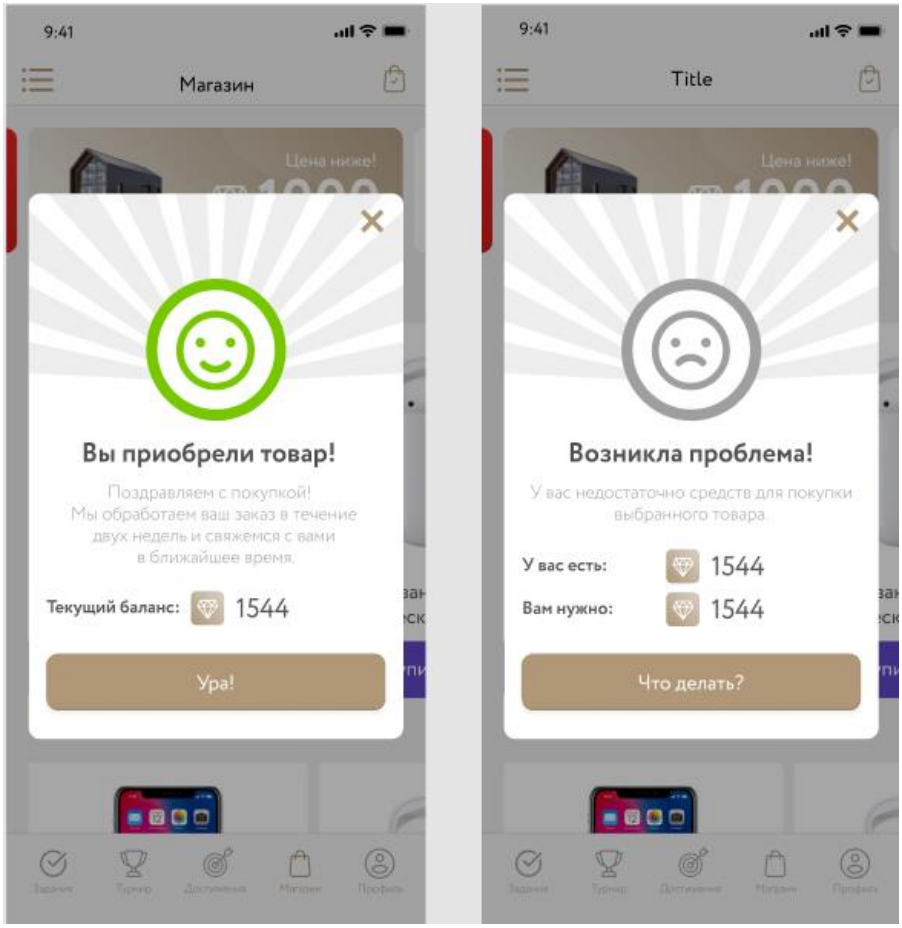
Рисунки 4.13 – Відсилання результату завдання

Після виконання завдання користувач має прикріпити докази (Скріншот, фотографію, тощо). Та може прикріпити своє повідомлення. Після натиснення кнопки «Відправити» завдання переходить в вкладку «На перевірці» та доставляється модераторам, які перевіряють виконання та нараховують винагороду. Після виконання декількох завдань у користувача буде достатньо валюти для покупки товару у магазині всередині додатку (рис. 4.14).



Рисунки 4.14 – Запуск процесу «Перевірка»

В залежності від кількості одиниць валюти та ціни товару, після покупки з’явиться вікно, у якому буде показано баланс користувача або скільки не вистачає для успішної придбання обраного товару (рисунки 4.15).



Рисунки 4.15 – Придбання товару

4.3 Висновки

У четвертому розділі магістерської кваліфікаційної роботи виконано аналіз методів тестування, за результатами якого було обрано метод тестування «чорної скриньки». За обраною методикою проведено тестування розробленого додатку, яке підтвердило його повну працездатність.

5 ЕКОНОМІЧНА ЧАСТИНА

5.1 Оцінювання комерційного потенціалу розробки

Метою проведення технологічного аудиту є оцінювання комерційного потенціалу розробки. Для проведення технологічного аудиту було залучено 2-х незалежних експертів. Такими експертами будуть Войтко В.В. та Ракитянська Г.Б.

Здійснюємо оцінювання комерційного потенціалу розробки за 12-ма критеріями за 5-ти бальною шкалою.

Результати оцінювання комерційного потенціалу розробки наведено в таблиці 5.1.

Таблиця 5.1 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали, посада експерта	
	1. Експерт 1	2. Експерт 2
	Бали, виставлені експертами:	
1	4	4
2	4	3
3	3	4
4	4	3
5	3	4
6	4	4
7	3	3
8	4	4
9	4	4
10	4	3
11	3	4
12	3	4
Сума балів	$СБ_1 = 44$	$СБ_2 = 44$
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_{1}^{3} СБ_i}{2} = 44$	

Отже, з отриманих даних таблиці 5.1 видно, що нова розробка має високий рівень комерційного потенціалу.

5.2 Прогнозування витрат на виконання науково-дослідної роботи та конструкторсько–технологічної роботи.

Для розробки нового програмного продукту необхідні такі витрати.

Основна заробітна плата для розробників визначається за формулою (5.1):

$$З_о = \frac{М}{Т_р} \cdot t, \quad (5.1)$$

де М- місячний посадовий оклад конкретного розробника;

$Т_р$ - кількість робочих днів у місяці, $Т_р$ = 22 дні;

t - число днів роботи розробника, t = 45 днів.

Розрахунки заробітних плат для керівника і програміста наведені в табл.5.2.

Таблиця 5.2 – Розрахунки основної заробітної плати

Працівник	Оклад М, грн.	Оплата за робочий день, грн.	Число днів роботи, t	Витрати на оплату праці, грн.
Науковий керівник	5500	250	6	1500
Інженер-програміст	4000	181,81	45	8181,45
Всього:				9681,45

Розрахуємо додаткову заробітну плату:

$$З_{дод} = 0,1 \cdot 9681,45 = 968,14 \text{ (грн.)}$$

Нарахування на заробітну плату операторів НЗП розраховується як 37,5…40% від суми їхньої основної та додаткової заробітної плати:

$$Н_{зп} = (З_о + З_р) \cdot \frac{\beta}{100}, \quad (5.2)$$

$$Н_{зп} = (9681,45+968,14) \cdot \frac{36,3}{100} = 3865,80 \text{ (грн.)}.$$

Розрахунок амортизаційних витрат для програмного забезпечення виконується за такою формулою (5.3):

$$А=\frac{Ц\cdot Н_а}{100}\cdot\frac{Т}{12}, \quad (5.3)$$

де Ц – балансова вартість обладнання, грн;

$Н_а$ – річна норма амортизаційних відрахувань % (для програмного забезпечення 25%);

Т – Термін використання (Т=3 міс.).

Результати розрахунку амортизаційних відрахувань наведено в табл. 5.3.

Таблиця 5.3 – Розрахунок амортизаційних відрахувань

Найменування програмного забезпечення	Балансова вартість, грн.	Норма амортизації, %	Термін використання, міс.	Величина амортизаційних відрахувань,грн
Персональний комп’ютер	10000	25	3	625
Всього:				625

Розрахуємо витрати на комплектуючі. Витрати на комплектуючі розрахуємо за формулою (5.4):

$$K = \sum_1^n Н_і \cdot Ц_і \cdot К_і, \quad (5.4)$$

де n – кількість комплектуючих;

Ні - кількість комплектуючих і-го виду;

Ці – покупна ціна комплектуючих і-го виду, грн;

Кі – коефіцієнт транспортних витрат (приймемо Кі = 1,1).

Витрати на комплектуючі, що були використані для розробки програмного забезпечення, зведені в табл. 5.4.

Таблиця 5.4 - Витрати на комплектуючі, що були використані для розробки ПЗ.

Найменування матеріалу	Одиниці виміру	Ціна, грн.	Витрачено	Вартість витрачених матеріалів, грн.
Флешка	шт.	180	1	180
Пачка паперу	уп.	115	1	115
Ручка	шт.	5	1	5
Всього з урахуванням транспортних витрат				330

Витрати на силову електроенергію розраховуються за формулою (5.5):

$$В_е=В\cdot П\cdot Ф\cdot К_п\ ; \tag{5.5}$$

де В – вартість 1кВт-години електроенергії (В=1,7 грн/кВт);

П – установлена потужність комп'ютера (П=0,6кВт);

Ф – фактична кількість годин роботи комп'ютера (Ф=200 год.);

$К_п$ – коефіцієнт використання потужності ($К_п< 1$, $К_п = 0,7$).

$$В_е= 1,7 \cdot 0,6 \cdot 200 \cdot 0,7 = 142,8 \text{ (грн.)}$$

Розрахуємо інші витрати $В_{ін}$.

Інші витрати $І_в$ можна прийняти як (100…300)% від суми основної заробітної плати розробників та робітників, які були виконували дану роботу, тобто вираз (5.6):

$$В_{ін} = (1..3) \cdot (З_о + З_р). \tag{5.6}$$

Отже, розрахуємо інші витрати:

$$В_{ін} = 1 * (9681,45 + 968,14) = 10649,59 \text{ (грн).}$$

Сума всіх попередніх статей витрат дає витрати на виконання даної частини роботи:

$$В = З_о + З_д + Н_{зп} + А + К + В_е + І_в$$

В = 9681,45+968,14+3865,80+625+330+142,8+10649,59= 26262,78 (грн.)

Розрахуємо загальну вартість наукової роботи $B_{заг}$ за формулою (5.7):

$$В_{заг} = \frac{В_{ін}}{\alpha} \quad (5.7)$$

де α – частка витрат, які безпосередньо здійснює виконавець даного етапу роботи, у відн. одиницях = 1.

$$В_{заг} = \frac{26262,78}{1} = 26262,78$$

Прогнозування загальних витрат ЗВ на виконання та впровадження результатів виконаної наукової роботи здійснюється за формулою (5.8):

$$ЗВ = \frac{В_{заг}}{\beta} \quad (5.8)$$

де β – коефіцієнт, який характеризує етап (стадію) виконання даної роботи.

Отже, розрахуємо загальні витрати:

$$ЗВ = \frac{26262,78}{0,9} = 29180,86 \text{ (грн.)}$$

5.3 Прогнозування комерційних ефектів від реалізації результатів розробки

Спрогнозуємо отримання прибутку від реалізації результатів нашої розробки. Зростання чистого прибутку можна оцінити у теперішній вартості

грошей. Це забезпечить підприємству (організації) надходження додаткових коштів, які дозволять покращити фінансові результати діяльності .

Оцінка зростання чистого прибутку підприємства від впровадження результатів наукової розробки. У цьому випадку збільшення чистого прибутку підприємства $\Delta П_i$ для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки, розраховується за формулою (5.9):

$$\Delta П_i = \sum_1^n (\Delta П_я \cdot N + П_я \Delta N)_i \qquad (5.9)$$

де $\Delta П_я$ – покращення основного якісного показника від впровадження результатів розробки у даному році;

N – основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки;

$П_я$ – основний якісний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки.

У результаті впровадження результатів наукової розробки витрати на виготовлення інформаційної технології зменшаться на 20 грн (що автоматично спричинить збільшення чистого прибутку підприємства на 20 грн), а кількість користувачів, які будуть користуватись збільшиться: протягом першого року – на 200 користувачів, протягом другого року – на 175 користувачів, протягом третього року – 125 користувачів. Реалізація інформаційної технології до впровадження результатів наукової розробки складала 600 користувачів, а прибуток, що отримував розробник до впровадження результатів наукової розробки – 300 грн.

Спрогнозуємо збільшення чистого прибутку від впровадження результатів наукової розробки у кожному році відносно базового.

Отже, збільшення чистого продукту $\Delta П_1$ протягом першого року складатиме:

$$\Delta П_1 = 20 \cdot 600 + (300 + 20) \cdot 200 = 76000 \text{ грн.}$$

Протягом другого року:

$$\Delta П_2 = 20 \cdot 600 + (300 + 20) \cdot (200 + 175) = 132000 \text{ грн.}$$

Протягом третього року:

$$\Delta П_3 = 20 \cdot 600 + (300 + 20) \cdot (200 + 175 + 125) = 172000 \text{ грн.}$$

5.4 Розрахунок ефективності вкладених інвестицій та період їх окупності

Визначимо абсолютну і відносну ефективність вкладених інвестором інвестицій та розрахуємо термін окупності.

Абсолютна ефективність $Е_{абс}$вкладених інвестицій розраховується за формулою (5.10):

$$Е_{абс} = (ПП - PV), \qquad (5.10)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДДКР, грн;

т – період часу, протягом якого виявляються результати впровадженої НДДКР, 3 роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,1;

t – період часу (в роках) від моменту отримання чистого прибутку до точки 2, 3,4.

Рисунок, що характеризує рух платежів (інвестицій та додаткових прибутків) буде мати вигляд рис. 5.1.

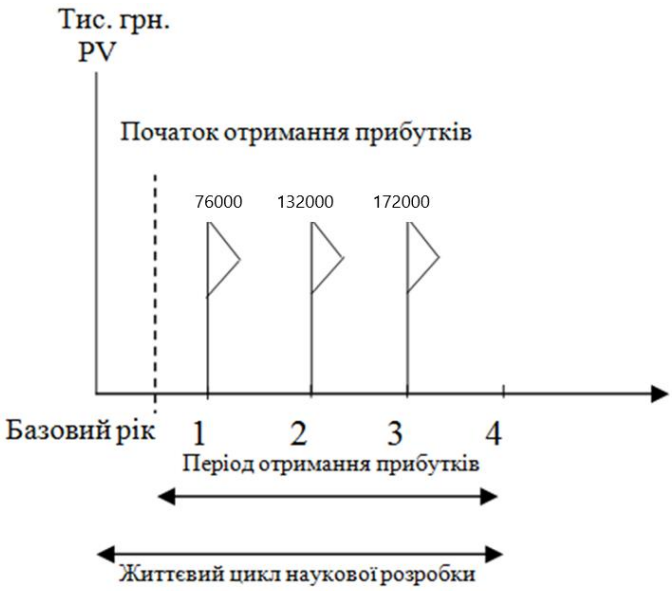


Рисунок 5.1 – Вісь часу з фіксацією платежів, що мають місце під час розробки та впровадження результатів НДДКР

Розрахуємо вартість чистих прибутків за формулою (5.11):

$$ПП = \sum_1^m \frac{\Delta П_i}{(1+\tau)^t} \quad (5.11)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДДКР, грн;

т – період часу, протягом якого виявляються результати впровадженої НДДКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,1;

t – період часу (в роках) від моменту отримання чистого прибутку до точки.

Отже, розрахуємо вартість чистого прибутку:

$$ПП = \frac{29180,86}{(1+0,1)^0} + \frac{76000}{(1+0,1)^2} + \frac{132000}{(1+0,1)^3} + \frac{172000}{(1+0,1)^4} = 308642,63 \text{ (грн.)}$$

Тоді розрахуємо $E_{абс}$:

$$E_{абс} = 308642{,}63 - 29180{,}86 = 279461{,}77 \text{ грн.}$$

Оскільки $E_{абс} > 0$, то вкладання коштів на виконання та впровадження результатів НДДКР буде доцільним.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$ за формулою (5.12):

$$E_в = \sqrt[T]{1 + \frac{E_{абс}}{PV}} - 1 \qquad (5.12)$$

де $E_{абс}$ – абсолютна ефективність вкладених інвестицій, грн;

PV – теперішня вартість інвестицій PV = ЗВ, грн;

$T_ж$ – життєвий цикл наукової розробки, роки.

Тоді будемо мати:

$$E_в = \sqrt[3]{1 + \frac{279466{,}71}{29180{,}86}} - 1 = 1{,}19 \text{ або } 119\ \%$$

Далі, розраховану величина $E_в$ порівнюємо з мінімальною (бар'єрною) ставкою дисконтування $\tau_{мін}$, яка визначає ту мінімальну дохідність, нижче за яку інвестиції вкладатися не будуть. У загальному вигляді мінімальна (бар'єрна) ставка дисконтування $\tau_{мін}$ визначається за формулою:

$$\tau = d + f,$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2019 році в Україні d = 0,2;

f – показник, що характеризує ризикованість вкладень, величина f = 0,1.

$$\tau = 0{,}2 + 0{,}1 = 0{,}3$$

Оскільки $E_в$ = 119% > $\tau_{мін}$ = 0,3 = 30%, то у інвестор буде зацікавлений вкладати гроші в дану наукову розробку.

Термін окупності вкладених у реалізацію наукового проекту інвестицій. Термін окупності вкладених у реалізацію наукового проекту інвестицій $T_{ок}$ розраховується за формулою:

$$T_{ок} = \frac{1}{E_в}$$

$$T_{ок} = \frac{1}{1{,}19} = 0{,}84 \text{ року}$$

Обрахувавши термін окупності даної наукової розробки, можна зробити висновок, що фінансування даної наукової розробки буде доцільним.

5.5 Висновки

У п’ятому розділі було проведено оцінювання комерційного потенціалу розробки, виконано технологічний аудит для оцінювання комерційного потенціалу розробки. Залучено 2-х незалежних експертів. Спрогнозовано витрати на виконання науково-дослідної роботи та конструкторсько-технологічної роботи.

На вісі часу з фіксацією платежів, що мають місце під час розробки та впровадження результатів НДДКР показано життєвий цикл програмної розробки, початок отримання прибутків та його період.

Вартість чистого прибутку складає 308642,63 (грн.). Абсолютна ефективність вкладених інвестицій: 279461,77 грн. Відносна ефективність вкладених інвестицій: 1, 19 або 119 %. Термін окупності: 0,84 року. Обрахувавши вартість чистого прибутку, абсолютну і відносну ефективність вкладених інвестицій та термін окупності даної наукової розробки, можна зробити висновок, що фінансування даної наукової розробки буде доцільним.

ВИСНОВКИ

У магістерській кваліфікаційній роботі розроблено додаток, що використовує метод гейміфікації для ефективного залучення нових користувачів.

Розглянуто особливості розробки мобільних додатків. Описано використання концепції гейміфікації.

Удосконалено метод гейміфікації подій, у якому використовуються адаптивні алгоритми підбору рівня завдань, що дозволяє підвищити активність користувачів шляхом їх зацікавлення процесом проходження ігрових завдань.

Запропоновано моделі системи гейміфікації подій, які орієнтовані на адаптивний підхід підбору завдань з урахуванням рівня і активності користувача, що дозволяє розширити комунікаційну взаємодію користувачів у процесі розвитку маркетингової стратегії компанії.

Були розроблені алгоритми роботи системи гейміфікації подій, програмні модулі системи з використанням технологій розробки мобільних додатків під операційні платформи Android/iOS. Розроблено програму клієнт-хмарної архітектури, для чого спроектовано та розроблено хмарне сховище даних й проаналізовані особливості бази даних та розроблена структура бази даних для мобільного додатку. Також розроблена база даних для показу рейтингу учасників системи.

Розроблено засоби адміністрування системи.

Проведено тестування розробленого мобільного застосунку, результати якого підтвердили коректність роботи програми.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Beginning Android Games / Robert Green. – NY: Apress, 2012. – 714p.

2. Grady B. Object-Oriented Analysis and Design with Applications/ B. Grady. – Amsterdam: Addison-Wesley Professional, 2009. – 720 p.

3. Horton's I. Beginning JavaSctipt/I. Horton's. – Amsterdam: Wrox, 2015. – 988 p.

4. Бурбело С.М. Розробка автоматизованої мобільної системи гейміфікації подій / С.М. Бурбело, С.В. Бевз, В.В. Войтко, А.В. Денисюк, В.О.Фролов// Молодь в науці: дослідження, проблеми, перспективи - 2019. Матеріали Всеукраїнської науково-практичної Інтернет-конференції [Електронний ресурс]. – Режим доступу: https://conferences.vntu.edu.ua/index.php/mn/mn2020/paper/viewFile/8470/7090

5. Stellman A. Head First NodeJS / A. Stellman. – Boston: O'Reilly Media, 2014. – 784 p.

6. Мови програмування — Stackoverflow. [Електронний ресурс]. – Режим доступу: https://ru.stackoverflow.com/tags/info

7. Нечітка логіка, нечіткі множини, м'які обчислення — часті питання Режим доступу http://www.znannya.org/?view=fuzzy-logic-q

8. Нечітка логіка: лекція – Режим доступу: https://www.victoria.lviv.ua/html/oio/html/theme11.htm

9. Бази знань. Інтелектуальні інформаційні системи [Електронний ресурс]. – Режим доступу: https://studfiles.net/preview/5474324/

10. Nodejs Documentation [Електронний ресурс]. Режим доступу до ресурсу: https://nodejs.org/en.

11. Fleder D., Hosanagar K. Blockbuster Culture's Next Rise or Fall: The Impact of Recommender Systems on Sales Diversity (журнал) // Management Science, Vol. 55, No. 5, May 2009. – P. 697-712.

12. Xiaoyuan Su and Taghi M. Khoshgoftaar. A Survey of Collaborative Filtering Techniques A Survey of Collaborative Filtering Techniques (журнал) //

Hindawi Publishing Corporation, Advances in Artificial Intelligence archive, USA. — 2009. — P. 1 - 19.

13. Zadeh, L. A. (1965). Fuzzy sets. *Information and Control* **8** (3): 338. doi:10.1016/S0019-9958(65)90241-X

14. Білоусова Л.І. Інформатика. Навчальний посібник // Білоусова Л.І., Муравка А.С., Олефіренко Н.В. - Харків: Факт, 2009. – 352 с.

15. User experience – [Електронний ресурс]: Режим доступу до матеріалу: https://en.wikipedia.org/wiki/User_experience.

16. Adobe Illustrator – [Електронний ресурс]: Режим доступу до матеріалу: https://ru.wikipedia.org/wiki/Adobe_Illustrator.

17. Importing Assets – [Електронний ресурс]: Режим доступу до матеріалу: https://docs.unity3d.com/560/Documentation/Manual/ImportingAssets.html

18. Creighton R. H. Unity 3D Game Development by Example Beginner's Guide / R.H. Creighton – Birmingham: Packt Publishing, 2010. – 364 p. – ISBN 978-1-849690-54-6.

19. Norton T. Learning C# by Developing Games with Unity 3D Beginner's Guide / T. Norton – Birmingham: Packt Publishing, 2013. – 271 p. – ISBN 978-1-84969-658-6.

20. Тестування програмного забезпечення – [Електронний ресурс]: Режим доступу до матеріалу: http://uk.wikipedia.org/wiki/ Тестування_програмного_забезпечення.

Додаток А Технічне завдання

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії

ЗАТВЕРДЖУЮ
д.т.н., проф. О. Н. Романюк
"____ "_____2019 р.

Технічне завдання
на магістерську кваліфікаційну роботу «Розробка методу та програмних засобів мобільної системи гейміфікації подій» за спеціальністю
121 – Інженерія програмного забезпечення

Керівник магістерської кваліфікаційної роботи:
_____ д.т.н., проф. Г.Б.Ракитянська
"____ "_____2019 р.
Виконав:
_____студент гр.1ПІ-18м В.О. Фролов
"____ "_____2019 р.

Вінниця – 2019 року

1. Найменування та галузь застосування

Магістерська кваліфікаційна робота: «Розробка методу та програмних засобів мобільної системи гейміфікації подій».

Галузь застосування – мобільні застосунки.

3. Підстава для розробки

Підставою для виконання магістерської кваліфікаційної роботи (МКР) є індивідуальне завдання на МКР та наказ № ректора по ВНТУ про закріплення тем МКР.

4. Мета та призначення розробки

Метою роботи є розширення маркетингового комунікаційного функціоналу системи через введення гейміфікації подій з використанням мобільної системи, що дозволить зацікавити нових та активізувати наявних користувачів системи у процесі вирішення робочих завдань.

Призначення роботи – розробка методів і засобів гейміфікації з використанням технологій розробки мобільних додатків.

3 Вихідні дані для проведення НДР

Перелік основних літературних джерел, на основі яких буде виконуватись МКР.

1. Nodejs Documentation [Електронний ресурс]. Режим доступу до ресурсу: https://nodejs.org/en.
2. Fleder D., Hosanagar K. Blockbuster Culture's Next Rise or Fall: The Impact of Recommender Systems on Sales Diversity (журнал) // Management Science, Vol. 55, No. 5, May 2009. – P. 697-712.
3. Xiaoyuan Su and Taghi M. Khoshgoftaar. A Survey of Collaborative Filtering Techniques A Survey of Collaborative Filtering Techniques (журнал) // Hindawi Publishing Corporation, Advances in Artificial Intelligence archive, USA. — 2009. — P. 1 - 19.

4. Тестування програмного забезпечення – [Електронний ресурс]: Режим доступу до матеріалу: http://uk.wikipedia.org/wiki/ Тестування_програмного_забезпечення.

4. Технічні вимоги

Розробити мобільний додаток на платформі iOS та Android, що дозволить користувачу у зручній та інтуітивній формі взаємодіяти з мобільною системою бренду у вигляді гри.

5. Конструктивні вимоги

Зовнішній вигляд додатку повинен бути приємним та естетичним, анімації плавними, робота додатку має корректно починатись навіть після блокування телефону.

Графічна та текстова документація повинна відповідати діючим стандартам України.

6. Перелік технічної документації, що пред'являється по закінченню робіт:

– пояснювальна записка до МКР;

– технічне завдання;

– лістинги програми.

7. Вимоги до рівня уніфікації та стандартизації

При розробці програмних засобів слід дотримуватися уніфікації і ДСТУ.

8. Стадії та етапи розробки:

№ з/п	Назва етапів магістерської кваліфікаційної Роботи	Строк виконання етапів роботи
1	Обґрунтування вибору методу розробки та постановка задачі дослідження	
2	Розробка структури та алгоритмів додатку	
3	Вибір середовища розробки, мови програмування та архітектури додатку	
4	Розробка клієнтської частини	
5	Розробка серверної частини	
6	Тестування роботи додатку	
7	Оформлення матеріалів до захисту МКР	

9. Порядок контролю та прийняття

Виконання етапів магістерської кваліфікаційної роботи контролюється керівником згідно з графіком виконання роботи.

Прийняття магістерської кваліфікаційної роботи здійснюється ДЕК, затвердженою зав. кафедрою згідно з графіком

Додаток Б. Програмний код додатку

Progress.js

```
@codePush(codePushOptions)
export default class Welcome extends Component {
 unsubscribe; async componentDidMount() {
 const amplitude = new RNAmplitude('');
 amplitude.logEvent('session-start', {
    user_id: auth().currentUser && auth().currentUser.uid,
    timestamps: +Date.now(),
    phone: auth().currentUser && auth().currentUser.phoneNumber
});

const onboarding = await AsyncStorage.getItem('@onboarding');
 if (!onboarding) {
     await AsyncStorage.setItem('@onboarding', 'true');
    return splashApp();
}
this.unsubscribe = auth().onAuthStateChanged(async user => {
 if (!user) {
    return nonloginedApp();
 }
const userRef = firestore().collection('qi_users').doc(user.uid);
const userData = await userRef.get();
 const profile = userData.data();
 }
render() {
     return ( <Loader/> );
}
}
```

Tasks.js

```
import React, {Component} from 'react';
import {
  StyleSheet,
  FlatList,
  View,
  Platform,
  Dimensions,
  Text,
  Image,
  ScrollView,
  TouchableOpacity
} from 'react-native';
import Icon from 'react-native-vector-icons/AntDesign';
import Entypo from 'react-native-vector-icons/Entypo';
import {TabView, SceneMap, TabBar} from 'react-native-tab-view';
import {Navigation} from 'react-native-navigation';
import DeviceInfo from 'react-native-device-info';
import RNAmplitude from 'react-native-amplitude-analytics';

import {Card, LabelLevel} from './components';
import {firestore, auth,
  // messaging
} from './services/firebase';
import {icons} from './svg';

function processToken(userData, token) {
  const uniqueId = DeviceInfo.getUniqueId();
  const name = DeviceInfo.getDeviceName();
```

```
  userData.devices = userData.devices || {};

  if (!userData.devices[uniqueId]) {
    userData.devices[uniqueId] = {
      name,
      token,
      createdAt: firestore.TIMESTAMP
    };
  }

  return userData.devices;
}

export default class Tasks extends Component {
  mounted;
  tasksSub;
  likesSub;
  userTasksSub;
  state = {
    users: [],
    sections: [],
    index: 0,
    routes: [
      {key: 'zero', title: 'ДОСТУПНЫЕ'},
      {key: 'first', title: 'ВЫПОЛНЯЮТСЯ'},
      {key: 'second', title: 'НА ПРОВЕРКЕ'}
    ]
  };
```

```
componentWillUnmount() {
  this.tasksSub && this.tasksSub();
  this.likesSub && this.likesSub();
  this.userTasksSub && this.userTasksSub();
}

updateBadge(key, value = 0) {
  let {routes = []} = this.state;
  routes = routes.map(r => {
    if (r.key === key) {
      r.badge = value
    }

    return r;
  });

  this.setState({routes});
}

 toDetail(task) {
    return Navigation.showModal({
       component: {
          name: 'modalTask',
          passProps: {
             task,
          changeTab: (index) => this.setState({index}),
          },
          options: {
             layout: {
```

```
                backgroundColor: 'rgba(0,0,0,0.6)'
            },
            modalPresentationStyle: 'overCurrentContext',
            topBar: {
                visible: false,
                drawBehind: true
            },
            animations: {
                showModal: {
                    enabled: false
                },
                dismissModal: {
                    enabled: false
                }
            }
        }
    }
  });
}

render() {
  const {moderate = [], progress = [], likes = {}} = this.state;
  const Moder = icons['moder'];
  const Failed = icons['failed'];
  const Done = icons['done'];

  return (
    <View style={styles.wrapStyle}>
      <TabView
        navigationState={this.state}
```

```
            renderTabBar={props => <TabBar
              {...props}
              style={{backgroundColor: 'transparent'}}
              inactiveColor={'black'}
              activeColor={'black'}
              renderBadge={({route}) => (
                route.badge ? <Text style={[{
                  backgroundColor: '#AF9778',
                  borderRadius: 9,
                  color: 'white',
                  padding: 4,
                  fontSize: 8,
                  top: 3,
                  textAlign: 'center',
                  right: 11
                }, route.badge > 100 ? {width: 24, right: 5} : {width:
18}]}>{route.badge}</Text> : null
              )}
              renderLabel={({route, focused}) => (
                <Text
                  style={[{fontSize: 10, fontFamily: 'Circe-Regular'}, focused ?
{color: 'black'} : {}]}>
                  {route.title}
                </Text>
              )}
              indicatorStyle={{
                backgroundColor: '#AF9778',
                height: 3
              }}
            />}
```

```
            renderScene={SceneMap({
              zero: () =>
                <ScrollView>
                  {this.state.sections.map(section =>
                    <LabelLevel user={this.state.level} toDetail={e =>
this.toDetail(e)} xp={section.xp} qarat={section.qarat} tasks={section.tasks}
level={section.level} locked={section.locked}/>
                  )}
                </ScrollView>,
              first: () => <FlatList
                data={progress}
                contentContainerStyle={styles.tabview}
                keyExtractor={item => item.id}
                renderItem={({item}) => <Card
                  key={item.id}
                        sub={item.moderationMessage ? <Failed/> : null}
                        action={() => this.toDetail(item)}
                  sub2={<Icon name='heart' size={25} color={likes[item.id] ? 'red' :
'lightgray'}/>}
                  actionText={'Сдать'}
                  {...item}
                />}
              />,
              second: () => <FlatList
                data={moderate}
                contentContainerStyle={styles.tabview}
                keyExtractor={item => item.id}
                renderItem={({item}) => <Card
                  key={item.id}
                  sub={<Moder />}
```

```
                    action={() => this.toDetail(item)}
                    actionText={'Подробней'}
                    {...item}
                  />}
                />
              })}
              onIndexChange={index => this.setState({index})}
              initialLayout={{width: Dimensions.get('window').width}}
            />
            { this.state.level == 4 ?
            <TouchableOpacity style={styles.bonusButton} onPress={() =>
this.toBonus()}>
              <Text style={styles.textBonusButton}>Бонусные задания</Text>
              <Entypo name='plus' size={30} color={'#AF9778'} />
            </TouchableOpacity> : null }
          </View>
      );
    }
}

const styles = StyleSheet.create({
  wrapStyle: {: {
    color: '#303030',
    fontFamily: 'Circe-Bold',
    fontSize: 20
  }
});
```

ДОДАТОК В
Ілюстративний матеріал до захисту
магістерської кваліфікаційної роботи

Ілюстративний матеріал до захисту магістерської кваліфікаційної роботи

Завідувач кафедри ПЗ, д. т. н., професор _____Романюк О.Н.

Науковий керівник, к. т. н., доцент кафедри ПЗ_____ Ракитянська Г.Б.

Рецензент, к.т.н., доцент кафедри КН_____ Арсенюк І.Р.

Нормоконтроль, к. т. н., доцент _____ Ракитянська Г.Б.

Виконавець, студент групи 1ПІ-18м _____Фролов В.О

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення

Розробка методу та програмних засобів мобільної системи гейміфікації подій

Студент 1ПІ-18м Фролов Віктор
Наук. керівник - к.т.н. Ракитянська Г.Б.

Визначення

Гейміфіка́ція (ігровізація, геймізація, англ. gamification) — використання ігрових практик та механізмів у неігровому контексті для залучення кінцевих користувачів до розв'язання проблем. Гейміфікація була досліджена у декількох царинах, серед яких: взаємодія з клієнтами, виконання фізичних вправ, повернення інвестицій, якість даних, пунктуальність та навчання. Більшість досліджень з ігровізації показали позитивні тенденції після гейміфікації.

Мета, об’єкт та предмет дослідження

- Метою роботи є розширення маркетингового комунікаційного функціоналу системи через введення гейміфікації подій з використанням мобільної системи, що дозволить зацікавити нових та активізувати наявних користувачів системи у процесі вирішення робочих завдань.

Об’єкт дослідження – процес гейміфікації подій з використанням технологій розробки мобільних додатків.

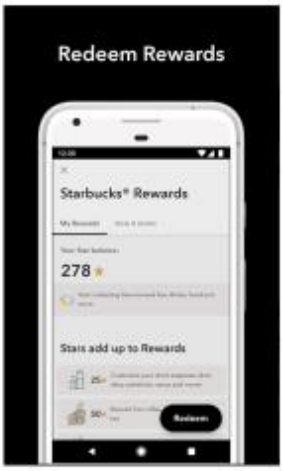
Предмет дослідження – програмні засоби реалізації процесу гейміфікації подій з використанням технологій розробки мобільних застосунків під операційні платформи Android/iOS.

Наукова новизна

1. Подальшого розвитку дістав метод гейміфікації подій, у якому, на відміну від існуючих, використовуються адаптивні алгоритми підбору рівня завдань, що дозволяє підвищити активність користувачів шляхом їх зацікавлення процесом проходження ігрових завдань.

2. Подальшого розвитку дістали моделі системи гейміфікації подій, які, на відміну від існуючих, орієнтовані на адаптивний підхід підбору завдань з урахуванням рівня і активності користувача, що дозволяє розширити комунікаційну взаємодію користувачів у процесі розвитку маркетингової стратегії компанії.

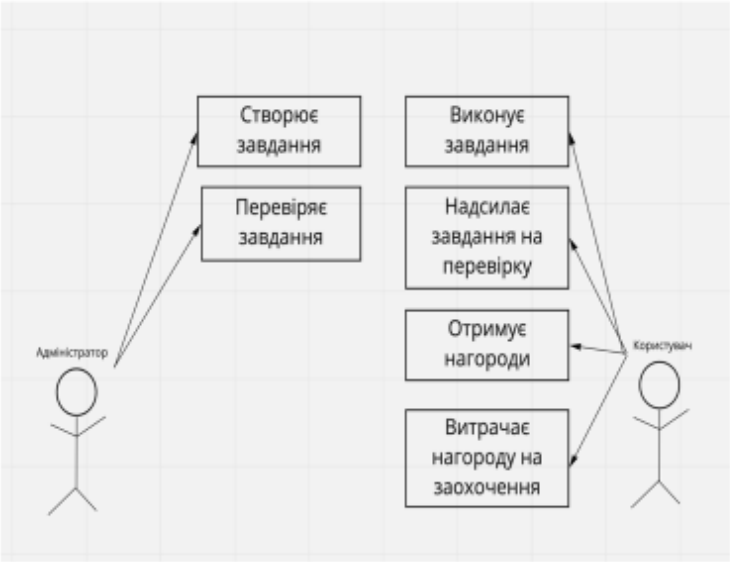
Порівняльний аналіз аналогів



Метод гейміфікації подій:

1. Авторизація користувача в мобільній системі.
2. Створення завдань, ігорового контекту, класифікованих за типом та складністю. Серед типів виділяють завдання: ігрові, інтелектуальні, сюжетні, з елементами гумору. Ці типи завдань також можуть поділятися на класи швидкої реалізації і ті, що потребують певних часових витрат на виконання.
3. Вибір стратегії ведення процесу гейміфікації з використанням адаптивних алгоритмів підбору завдань. При реалізації адаптивних алгоритмів враховуються зацікавленість й активність користувача в процесі проходження завдань, що впливає на автоматичний підбір завдань різного рівня складності та очікуваної тривалості його виконання.
4. Виконання завдань користувачем та викладення результатів у середовищі мобільного додатку системи гейміфікації.
5. Оцінювання результатів виконаної користувачем роботи.
6. Надання бонусів та інших засобів заохочення учасників гейміфікованого процесу маркетингової стратегії розвитку компанії.

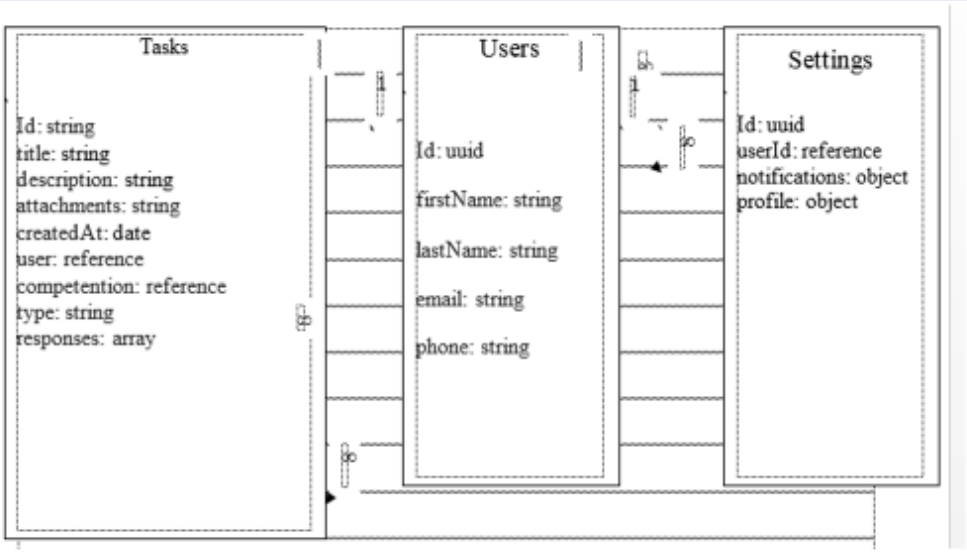
Узагальнена модель процесу гейміфікації подій



Модель системи гейміфікації подій

Адаптований алгоритм гейміфікації

Структура бази даних системи гейміфікації подій



Тестування програми

Тестування програми

Тестування програми

Висновки

У магістерській кваліфікаційній роботі розроблено додаток, що використовує метод гейміфікації для ефективного залучення нових користувачів.

Розглянуто особливості розробки мобільних додатків. Описано використання концепції гейміфікації.

Удосконалено метод гейміфікації подій, у якому використовуються адаптивні алгоритми підбору рівня завдань, що дозволяє підвищити активність користувачів шляхом їх зацікавлення процесом проходження ігрових завдань.

Запропоновано моделі системи гейміфікації подій, які орієнтовані на адаптивний підхід підбору завдань з урахуванням рівня і активності користувача, що дозволяє розширити комунікаційну взаємодію користувачів у процесі розвитку маркетингової стратегії компанії.

Були розроблені алгоритми роботи системи гейміфікації подій, програмні модулі системи з використанням технологій розробки мобільних додатків під операційні платформи Android/iOS. Розроблено програму клієнт-хмарної архітектури, для чого спроектовано та розроблено хмарне сховище даних й проаналізовані особливості бази даних та розроблена структура бази даних для мобільного додатку. Також розроблена база даних для показу рейтингу учасників системи.

Розроблено засоби адміністрування системи.

Проведено тестування розробленого мобільного застосунку, результати якого підтвердили коректність роботи програми.