Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інтелектуальних інформаційних технологій та автоматизації
(повне найменування інституту, назва факультету (відділення))

Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

# МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

**«Інформаційна технологія гейміфікації процесу тестування під час навчання»**

Виконав: студент 2-го курсу, групи 3КН-22м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
Вдовиченко А. І.
(прізвище та ініціали)

Керівник: к. т. н., доцент каф. КН
Арсенюк І. Р.
(прізвище та ініціали)
«07» 12 2023 р.

Опонент:к. т. н, доцент каф. САІТ
Жуков С. О.
(прізвище та ініціали)
«07» 12 2023 р.

**Допущено до захисту**
Завідувач кафедри КН
д. т. н., проф. Яровий А. А.
(прізвище та ініціали)
«08» 12 2023 р.

Вінниця ВНТУ - 2023 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації Кафедра комп'ютерних наук
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань – 12 «Інформаційні технології»
Спеціальність – 122 «Комп'ютерні науки»
Освітньо-професійна програма – «Системи штучного інтелекту»

**ЗАТВЕРДЖУЮ**
**Завідувач кафедри КН**
**Д. т. н., проф. Яровий А. А.**
29. 08. **2023 року**

**З А В Д А Н Н Я**

**НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ**

Вдовиченко Андрій Іванович
(прізвище, ім`я, по батькові)

1. Тема роботи Інформаційна технологія гейміфікації процесу тестування під час навчання

керівник роботи _ к.т.н., доцент кафедри КН Арсенюк І. Р.
затверджені наказом ВНТУ від «18» вересня 2023 року № 247

2. Строк подання студентом роботи «13» листопада 2023 року

3. Вихідні дані до роботи:
Вхідні дані: наявність трьох різних режимів тестування (часовий, чекпойнти і питання-відповідь); можливість завантаження тестів з файлів .doc та .txt; можливість підвантаження бази до 1000 питань 4) об'єктно-орієнтована мова програмування.

4. Зміст текстової частини:
Вступ, Аналіз предметної області гейміфікації, Розробка алгоритму роботи програмного модуля гейміфікації, Реалізація програмного забезпечення програмного модуля гейміфікації, Економічна частина, Висновки.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень)

1. Консультанти розділів роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | виконання прийняв |
| 1-3 | Арсенюк І.Р., к.т.н., доц. каф. КН | 29.08.23 | 01.11.23 |
| 4 | Д.к. ЕПВМ Кавецький В.В. | 8.09.23 | 23.10.23 |
| | | | |

2. Дата видачі завдання 29.08 2023 року

КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів магістерської кваліфікаційної роботи | Строк виконання етапів роботи |
|---|---|---|
| 1 | Аналіз предметної області гейміфікації | 01.09.23 - 05.09.23 |
| 2 | Розробка алгоритму роботи програмного модуля гейміфікації | 06.09.23 - 16.09.23 |
| 3 | Реалізація програмного забезпечення програмного модуля гейміфікації | 17.09.23 - 07.10.23 |
| 4 | Підготовка економічної частини | 08.09.23 - 23.10.23 |
| 5 | Апробація та/або впровадження результатів дослідження | 24.10.23 - 01.11.23 |
| 6 | Оформлення пояснювальної записки, графічного матеріалу та презентації | 02.11.23 - 10.11.23 |
| | | |

Студент _____ (підпис) Вдовиченко А.І.

Керівник роботи _____ (підпис) Арсенюк І.Р.

# АНОТАЦІЯ

УДК 004.891.2

Вдовиченко А.І. Програмний модуль гейміфікації процесу тестування під час навчання. Магістерська кваліфікаційна робота зі спеціальності 122 – Комп'ютерні науки, освітня програма – Системи штучного інтелекту. Вінниця: ВНТУ 2022. 121 с.

На укр. мові. Бібліогр.: 24 назв; рис.: 27.

Досліджено важливість ролі гейміфікації під час розробки освітніх систем. Проаналізовано передумови застосування та основні принципи гейміфікації. Розглянуто основні принципи розробки існуючого програмного забезпечення для гейміфікації, а також здійснено аналіз його переваг та недоліків, на базі якого здійснено постановку задачі дослідження.

Розроблено алгоритм роботи модуля гейміфікації вивчення різних дисциплін, на базі якого створено відповідний програмний модуль. Для розробки програмного забезпечення було обрану об'єктно-орієнтовану мову програмування C#, а також використано бібліотеки LINQ to XML, LINQ to Objects, LINQ to Entities та інші. Основними перевагами розробленого модуля є відсутність необхідності обов'язкової авторизації, наявність кількох режимів тестування, а також дружність його інтерфейсу та лаконічність подання вікон та кнопок. Реалізовано адаптивне тестування з використанням машини нтучного інтелекту.

Розроблений програмний модуль протестовано у різних режимах та підтверджено коректність його роботи. Крім того тестування показало підвищення якості опанування матеріалу дисциплін. Мету роботи повністю досягнуто.

Ключові слова: геймифікація тестування, адаптивне тестування, освіта, програмний модуль, дисципліна.

## ABSTRACT

Vdovichenko AI Software module for gamification of the testing process during training. Bachelor's thesis in specialty 122 - Computer Science, educational program - Artificial Intelligence Systems. Vinnytsia: VNTU, 2022. 121 p.

In Ukrainian language. Bibliogr .: 24 titles; fig .: 27.

The importance of the role of gamification in the development of educational systems has been studied. Prerequisites and basic principles of gamification are analyzed. The basic principles of development of the existing software for gamification are considered, and also the analysis of its advantages and lacks is carried out, on the basis of which the statement of a research task is carried out.

The algorithm of work of the module of gamification of studying of various disciplines on the basis of which the corresponding software module is created is developed. The C # object-oriented programming language was chosen for software development, and the LINQ to XML, LINQ to Objects, LINQ to Entities, and other libraries were used. The main advantages of the developed module are the absence of the need for authorization, the presence of several test modes, as well as the friendliness of its interface and concise presentation of windows and buttons. The developed software module was tested in different modes and confirmed the correctness of its work. In addition, testing has shown an improvement in the quality of mastering the material of disciplines. The goal of the work has been fully achieved.



Key words: gamification of testing,adaptive testing, education, program module, discipline..

# ЗМІСТ

# ВСТУП

**Актуальність теми дослідження.** На сьогоднішній день значного поширення набуло поняття гейміфікації процесу навчання та дозвілля. Гейміфікацію застосовують у навчальних закладах для заохочення учнів та кращого засвоєння матеріалу. 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.

Актуальність геймифікації на сьогоднішній день зумовлена необхідністю зацікавити учнів у процесі навчання. Зі збільшенням рівня взаємодії учнів та студентів з інтерактивними програмами та гаджетами збільшився і рівень вимог до об`єкта та процесу, що беруть участь у навчанні, а саме їх можливість утримати увагу та інтерес учня, що значно підвищує кількість засвоюваної інформації. Особливо зіграло роль підвищення кількості дистанційного навчання в умовах пандемії та війни, коли необхідно зацікавити та мотивувати до здобування знань навіть у таких надважких умовах з викликами сучасності.

За допомогою інтерактивних елементів та ігрових модулів, що залучаються до виконання навчального завдання, динамічних геймінгових структур в процесі засвоєння нового матеріалу чи систем оцінок та винагород у складанні результатів оцінювання роботи учня, досягається набагато вищий рівень зацікавленості учнів у процесі навчання, підтримується більш якісний ступінь здорової конкуренції між учасниками навчального процесу, а також з`являються нові інструменти для моніторингу якості освіти на менеджменту архівів досягнень та здобутків тих, хто залучений до такого процесу навчання.

**Зв'язок роботи з науковими програмами, планами, темами.** Магістерська робота виконана відповідно до напрямку наукових досліджень кафедри комп’ютерних наук Вінницького національного технічного

університету 22 К1 «Моделі, методи, технології та пристрої інтелектуальних інформаційних сис/тем управління, економіки, навчання та комунікацій» та плану наукової та навчально-методичної роботи кафедри.

**Мета та задачі дослідження.** Метою дослідження є підвищення якості вивчення різних дисциплін шляхом впровадження гейміфікації процесу тестування та вдосконаленого алгоритму адаптивного тестування.

Для досягнення поставленої мети слід розв'язати такі задачі:

1) проаналізувати предметну область гейміфікації;

2) розробити алгоритм роботи програмного модуля гейміфікації;

3) вдосконалити алгоритм адаптивного тестування на основі моделі Раша;

4) реалізувати програмне забезпечення гейміфікації;

5) виконати тестування розробленого ПЗ;

6) проаналізувати переваги розробки перед аналогами.

**Об'єкт дослідження** – процес використання гейміфікації тестування під час вивчення різних дисциплін.

**Предмет дослідження** – програмне забезпечення для гейміфікації процесу тестування під час вивчення дисциплін.

**Методи дослідження**. У роботі використано такі методи наукових досліджень: теорія адаптивного тестування; класичний метод тестування; методи об'єктно-орієнтованого програмування.

**Наукова новизна.** Вдосконалено інформаційну модель Раша, яка відрізняється від існуючих тим, що додаткоо використовується модуль перевірки знань користувача, який дозволяє підвищити якість вивчення різних освітніх дисциплін.

**Практичне значення одержаних результатів** полягає у можливості використання розробленого веб-додатку для геймифікації вивчення різних дисциплін.

Розроблено доповнений алгоритм адаптивного тестування на основі моделі Раша.

Розробено програмне забезпечення для проходження тестування під час навчання.

**Достовірність теоретичних положень** магістерської кваліфікаційної роботи підтверджується коректністю постановки завдання, коректністю роботи програмних модулів, тестуванням програмної реалізації інформаційної технології гейміфікації роцесу тестування під час навчання. Адекватність розробленої технології підтверджується результатами тестування та експериментальних досліджень.

**Апробація.** Результати дослідження були представлені та обговорені на Всеукраїнській науково-практичній Інтернет-конференції студентів, аспірантів та молодих науковців «Молодь в науці: дослідження, проблеми, перспективи (МН-2024)» м.Вінниця, Україна(2023).

**Публікації.** За результатами дослідження опубліковано: 1 тези [1].

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ГЕЙМІФІКАЦІЇ ТЕСТУВАННЯ

## 1.1 Сучасний стан гейміфікації тестування

Лю та ін. (2017) визначають гейміфікацію як об'єднання елементів ігрового дизайну в цільову систему, зберігаючи інструментальні функції цільової системи. Це визначення починається з принципу, згідно з яким гейміфікація є ігровим рівнем у неігровій системі (Santhanam et al. 2016). Дизайн гейміфікації додає функції, зосереджується на стимулюванні участі користувачів і зберігає всі оригінальні інструментальні функції цільової системи, тоді як гра пожертвує деякими інструментальними функціями цільової системи, щоб зберегти свою розважальну цінність (Liu et al. 2017). ). Детердінг та ін. (2011b) стверджують, що гейміфікована система може бути або не бути в серйозному контексті, але вона точно не вимагає повноцінної системи, як ігри. У цьому дослідженні ми керуємося визначенням гейміфікації від Liu et al. (2017) як спосіб встановлення меж аналізу, відфільтровуючи будь-яку повноцінну гру. [3]

Оскільки дизайн гейміфікації запозичує елементи з відеоігор, він може стимулювати подібний гедонічний досвід, викликаючи ігрову поведінку гравців (Kankanhalli та ін. 2012). У той же час, щоб бути ефективним,

Системи гейміфікації повинні використовувати цей досвід, щоб змінити поведінку людини для досягнення бажаного результату. У літературі стверджується, що цей подвійний ефект гейміфікації необхідний для успіху, проте використовуючи різні концепції (Burke and Hiltbrand 2011; Hamari et al. 2014b; Kankanhalli et al. 2012; Nel et al. 1999; Webster and Ahuja 2006). [4]

Щоб забезпечити таксономію щодо термінології гейміфікації в IS, Liu et al. (2017) класифікують ефект елементів гейміфікації як експериментальні результати та інструментальні результати. Експериментальний результат зазвичай пов'язаний із сприйняттям користувача, таким як почуття, думка чи емоція, тоді як інструментальні результати пов'язані з утилітарним результатом гейміфікації. Лю та ін. (2017) також встановили набір принципів, які описують,

як гейміфікація може забезпечити значущу взаємодію. Автори стверджують, що експериментальні результати повинні відповідати контексту завдання з гейміфікованими елементами та бажаними інструментальними результатами. Враховуючи ширше визначення результатів гейміфікації, ми використовуємо систему подвійних результатів (Liu et al. 2017) для класифікації основних категорій результатів гейміфікації, представлених на рисунку 1.1.

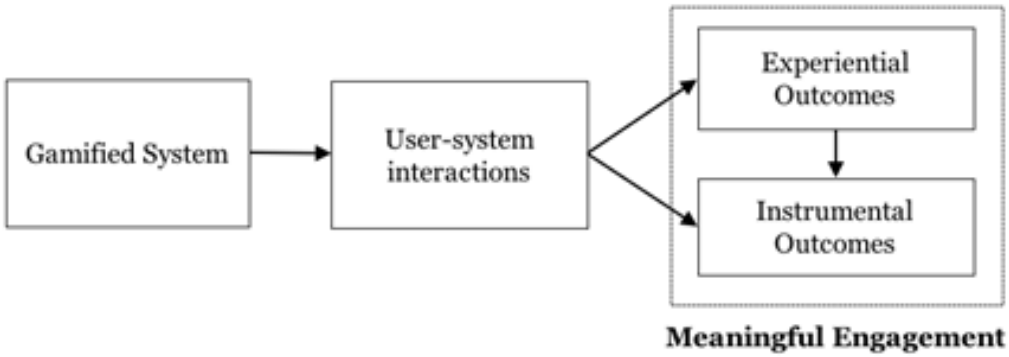



Рисунок 1.1 – Класифікація основних категорій геймифікації

Література, присвячена гейміфікації в освітньому контексті, успадковує характеристики літератури про гейміфікацію в цілому. Наприклад, принцип подвійного результату (досвід-інструментальний) також можна ідентифікувати в гейміфікованих системах, призначених для покращення навчання (Дічев та Дічева 2017; Фітц-Вальтер та ін. 2017; Ландерс 2014; Луєстин та ін. 2017). У цьому випадку експериментальні результати здебільшого пов’язані з гедонічними чи афективними результатами, тоді як інструментальні результати пов’язані з результатами навчання, які можна спостерігати, такими як успішність навчання, утримання та участь учнів (Дічев та Дічева 2017; Хамарі та ін. 2014b). Однак така сама невідповідність класифікації результатів, представлена в літературі з гейміфікації, також присутня в дослідженнях, присвячених освітньому контексту. Наприклад, в огляді літератури Дічев та Дічева (2017) перерахували часті категорії результатів, які застосовуються в освітніх гейміфікованих системах (тобто набуття знань, сприйняття, поведінка,

залучення, мотивація та соціальні), але не було надано додаткової інформації про те, чи ці результати є експериментальними або інструментальними.

## 1.2 Аналіз відомих систем геймифікації

Проаналізуємо деякі відомі найпопулярніші реалізації сучасних систем гейміфікації.

Duolingo [5] – це величезна онлайн-спільнота, яка поєднує в собі можливість вивчення мови через інтернет з платним сервісом для перекладу текстів. Послуга призначена для того, щоб студенти могли навчатися іноземної мови в Інтернеті, допомагаючи перекладати веб- сайти та документи. Користувачі-початківці займаються перекладом базових, простих речень з Інтернету, в той час як досвідчені отримують куди більш складні завдання. У результаті розвивається кожен, виконуючи свою частину «роботи».

У кожному з цих випадків Duolingo надає необхідні засоби для навчання і перекладу, щоб допомогти студентам правильно зрозуміти і запам’ятати слова, з якими вони стикаються. Кожен студент також може голосувати за якість перекладів інших користувачів, надаючи їм цінну інформацію для подальшого розвитку та покращення якості своєї роботи. Топ перекладів з кожного рівню доступний для публічного перегляду .

Поки студенти вивчають мову, вони заробляють очки вмінь за кожний завершений урок або перекладений контент. Чим більше завдань виконує «гравець», тим швидше зростає його «рівень». Ця реалізація в рамках інтернету приносить відверто більші плоди, ніж звичайний комп’ютеризований перекладач.

Сайт також містить в себе статистику, як багато часу знадобилося користувачеві для виконання завдання. Неправильні відповіді призводять до втрати очок і «життів» . Так як система адаптивна, вона відстежує кожен закінчений урок, переклад, тестування та практичне заняття, щоб забезпечити зворотний зв’язок для студента і планувати майбутні уроки і завдання

перекладу, які дозволять краще задовольняти їх потреби. Все це призводить до величезного успіху в сфері гейміфікації в освіті. На рисунку 1.2 та 1.3 зображено скріншот даного сайту.

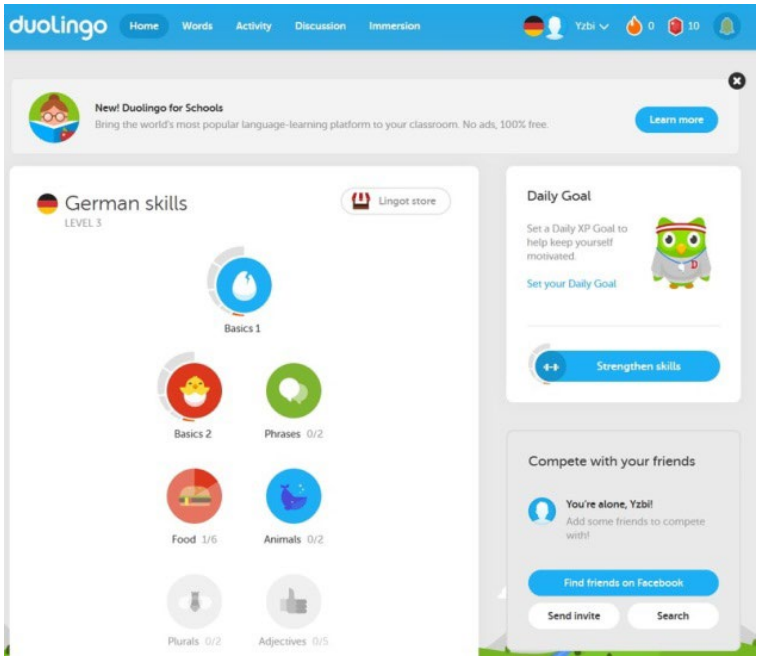



Рисунок 1.2 – Скріншот сайту Duolingo

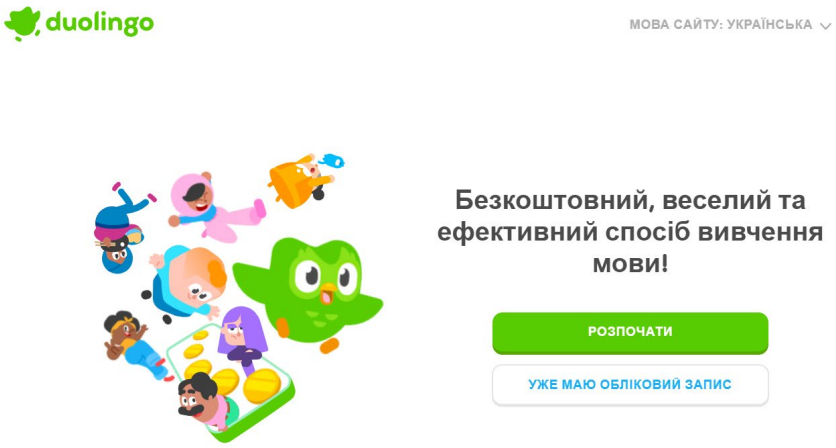



Рисунок 1.3 – Скріншот  першої стрінки Duolingo

Goalbook– це онлайн-платформа, яка допомагає вчителям, батькам і самим учням спільно відслідковувати прогрес навчання. Об'єднуючи особливості соціальних мереж та індивідуальної програми навчання (IEP), програма робить процес утворення простим і для студентів, і для викладачів. Учні спільно з вчителям формують цілі та кроки-завдання, які потрібно зробити щоб досягти цих цілей [6].

З GoalBook вчитель може легко отримати доступ до профілів всіх учнів і переглянути їх цілі. Учитель може стежити за успіхами кожного студента, з якою швидкістю і як саме вони виконують завдання. Коли мета буде досягнута, вчитель може швидко оновити анкету студента, а потім поділитися його успіхами з іншими. Через звичайний сайт педагог може легко оновлювати і відзначати досягнення будь-якого з учнів, а також бачити, як вони на них реагують.

Потужний інструмент гейміфікації для будь-якої спеціальної педагогічної освіти, GoalBook економить годинник обліку, що дозволяє негайно повідомляти батькам своїх учнів і первинним інструкторам про будь-які зміни, прогрес або проблеми. Відмінне рішення освітньої гейміфікації (рис.1.4-1.5).


Рисунок 1.4 – Скріншот онлайн-платформи Goalbook

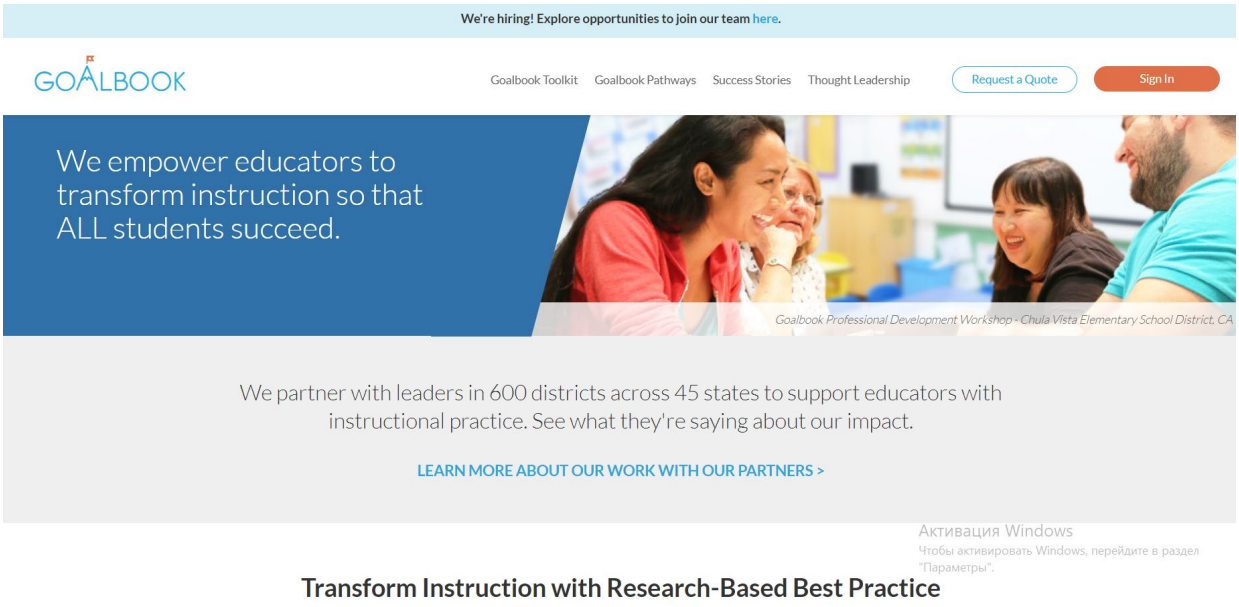



Рисунок 1.5 – Скріншот початкового екрану Goalbook

Course Hero [7] являє собою інтернет-платформу для навчання студентів і портал для педагогів, щоб поширювати свої освітні курси та програми. Сайт збирає і організовує навчальні матеріали, які були завантажені педагогами, і формує широке сховище даних. Платформа надає такі матеріали, як навчальні плани, екзаменаційні квитки і навчальні посібники. Крім того, Course Hero пропонує доступ до зворотного зв'язку з педагогами, цифрові картки і відеолекції.

Цифровий flash-додаток дозволяє студентам створювати свої власні навчальні програми, які можуть стати доступними для інших. Це дає можливість задавати темп вивчення, щоб допомогти максимізувати число учасників. Крім того, система винагороджує студентів різними рівнями доступу на основі їх результатів і оцінок.

Характерною особливістю Course Hero є «Розділ курсів», який пропонує широкий спектр безкоштовних і платних онлайн-курсів. Кожен курс зазвичай складається з розділів, які викладаються у формі відео і статей, а також постійним тестуванням для закріплення матеріалу.

Деякі курси додатково згруповані в три основні напрями: підприємництво, бізнес і веб-програмування. Для студентів, які закінчують 5 або більше курсів в одному з напрямків, Course Hero надає унікальні нагороди та заохочення. Наприклад, запрошення на презентацію власного бізнес-плану, співбесіду в хорошій фірмі та/або грошовий приз(рис. 1.6-1.7).

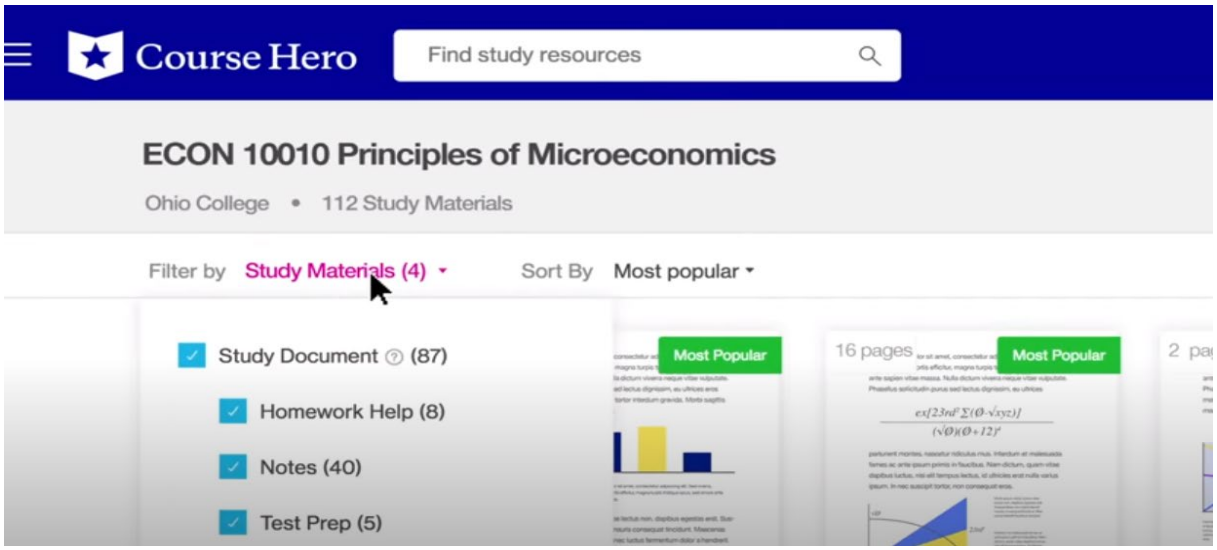



Рисунок 1.6 – Скріншот онлайн-платформи Course Hero

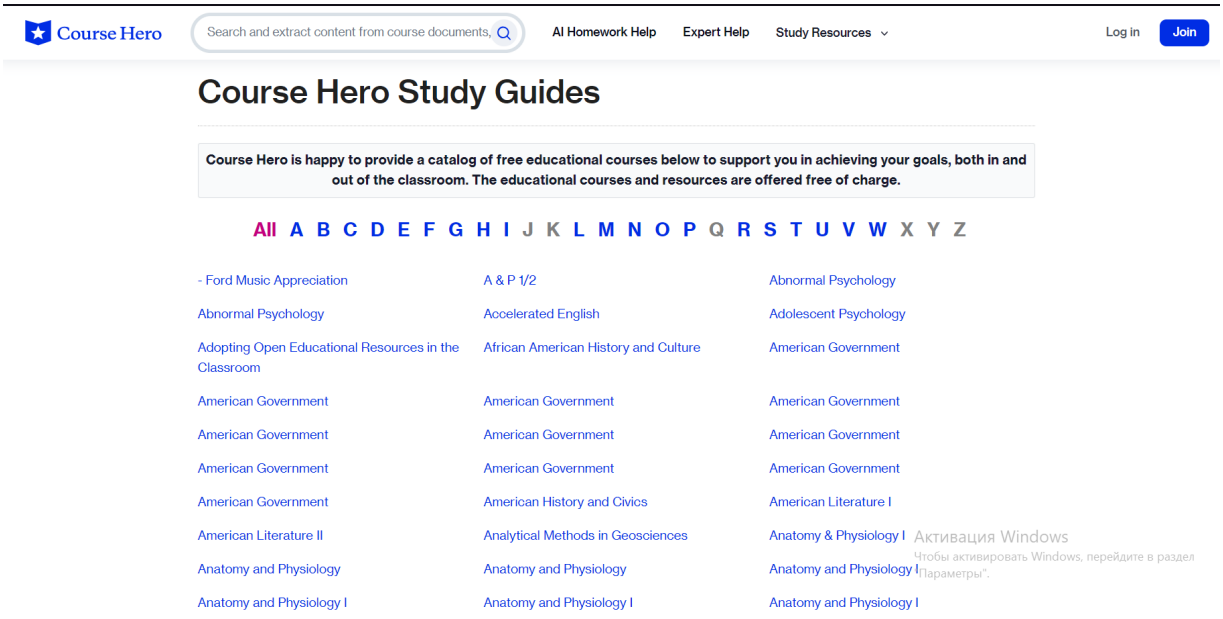



Рисунок 1.7 – Скріншот помічника онлайн-платформи Course Hero

QUIZZIZ - Популярний у світі сервіс, який забезпечує дистанційне навчання через створення поточних, контрольних та домашніх завдань у форматі вікторин і тестів, організації змагань. Налічує мільйони користувачів у понад 100 країнах світу.

Сайт містить величезну кількість готових завдань з найрізноманітніших дисциплін та галузей, однак учитель може створювати нові оригінальні вікторини та публікувати їх у своєму профілі. Забезпечується виконання завдань класом у режимі реального часу та можливість учителя відслідковувати результати кожного школяра, формувати звітні дані за проведену гру.

Доступний україномовний інтерфейс сайту. Зареєструватися можна через Google акаунт або введення даних вручну. Учнів додаємо у розділі «Класи» шляхом відправки школярам спеціального посилання-запрошення або через Google Classroom (якщо ви маєте зареєстровані класи на платформі).

Уже готовий тест/вікторину можна виконувати з учнями в режимі онлайн (кнопка «Играть вживую») або давати як домашнє завдання (кнопка «HW»). Під час роботи режиму Live учитель має змогу безпосередньо моніторити процес виконання роботи учнями(рис. 1.8).

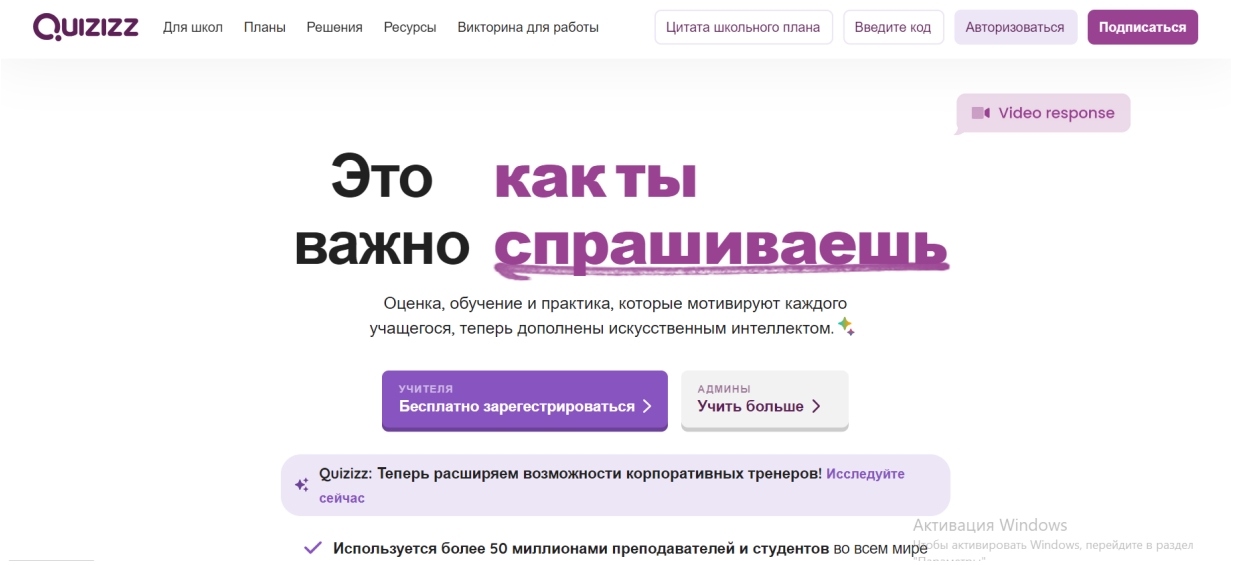



Рисунок 1.8 – Скріншот помічника онлайн-платформи QUIZZIZ

**Екзамен** - система, призначена для автоматизації процесу підготовки і проведення екзаменів, заліків й інших тестових заходів в навчальному закладі. Програмне забезпечення складається з двох модулів: автоматизованого робочого місця студента і серверного модуля. Автоматизоване робоче місце студента використовується для тестування і виконує такі функції: контроль доступу студента до тестових завдань, визначення часу виконання тестового завдання, а також перевірка відповідей і визначення кількості набраних балів. Серверний модуль поєднує функції тестового сервера та робочого середовища викладача при підготовці тестових завдань. Модуль виконує функції створення бази контрольних питань та зберігання інформації про студентів. Недоліками цієї системи є неможливість продовження тестування після втрати зв'язку з веб-сервером, визначення рівня підготовки користувачів і комерційного використання.

**Аргус-м** — це тестова система, реалізована як модуль PHP. Функціональність системи визначається можливостями сервера Apache, що використовується для зберігання контрольних матеріалів і результатів тестів. Особливістю системи Аргус-м є можливість створення завдання сертифікації на основі набору правил, об'єднаних у схемі сертифікації, яка визначає близько тридцяти параметрів: опис тестової області, з якої створюється завдання; правила відбору питань; обмеження кількості запитань і варіантів відповідей; правила навігації та перегляду процесу атестації; обмеження доступу тощо. Ця структура дозволяє створювати різноманітні завдання сертифікації, які можна легко повторити у разі повторної сертифікації. Недоліками системи є: неможливість створення та редагування персональних сертифікатів, відсутність підтримки W3C WAI та комерційність додатку.

**Brainbench** — це система перевірки знань для онлайн-сертифікації. Особливістю програми є наявність комерційних та безкоштовних тестів з різних дисциплін: інформатика, комп'ютерні мережі, прикладна лінгвістика тощо. Система створена за такими технологіями: XML, PHP та Java. Недоліками системи є: відсутність процесу авторизації й аутентифікації – через проходження

тестів іншим користувачем; використання пошукових систем глобальної мережі Інтернет при проходженні тестів; обмежена база запитань і неможливість їх заповнення у безкоштовних версіях програм.

**Neyron** — відкрита система тестування для перевірки знань користувачів у контексті процесу навчання. Ця система є спільною розробкою викладачів і студентів, переважно призначена для самооцінки студентів. Недоліками системи є: непридатність для використання в навчальному процесі через непродуману систему оцінювання знань та наявність тоталізатора для ставок на питання, що не відповідають принципам навчального процесу; неефективна система поповнення бази знань – через вільний доступ до неї будьякого користувача.

## 1.3 Постановка задачі дослідження

1. Огляд завдання

Необхідно створити інтерактивну тестову систему, яка надасть користувачам можливість пройти вікторину у різних режимах, з урахуванням різних функцій, включаючи нейромережевий модуль для аналізу та оцінки відповідей користувачів. Також в системі буде використовуватись адаптивна система тестування знань з використаннм технології машинного навчання.

2. Основні вимоги до системи:

2.1 Підтримка файлової системи.

Система повинна забезпечити зручний доступ та управління файлами, що містять вікторини, запитання та відповіді, у текстовому форматі.

2.2 Різні режими вікторини.

Система має надавати можливість користувачам обирати режим вікторини: за певний час, з врахуванням кількості чекпойнтів або в режимі «відповідь є у питанні».

2.3 Висока продуктивність.

Система повинна забезпечити обробку до 1000 питань та витримувати навантаження великої кількості користувачів одночасно.

2.4 Адаптація вітань.

Після завершення вікторини, система має виводити різні вітання в залежності від успішності користувача, з використанням нейромережевого модулю для вибору відповідного тексту.

2.5 Підтримка ОС Windows.

Система повинна бути сумісною з операційною системою Windows.

2.6 Безреєстраційний доступ

Система повинна дозволяти користувачам проходити вікторини без обов'язкової реєстрації та авторизації.

2.7. Завантаження тестів.

Система має надати можливість користувачам завантажувати свої власні тести, попередньо оформивши їх у текстовому форматі.

3. Нейромережевий модуль:

3.1 Аналіз та оцінка відповідей.

Нейромережевий модуль повинен аналізувати текстові відповіді користувачів та надавати оцінку їх правильності.

3.2 Генерація вітань.

Модуль також використовуватиметься для генерації вітань в кінці гри, з урахуванням стилю та успішності користувача.

4. Технічні обмеження:

4.1 Мови програмування.

Система повинна бути розроблена з використанням мови програмування, що підтримується ОС Windows.

4.2 Технічні вимоги.

Додаткові технічні вимоги та рекомендації надаються розробникам у документації.

5. Тестування та валідація:

5.1 Модульне тестування.

Система повинна пройти етап модульного та інтеграційного тестування перед випуском.

5.2 Валідація.

Важливі функції системи, включаючи нейромережевий модуль, повинні бути перевірені на реальних користувачах для валідації та збору відгуків.

6. Документація:

6.1 Технічна документація.

Повна технічна документація має бути розроблена для забезпечення зручності розробки, супроводу та розуміння системи.

## 1.4 Висновок до розділу 1

Обгрунтовано доцільність створення додатку гейміфікації процесу тестування під час навчання.

Розглянуто та проаналізовано предметну область систем гейміфікації тестування, досліджено їх основні особливості та принципи.

Проаналізовано ряд програм-аналогів, які забезпечують навчання у процесі гри, містять рейтинги та призи, а також заохочують та мотивують до вивчення. Наведено скріншоти роботи програм аналогів, для кращого розуміння області з якою маємо роботу. У результаті аналізу виявлено ряд недоліків подібних систем, серед них: недостатня універсальність додатку, необхідність обов’язкової авторизації, загальна складність інтерфейсу. Здійснено постановку задачі дослідження та сформульовано основні вимоги до розроблюваного програмного забезпечення для гейміфікації вивчення різних дисциплін.

# 2 ПРОЕКТУВАННЯ РОБОТИ ПРОГРАМНОГО МОДУЛЯ ГЕЙМІФІКАЦІЇ

## 2.1 Варіантний аналіз шляхів розробки модуля гейміфікації тестування

Алгоритм роботи програмного модуля гейміфікації можна побачити на рисунку 2.1.


Рисунок 2.1 – Алгоритм роботи модуля гейміфікації

Детальніше опишемо розроблений алгоритм. Спершу, користувач обирає з проводника та завантажує файл з потрібною вікториною, яка оформлена у правильному для проходження форматі. За це відповідає файлова система

додатку.  Файлова система – порядок, що визначає спосіб організації, зберігання та найменювання даних на носіях інформації в комп'ютерах, а також в іншому електронному обладнанні: цифрових фотоапаратах, мобільних телефонах і т.п. Файлова система визначає формат вмісту та фізичного зберігання інформації, яку прийнято групувати у вигляді файлів. Конкретна файлова система визначає розмір імені файлу (папки), максимальний можливий розмір файлу і розділу, набір атрибутів файлу. Деякі файлові системи надають сервісні можливості, наприклад, розмежування доступу або шифрування файлів. Файлова система пов'язує носій інформації з одного боку і API для доступу до файлів – з іншого. Коли прикладна програма звертається до файлу, вона не має жодного уявлення про те, яким чином розташована інформація в конкретному файлі, так само, як і те, на якому фізичному типі носія (CD, жорсткому диску, магнітній стрічці, блоці флеш-пам'яті або іншому) він записаний. Все, що знає програма – це ім'я файлу, його розмір і атрибути. [8]

На наступному кроці відбувається вибір режиму проходження вікторини, а саме можна обрати проходження вікторини за певний час, проходження вікторини з врахуванням кількості чекпойнтів, а також проходження вікторини в режимі «відповідь є у питанні». [9]

Далі користувач надає відповіді на усі питання  з вікторини, в ході проходження система повідомляє про коректність або некоректність наданої користувачем відповіді.

У випадку, якщо вікторину пройдено, користувач отримує вітальну картку згенеровану з допомогою нейромережевого модуля. Вітальні картки можуть бути декількох видів – кожна з них надається при наборі певної кількості балів за вікторину. Якщо ж користувач надав дуже мало правильних відповідей, або не надав їх взагалі – на екрані з'являється повідомлення про закінчення гри. Якщо вікторину не пройдено – повернення до попереднього кроку. [10]

Адаптивне тестування може бути організовано по-різному. Найпростіший варіант – це приписати кожному тестовому запитанню міру складності. При правильній відповіді студента наступне запитання буде більш складним, при

неправильній – менш складним. Також тестові запитання можуть бути прив’язані до певних тем, навчальних матеріалів. Якщо у ході відповіді студент зробить помилку, наступне запитання буде з тієї теми, яку він не досить добре знає. Якщо студент правильно відповість на запитання, йому пропонується запитання з іншої теми. При адаптивному тестуванні може використовуватись тимчасова модель студента, в яку заносяться дані про успішність студента в поточний момент. Після закінчення тесту можливе оновлення загальної моделі. Перехід від традиційної класичної теорії тестів до адаптивного тестування в практичному плані неминуче пов’язаний з деякими труднощами: [11]

- з необхідністю залучення досить складного математичного апарату;
- з використанням дорогої комп’ютерної техніки і розробки спеціальних програмних продуктів;
- знеобхідністю проведення ретельної роботи зі стратифікації вибірки випробовуваних статистик з математичними моделями вимірювання.

Адаптивне тестування – різновид тестування, при якому порядок представлення запитань (або складність) залежить від відповідей того, хто тестується, на попередні запитання [12], [13]. Використання адаптивних тестів дозволяє:

− підлаштовуватися під індивідуальні можливості студента – виключаються завдання, які або занадто складні, або занадто легкі;

− підвищити точність оцінки рівня знань сильних і слабких студентів завдяки використанню більшого банку запитань різного рівня складності;

− зменшити тривалість тесту і кількість поставлених запитань, необхідних для досягнення достатньої точності оцінки рівня знань студента;

− знизити ступінь втомленості студента;

− забезпечити конфіденційність за рахунок надання кожному студентові індивідуального набору тестових завдань, що відповідають його рівню знань;

− спростити процедуру внесення змін у банк тестових завдань, які будуть автоматично враховані адаптивним алгоритмом.

Цілі адаптивного тестування:

− орієнтація в навчальному процесі на адаптивні методи навчання і контролю; − організація на самостійну роботу учнів за допомогою адаптивних контрольнонавчальних програм;

− індивідуалізація навчання;

− використання методів адаптивного тестування як основи для реалізації методик для збільшення продуктивного навчання. У технології адаптивного тестування можна виділити наступні напрями:

− розробка і створення теоретико-методологічного обґрунтування тестування, методів, алгоритмів і засобів технології адаптивного тестування;

− розробка адаптивного тестування як методу навчання і контролю, що є частиною загальної дидактичної системи навчання, що знижує частку викладацької праці і дозволяє викладачеві перерозподілити час навчання і оптимізувати навчальний процес;

− розробка на базі адаптивного тестування методології розвивального, продуктивного навчання і інших педагогічних інновацій;

− розробка методик, направлених на збільшення частки самостійного навчання в освітньому процесі;

− адаптивне тестування як методика для організації процесу самоосвіти;

− розробка програмно-інструментальних засобів, що забезпечують функціонування адаптивного тестування.

Оцінка якості навчання є одним з основних чинників підвищення ефективності освітнього процесу. Такий аналіз є складною багатофакторною залежністю з великим числом змінних. Проведення подібного аналізу часто вимагає великих витрат сил і часу на проведення статистичних розрахунків. Оцінювання якості навчання з використанням комп’ютерних технологій дозволяє значно скоротити час і витрати, підвищує інформативність результатів. Тому доцільно використовувати комп’ютерне адаптивне тестування (КАТ).

## 2.2 Проектування UML-діаграм

UML (англ. Unified Modeling Language) — уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи, яка називається UML-моделлю. UML був створений для визначення, візуалізації, проєктування й документування в основному програмних систем. UML не є мовою програмування, але в засобах виконання UML-моделей як інтерпретованого коду можлива кодогенерація.

Перша версія (1.0) UML вийшла 13 січня 1997, вона була створена консорціумом UML Partners за запитом Object Management Group (OMG) — організації, відповідальної за прийняття стандартів в галузі об'єктних технологій і баз даних. Після обговорення, у вересні 1997 року, версія 1.1 UML була представлена на голосування в OMG. Розробку UML підтримали і вже тоді використовували як стандарт такі гранди ринку інформаційних технологій, як Microsoft, IBM, Hewlett-Packard, Oracle, DEC, Sybase, Logic Works й інші.
Поточна версія — 2.0.

UML може бути застосовано на всіх етапах життєвого циклу аналізу бізнес-систем і розробки прикладних програм. Різні види діаграм які підтримуються UML, і найбагатший набір можливостей представлення певних аспектів системи робить UML універсальним засобом опису як програмних, так і ділових систем.

Діаграми дають можливість представити систему (як ділову, так і програмну) у такому вигляді, щоб її можна було легко перевести в програмний код.

Основною причиною використання мови UML є спілкування розробників між собою.

Крім того, UML спеціально створювалася для оптимізації процесу розробки програмних систем, що дозволяє збільшити ефективність їх реалізації у кілька разів і помітно поліпшити якість кінцевого продукту.

UML чудово зарекомендувала себе в багатьох успішних програмних проєктах. Засоби автоматичної генерації кодів дозволяють перетворювати моделі мовою UML у вихідний код об’єктно-орієнтованих мов програмування, що ще більш прискорює процес розробки.

Практично усі CASE-засоби (програми автоматизації процесу аналізу і проєктування) мають підтримку UML. Моделі розроблені в UML, дозволяють значно спростити процес кодування і направити зусилля програмістів безпосередньо на реалізацію системи.

Діаграми підвищують супроводжуваність проєкту і полегшують розробку документації.

UML необхідний:

- керівникам проєктів, які керують розподілом завдань і контролем за проєктом
- проєктувальникам інформаційних систем які розробляють технічні завдання для програмістів;
- бізнес-аналітикам, які досліджують реальну систему і здійснюють інжиніринг і реінжиніринг бізнесу компанії;
- програмістам які реалізують модулі інформаційної системи.

При модифікації системи об'єктний підхід дозволяє легко включати в систему нові об'єкти і виключати застарілі без істотної зміни її життєздатності. Використання побудованої моделі при модифікаціях системи дає можливість усунути небажані наслідки змін, оскільки вони не ламають структури системи, а тільки змінюють поведінку об'єктів.

Діаграма класів (англ. class diagram) — статичне представлення структури моделі. Відображає статичні (декларативні) елементи, такі як: класи, типи даних, їх зміст та відношення. Діаграма класів може містити позначення для пакетів та може містити позначення для вкладених пакетів. Також, діаграма класів може

містити позначення деяких елементів поведінки, однак їх динаміка розкривається в інших типах діаграм.

Діаграма класів служить для представлення статичної структури моделі системи в термінології класів об'єктноорієнтованого програмування. На цій діаграмі показують класи, інтерфейси, об'єкти й кооперації, а також їхні відносини.

На рисунку 2.2 зображено діаграму класів для проектування програмного модуля гейміфікації вікторин освітнього спрямування. Діаграма класів має досить просту структуру, та всього чотири базових класи, що зумовлено віконною побудовою додатку.


Рисунок 2.2 – UML-діаграма класів додатку гейміфікації для вивчення різних дисциплін

Діаграма прецедентів (або діаграма варіантів використання) — в UML, діаграма, на якій зображено відношення між акторами та прецедентами в системі.

Діаграма прецедентів є графом, що складається з множини факторів, прецедентів (варіантів використання) обмежених межею системи (прямокутник), асоціацій між акторами та прецедентами, відношень серед прецедентів, та

відношень узагальнення між акторами.[1] Діаграми прецедентів відображають елементи моделі варіантів використання.

Суть діаграми прецедентів полягає в тому, що проєктована система подається у вигляді множини сутностей чи акторів, що взаємодіють із системою за допомогою так званих варіантів використання. Варіант використання (англ. use case) використовують для описання послуг, які система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, який виконує система під час діалогу з актором. При цьому нічого не говориться про те, яким чином буде реалізовано взаємодію акторів із системою.

У мові UML є кілька стандартних видів відношень між акторами і варіантами використання:

- асоціації (англ. association relationship)
- включення (англ. include relationship)
- розширення (англ. extend relationship)
- узагальнення (англ. generalization relationship)

При цьому загальні властивості варіантів використання можна подати трьома різними способами, а саме — за допомогою відношень включення, розширення і узагальнення.

Відношення асоціації — одне з фундаментальних понять у мові UML і в тій чи іншій мірі використовується під час побудови всіх графічних моделей систем у формі канонічних діаграм.

Включення (англ. include) у мові UML — це різновид відношення залежності між базовим варіантом використання і його окремим випадком. При цьому відношенням залежності (англ. dependency) є таке відношення між двома елементами моделі, за якого зміна одного елемента (незалежного) спричиняє зміну іншого елемента (залежного).

Відношення розширення (англ. extend) визначає взаємозв'язок базового варіанту використання з іншим варіантом використання, функціональна поведінка якого залучається базовим не завжди, а тільки за виконання додаткових умов.

На рисунку 2.3 зображено діаграму прецедентів для системи гейміфікації, що зображує які дії може проводити учень в процесі проходження вікторини.

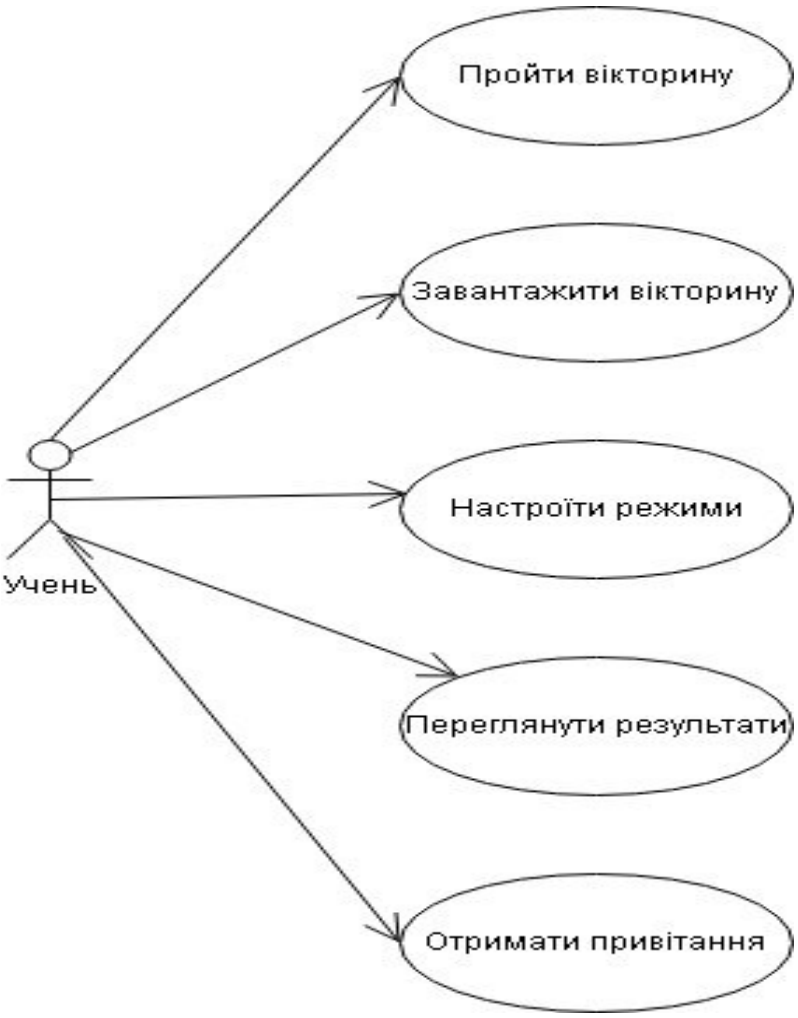



Рисунок 2.3 – UML-діаграма прецедентів додатку гейміфікації для вивчення різних дисциплін

Не менш важливим буде додати структурну схему геймифікації десктопного додатку.

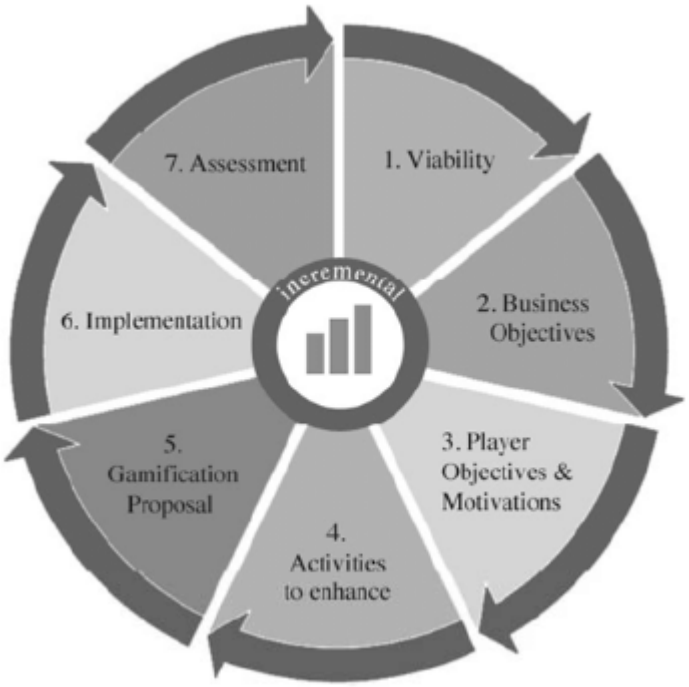



Рисунок 2.4 – Структурна схема фаз геймифікації додатку

На першому етапі фреймворку розглядається доцільність впровадження гейміфікації в організації програмного забезпечення. На другому етапі встановлюються деякі бізнес-цілі, щоб визначити, чи можлива гейміфікація. На третьому етапі досліджуються мотивації та профілі всіх професійних груп. Пізніше, на четвертому етапі, визначаються та обговорюються дії щодо гейміфікації, а також розглядаються деякі важливі аспекти пропозиції SPI. П'ятий етап є ядром гейміфікації. На цьому етапі розробляється пропозиція гейміфікації. Ця пропозиція спрямована на групи спеціалістів із програмного забезпечення. Крім того, встановлюються метрики та методи оцінювання, а також процеси зворотного зв’язку. На наступному етапі випускається впровадження пропозиції гейміфікації. Концепція гейміфікації завершується аналізом результатів і досягнутих цілей.

ЕТАП 1: ЗДІЙСНЕННІСТЬ Не всі ситуації підходять для застосування гейміфікації [Міттельмарк і Річчіо 12, Вербах і Хантер 12], і важливо передбачити це, щоб уникнути потенційних конфліктів [Берк 12]. Тому необхідно мати підтримку вищого керівництва для короткого аналізу діяльності

SPI та організаційного середовища [Herranz et al. 13]. Перед застосуванням фреймворку необхідно провести належну оцінку, щоб перевірити, чи сприятиме гейміфікація досягненню бізнес-цілей [Вербах і Хантер 12]. Крім того, перевірити, чи можливо підвищення мотивації та відданості, одночасно зменшуючи опір змінам.

ЕТАП 2: БІЗНЕС-ЦІЛІ Бізнес-цілі, очікувані від впровадження цієї основи, повинні бути визначені простим і реалістичним способом [Herranz et al. 13]. Крім того, щоб досягти стійкої пропозиції щодо гейміфікації, ці бізнес-цілі повинні бути узгоджені з цілями залучених груп [Вербах і Хантер 12]. Буде врахована інфраструктура SPI в організації.

ЕТАП 3: ЦІЛІ ТА МОТИВАЦІЯ КОРИСТУВАЧА Не всі однаково реагують на однакові подразники [Вербах і Хантер 12], і це можна побачити в будь-якій ініціативі SPI [Йогансен і Пріс-Хеє 07]. Таким чином, необхідно проаналізувати мотиваційні фактори, як внутрішні, так і зовнішні, залучених професійних груп, щоб виконати точний опис учасників (гравців) у пропозиції гейміфікації.

Для цього аналізу знадобиться вивчення мотиваційних факторів [Baddoo і Hall 02, 03] для кожної групи спеціалістів із програмного забезпечення (вище керівництво, керівники проектів і розробники). Буде доцільно ідентифікувати кожну групу професіоналів або ролі SPI з типом гравця в класифікації гравців для пропозиції гейміфікації [Bartle 96]. Хоча ця класифікація не призначена для узагальнення всіх видів ігор, вона вважається хорошою евристикою щодо того, як люди беруть участь у системі гейміфікації [Hägglund 12].

ЕТАП 4: ДІЯЛЬНІСТЬ ДЛЯ ПОКРАЩЕННЯ На цьому етапі деякі види діяльності будуть визначені та проаналізовані. Ці заходи будуть тими, які планується просувати в рамках пропозиції SPI. Під час цієї фази слід оцінити потенційний опір змінам і також визначити показники SPI, які визначають успіх або невдачу таких заходів.

ЕТАП 5: ПРОПОЗИЦІЯ ГЕЙМІФІКАЦІЇ Пропозиція гейміфікації є ядром структури гейміфікації та створює ціннісну пропозицію гейміфікації. Для цього

необхідно визначити: 1. Динамічні, механічні та ігрові елементи для ініціативи SPI. Цей пункт стосується пунктів, описаних у розділі 3.1. Спочатку визначається пропозиція високого рівня, а пізніше вона налаштовується (низький рівень, поглиблений) для кожної групи спеціалістів із програмного забезпечення. 2. Метрики для кожного з елементів гри, визначених вище. Буде необхідно контролювати результати пропозиції гейміфікації, щоб мотивувати та залучати всі професійні групи, включаючи вище керівництво [Herranz et al. 13]. 3. Процес зворотного зв'язку, за допомогою якого користувач отримує інформацію про свою діяльність у режимі реального часу. Це посилить його мотивацію та спонукатиме його продовжувати пропозицію [Herranz et al. 13]. При визначенні пропозиції гейміфікації слід враховувати деякі аспекти, відображені в моделях, пов’язані з опором змінам [Kotter 09] і з управлінням зобов’язаннями [Conner and Patterson 82]. Крім того, було б доцільно мати на увазі деякі ключові елементи, які оптимізують виконання завдань у гейміфікації під час застосування зміни поведінки [Frang and Mellstrand 12].

ЕТАП 6: ВПРОВАДЖЕННЯ На цьому етапі виконується та реалізується пропозиція гейміфікації з попереднього етапу на технологічному рівні [Herranz et al. 13]. Однак перш ніж реалізувати пропозицію, необхідно повідомити про це всіх в організації. Поточна ситуація, а також потреби в покращенні та цілі, яких необхідно досягти, повинні бути надані прозорим способом. Нарешті, щоб уникнути суперечок щодо використання ігрових елементів у робочому середовищі, не слід використовувати термін «гейміфікація» [Epstein 13].

ЕТАП 7: ОЦІНЮВАННЯ На останньому етапі схеми гейміфікації будуть проаналізовані результати та цілі, досягнуті в такій ітерації [Herranz et al. 13], а уроки, отримані під час впровадження процесу, будуть зібрані для керування пропозиціями SPI у наступних ітераціях [Layman 05].

## 2.3 Практичний аналіз доцільності застосування геймифікації при проектуванні додатку на конкретному прикладі

Розглянемо різницю між поняттям гри і поняттям геймифікацію, та пояснимо в яких випадках доцільно, а в яких недоцільно використовувати даний підхід та в яких обсягах.

Посилаючись на Бернарда Сьюта, гра містить у собі:

-Ціль – це конкретний результат гри, до якого прагнуть гравці. Вона привертає увагу гравця та координує його дії, в результаті в учасників виникає відчуття мети. Якщо це забіг на сто метрів, то ціль це пробігти від старту до фінішу. Якщо це шахи, то ціль – «вбити» короля опонента.

-Правила, що роблять активність грою. Правила встановлюють обмеження способи досягнення мети, що спонукає гравців досліджувати незвідані простору можливостей. У шахах заборонено просто скинути всі шахи опонента з дошки. Заборонено і ходити пішки через усю карту. Кінь – це кінь, королева – це королева.

-Система зворотного зв'язку повідомляє гравцям, наскільки вони близькі до досягнення мети. Ця інформація може бути представлена у вигляді очок, рівнів, рахунку або індикатора проходження гри. У найпростішому випадку система зворотного зв'язку зводиться до розуміння гравцем цільового результату: «Гра буде закінчена, коли…» Зворотний зв'язок у реальному часі ніби запевняє гравців, що ціль досяжна, і мотивує продовжувати гру.

І нарешті, добровільна участь вимагає, щоб кожен гравець усвідомлено та самостійно взяв мету, правила та систему зворотного зв'язку. Усвідомлення створює загальну основу спільної гри великої кількості людей. Тобто учасники дотримуються правил і роблять це добровільно. Якщо учасників примушують дотримуватись правил, це вже не гра.

А якщо гейміфікація походить від ігор, то вона також повинна містити в собі:

-добровільність участі

-навчання чомусь або вирішення певної проблеми.

-мета та правила досягнення цієї мети

-наратив. А саме структура, яка поєднує фрагменти гри або ігрову систему в єдине ціле. Інакше є ризик, що система гейміфікації буде просто купою абстрактних речей.

-правильний баланс системи. Нестача складності зробить гру нецікавою, а велика кількість правил і складності збільшить поріг входження і, відповідно, активну аудиторію системи.

Гейміфікація – це не спроба перетворити все на гру. Вона використовується не тільки для залучення користувачів. І це не лише бали, бейджі та таблиці лідерів.

Гейміфікація – це вивчення механіки ігор (не тільки віртуальних) з розумінням, чому людям подобаються ігри. Якщо розібратися в цьому питанні, то розробники можуть збільшити потік цікавості до їхніх додатків. Розібратися в цьому допоможе вивчення геймдизайну, психології та стосунків між людьми.

Піраміда елементів – це структура стандартних елементів у гейміфікації та зв'язок між цими елементами.

Ця піраміда не відбиває всі можливі елементи. І це не означає, що кожен елемент слід впроваджувати у свою систему. На рис. 2.4 зображено піраміду елементів.

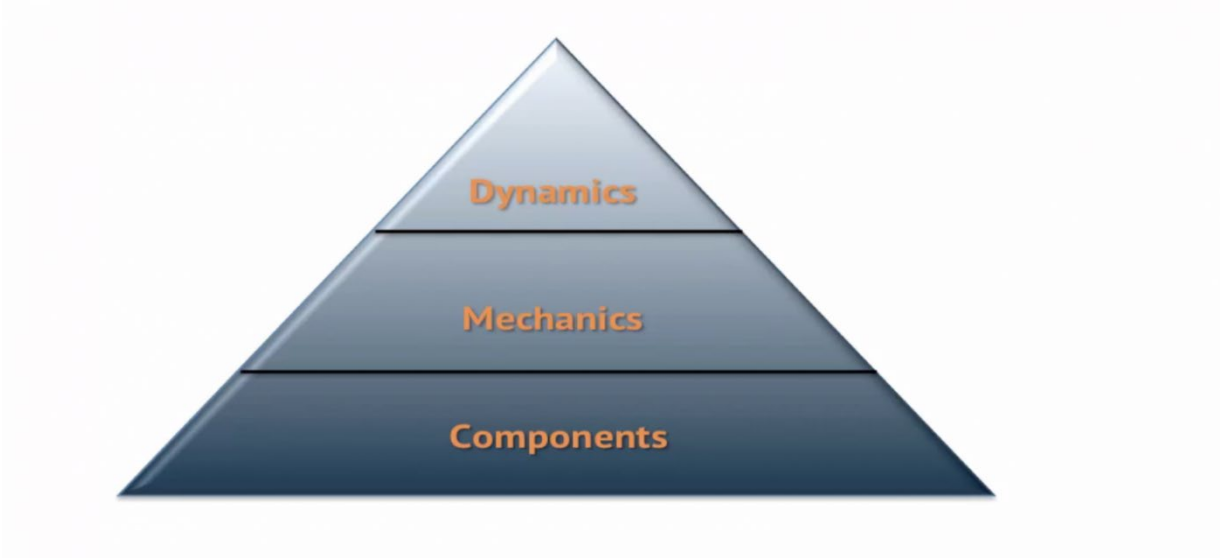



Рисунок 2.4 – Піраміда елементів

На верху піраміди знаходяться Динаміки. Це найбільш високорівневі концепції в іграх чи гейміфікованих системах. ЇХ можна уявити це як граматику в мові, а саме заховані структури, які роблять весь досвід пов'язаним та мають регулярні патерни.

Цей рівень містить:

Constraints – Кожна гра повинна мати обмеження. Штучно обмежуючи свободу, ігри створюють осмислену мету та додають цікаві проблеми, вирішення яких не дратує гравців. Таким чином, уявлення про те, які обмеження накладаються на користувачів є важливою динамікою, про яку повинен думати будь-який дизайнер гейміфікованих систем.

Emotions – Ігри можуть викликати практично будь-які емоції, від радості до смутку та всього, що між ними. Емоційна палітра гейміфікації, як правило, дещо обмеженіша, тому що ми маємо справу з реальним світом, неігровим контекстом. Але все ще існує безліч емоційних важелів, які можна використовувати, щоб зробити досвід користувача більш насиченим.

Narrative – структура, яка поєднує фрагменти гри або ігрової системи в єдине ціле. Наратив може бути явним, сюжетною лінією у грі чи неявним. Без наративу є ризик, що гейміфікована система буде просто купою абстрактних речей.

Progressive – це означає, що користувач бачить початок шляху, у якого він зараз знаходиться, і те, куди він прийде якщо буде дотримуватися правил. Прогресія не обов'язково вимагає присутності рівнів, шкал прогресу та окулярів. Але це типові компоненти, що використовуються створення даної динаміки

Relationships – ця динаміка розповідає про те, як люди взаємодіють один з одним. Сюди входять друзі, члени команд та противники. Ці динаміки дуже добре розвинені у таких іграх як WoW, La2 та інших MMO іграх.

Потім йдуть механіки — набір правил і способів, який певним чином реалізує деяку частину інтерактивної взаємодії гравця та гри. Усі безліч ігрових механік гри формують конкретну реалізацію її ігрового процесу.

І найнижчий рівень піраміди, компоненти. З цими елементами користувачі взаємодіють безпосередньо і, переважно, безпосередньо спостерігають лише цей рівень. Це певні блоки, з яких збираються механіки та динаміки.

Є ще один рівень поза цією пірамідою – це естетика (Aesthtic). Вона визначає бажані емоції, які мають виникнути у гравця, коли він взаємодіє з грою.

Для гейміфікованих систем, як і для ігор, немає правил щодо того, що правильно, а що ні. Але є моменти, яких краще уникати, особливо якщо у розробника немає досвіду впровадження геймифікації.

-Примушувати скористатися системою. Головне правило – гра має бути добровільною.

-Заохочувати та карати матеріально.

-Впроваджувати гейміфікацію для гейміфікації. Гейміфікація повинна спрямовувати людей і допомагати їм вирішувати проблеми, або вчитися.

-Робити дуже багато правил і обмежень. Чим більше правил, тим складніше запам'ятати їх, чим більше обмежень, тим більше обмежений гравець. А ніхто не любить, коли їх обмежують.

-Не продумувати баланс. У грі має бути баланс та прогресія складності, яка враховуватиме як інтереси новачків, так і потреби майстрів.

## 2.4 Основні поняття та величини для адаптивного тестування

Однопараметрична модель Раша Вводяться 2 основних параметри: θ – латентний параметр рівня знань учасника тестування і b – параметр складності завдання. Для однопараметричної моделі Раша використовується формула правильної відповіді на питання

$$P_j = \frac{e^{(\theta - b_j)}}{1 + e^{(\theta - b_j)}} \quad (2.1)$$

Ймовірність правильної відповіді на питання залежить тільки від різниці між рівнем знань учасника тестування і складністю питання. Модель Раша не

дозволяє питанням розрізнятися за дискримінацією: питання можна розташувати тільки за рівнем їх складності.

Двопараметрична модель Раша Введемо початкові величини та поняття. Адаптивний тест складається із m питань. Нехай маємо змінні відповіді $U_j, j = 1,2, \dots, n.$ які можуть приймати значення 1, якщо відповідь правильна, і 0, коли невірна, тобто

$$U_j = \begin{cases} 0, \text{якщо відповідь неправильна} \\ 1, \text{якщо отримана правильна відповідь} \end{cases}$$

Логістична модель з двома параметрами (2PL модель):

$$P\left(U_j = 1 \middle| \theta\right) = \frac{e^{a_j(\theta - b_j)}}{1 + e^{a_j(\theta - b_j)}} \qquad 2.2$$

де $a_j$ – диференціююча здатність тестового j завдання(коефіцієнт дискримінації), $b_j$ – складність завдання j, $\theta$ – латентий параметр рівня знань учасника тестування. Будемо мати на увазі, що $i$ = 1,2..m; $j$ = 1,2, …, $n$, $m$ – кількість учасників тестування, n – кількість тестових завдань.[18-19]

## 2.5 Висновок

В даному розділі спроектовано простий алгоритм роботи додатку гейміфікації процесу тестування, що враховує всі необхідні елементи, які мають бути залучені до навчального процесу, а також окреслено ряд методик, що задовольняють всі необхідні стандарти навчання, для проведення тестування майбутнього додатку,який буде створено згідно зі всіми поставленими вимогами та деталями до технічного завдання. Побудовано декілька наочних UML-діаграм, що відображають елементи та можливості майбутнього додатку, для кращого розуміння структури програми та візуалізації потреб користувача, який буде брати участь у навчальному процесі з використанням технології геймифікації та додатку, що ці технології вводить в навчальний процес. Описано принцип роботи алгоритму адаптивного тестування та моделі Раша. Вдосконалено алгоритм за рахунок нового модуля.

# 3 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ПРОГРАМНОГО МОДУЛЯ ГЕЙМІФІКАЦІЇ

## 3.1 Обґрунтування середовища і мови програмування

На сьогоднішній день існує надзвичайно багато мов програмування. Усі вони досить потужні, мають багато можливостей, але кожна з них має свої особливості. Проаналізуємо найбільш поширені мови програмування, а саме C#, C++ і Java(рис. 3.1-3.3).

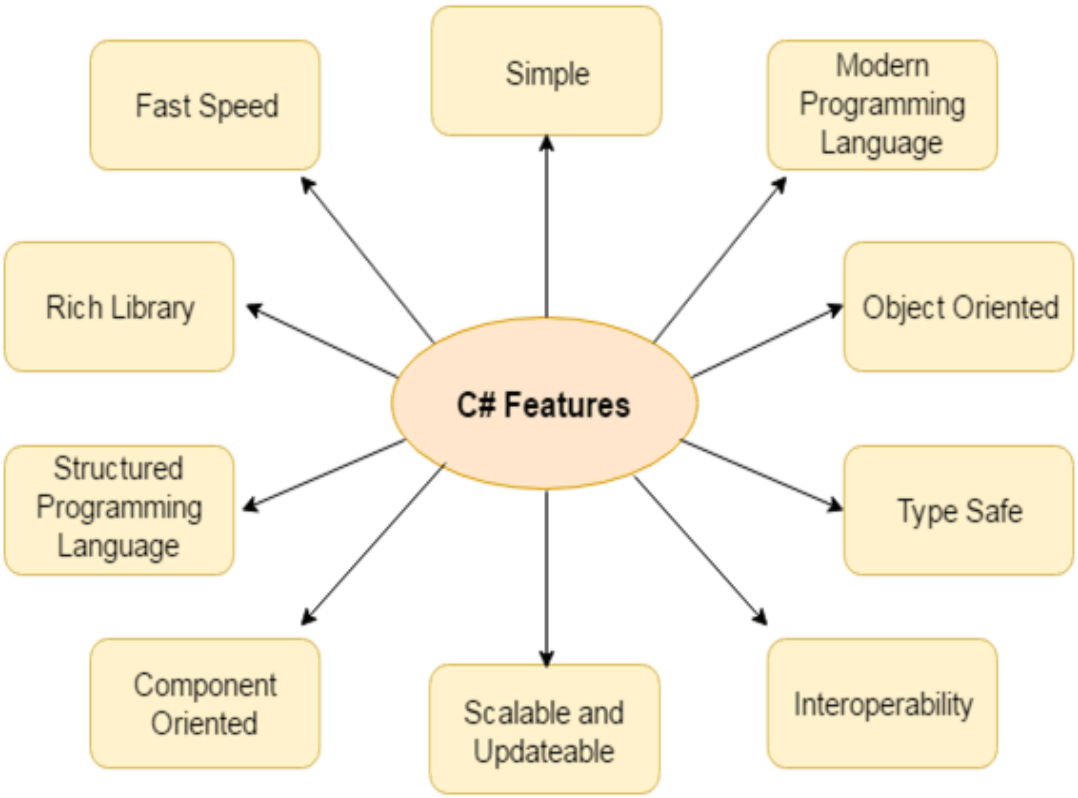



Рисунок 3.1 – Схематичне зображення переваг мови C#

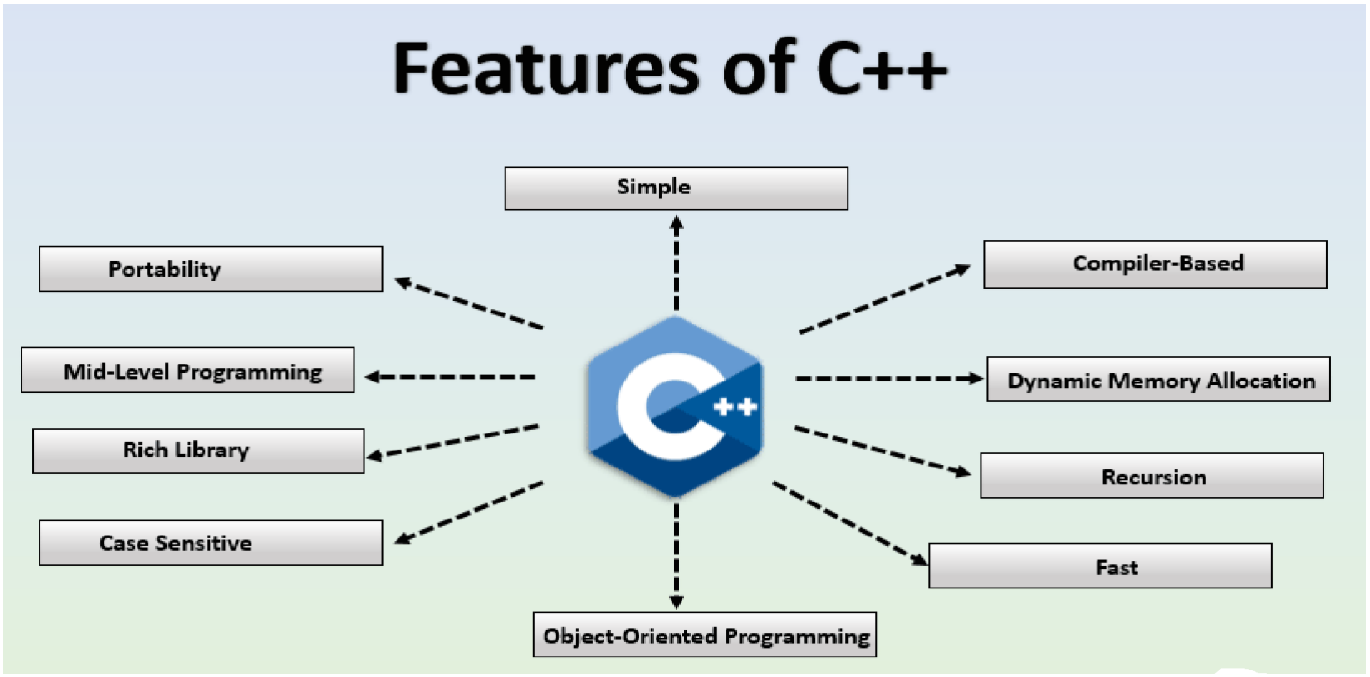



Рисунок 3.2 – Схематичне зображення переваг мови C++

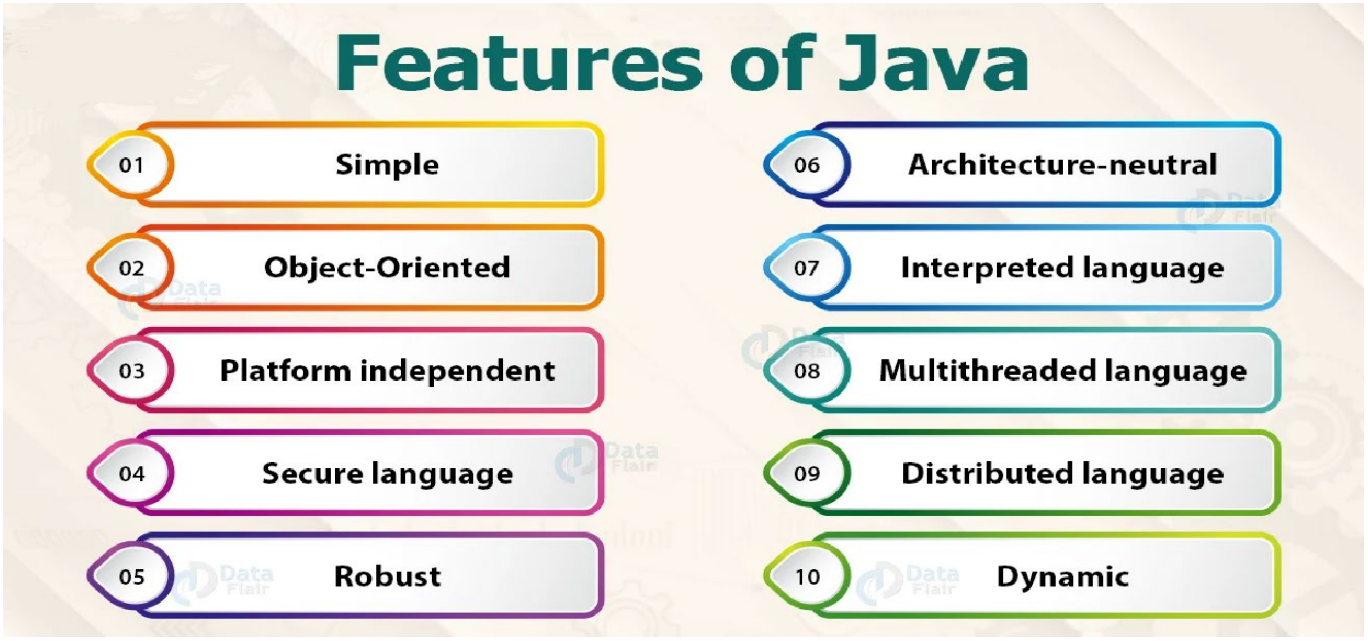



Рисунок 3.3 – Схематичне зображення переваг мови Java

Порівняльну характеристику трьох вищенаведених мов програмування, що дозволяє проаналізувати переваги і недоліки наведено у таблиці 3.1.

Таблиця 3.1 – Порівняльна характеристика деяких мов програмування

| Мова програмування | Сфера застосування | Переваги | Недоліки |
|---|---|---|---|
| C# | Розробка ОС та офісних рішень для платформи Microsoft, розробка веб-додатків, настільних графічних додатків (використовуються платформи Windows Forms и WPF), серверних додатків в сфері створення динамічного веб-контенту за допомогою платформи ASP.NET, мобільних додатків для операційної системи Windows Phone, що розроблена компанією Microsoft. | • Висока швидкість розробки в малобюджетних проектах<br>• Простота і лаконічність коду<br>• Наявність власного середовища розробки<br>• Можливості спадкування й універсалізації<br>• Типи, вбудовані в мову, представлені класами<br>• Збереження і об’єднання кращих рис мов C і C++ | • Проблеми при розробці додатків під не-Windows платформи<br>• Сильна прив’язка до Microsoft<br>• Нема самодостатньості додатків, так як використовується .net framework<br>• Відсутність повноцінних деструкторів, повноцінних макросів, дуже обмежена підтримка шаблонів |

| | | | |
|---|---|---|---|
| | | • Підвищення ефективності коду, середовище CLR являє собою компілятор проміжної мови<br>• Зручність побудови різних типів додатків дозволяє легко розробляти Web-служби | |
| C++ | Створення ОС, прикладних програм, драйверів приладів, додатків для вбудованих систем, високопродуктивних серверів, а також програмування ігор. | • Висока сумісність з мовою C<br>• Підтримка різних стилів програмування<br>• Можливість побудови загальних контейнерів і алгоритмів для різних типів даних<br>• Кросплатформеність | • Погано продуманий синтаксис робить мову незручною в деяких задачах<br>• Нема самодостатньості додатків, для цього використовується runtime<br>• Продуктивність праці програмістів на мові - |

Продовження таблиці 3.1

| | | | |
|---|---|---|---|
| Java | Створення десктопів і аплетів, мобільних додатків, серверних додатків, що орієнтовані на роботу з мережею, програмування ігор. | • Універсальність і широта застосування<br>• Кросплатформеність<br>• Відкритий вихідний код і велика кількість бібліотек<br>• Гнучка система безпеки<br>• Висока продуктивність<br>• Багатозадачність<br>• Незалежність від архітектури, тобто оновлення ОС, модернізація процесора чи зміна об’єму пам’яті не спричиняє збоїв програми | • Мова не має таких властивостей об'єктно-орієнтованої мови, як індивідуальні змінні та множинне спадкування<br>• Відсутність спливаючих підказок до методів, які можна примінити до певного об’єкта<br>• Низька швидкодія, підвищені вимоги до об’єму оперативної пам’яті |

Отже, по-перше, C# є актуальною мовою на даний момент, оскільки дозволяє найбільш раціонально створювати популярні Інтернет-додатки. По-друге, мова C# тісно інтегрована з мовою XML та різноманітними веб-технологіями. По-третє, вона інтегрувала в собі переваги мов C++ і Java, що й зумовлює популярність даної мови серед розробників. Нарешті, у C# виключені деякі спірні директиви та макроси.

Для розробки даної системи використовувалося середовище розробки Microsoft Visual Studio 2017. Visual Studio включає в себе редактор вихідного коду з підтримкою технології IntelliSense і можливістю найпростішого рефакторинга коду. Вбудований дебагер може працювати як відладчик рівня вихідного коду, так і відладчик машинного рівня. Решта вбудованих інструментів включають в себе редактор форм для спрощення створення графічного інтерфейсу додатку, веб-редактор, дизайнер класів і дизайнер схеми баз даних. Visual Studio дозволяє створювати і підключати сторонні додатки (плагіни) для розширення функціональності практично на кожному рівні, включаючи додавання підтримки систем контролю версій вихідного коду (як, наприклад, Subversion і Visual SourceSafe), додавання нових наборів інструментів (наприклад, для редагування і візуального проектування коду на предметно-орієнтованих мовах програмування) або інструментів для інших аспектів процесу розробки програмного забезпечення (наприклад, клієнт Team Explorer для роботи з Team Foundation Server). Робота та характеристики Visual Studio знаходяться на рисунках 3.4-3.5 .

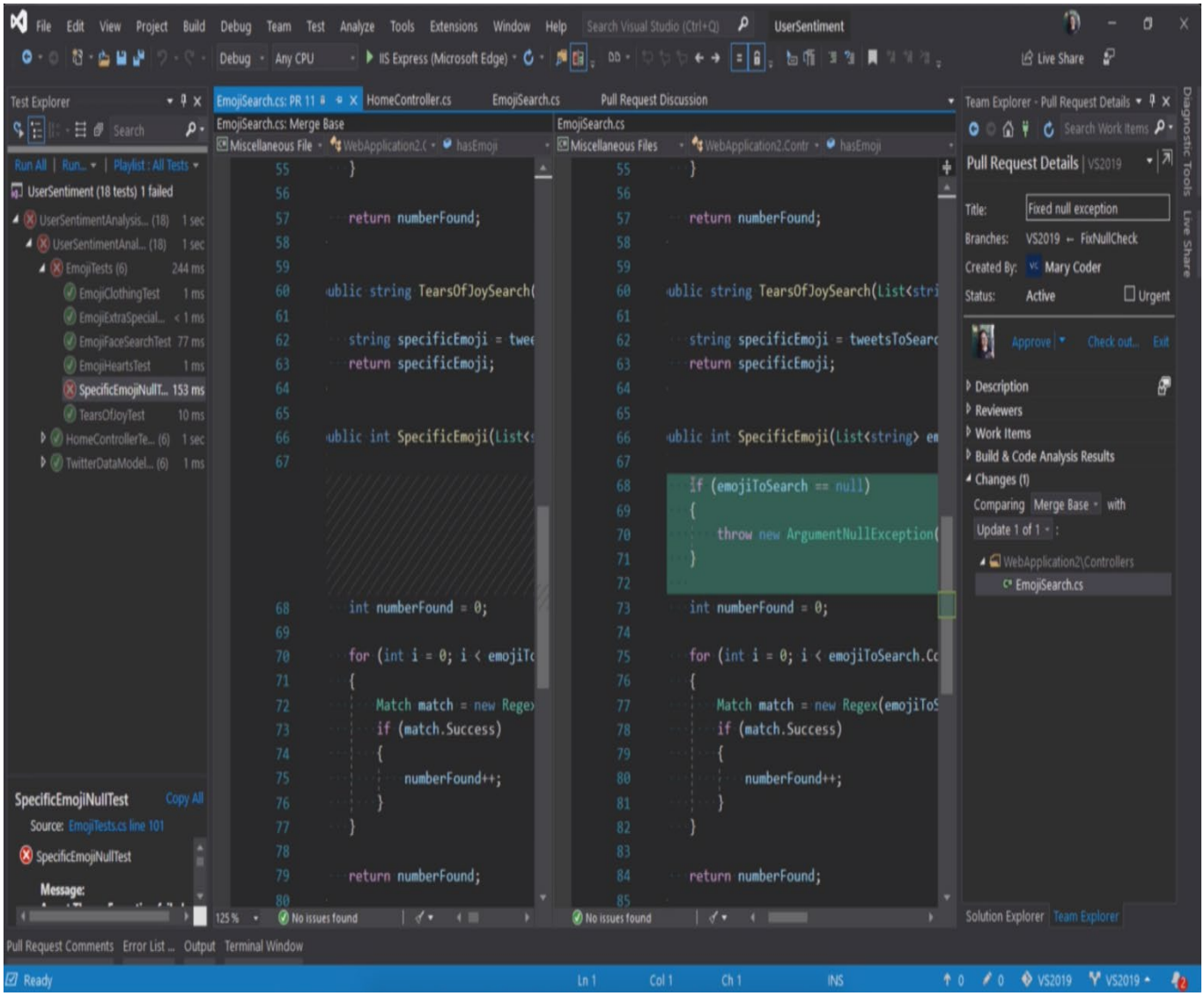



Рисунок 3.4 – Скріншот додатку Visual Studio

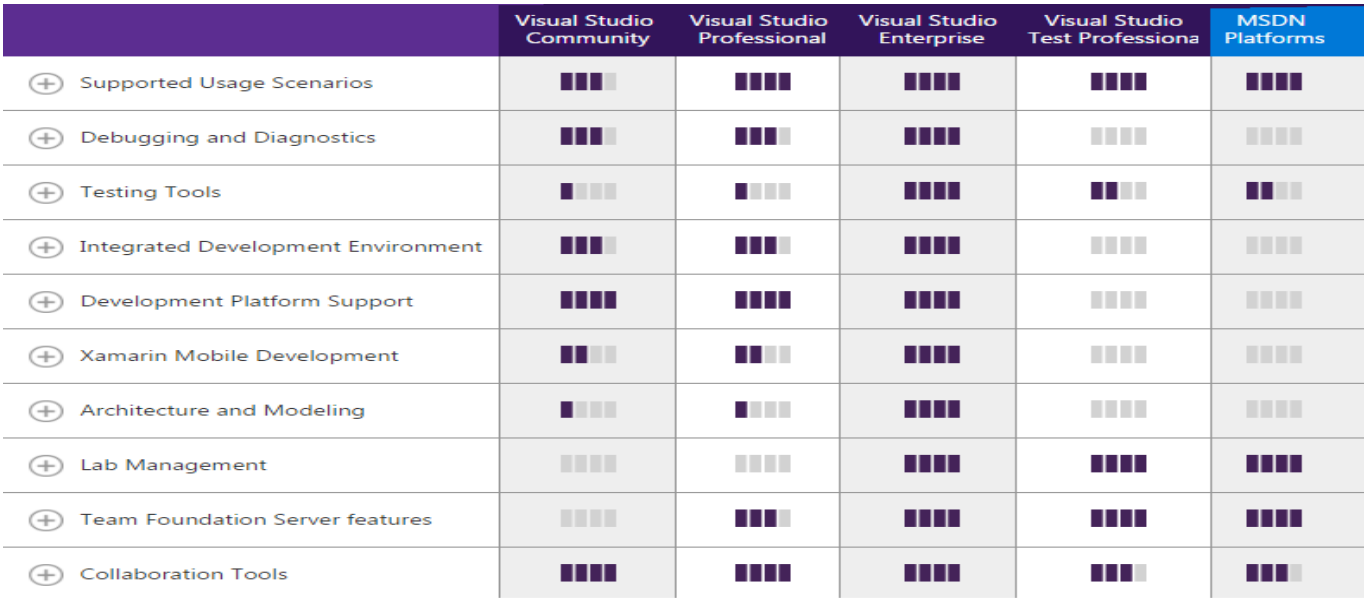



Рисунок 3.5 – Порівняння різних версій Visual Studio

### 3.2 Розробка програмного модулю гейміфікації

При ООП програма являє собою опис об'єктів, їх властивостей (або атрибутів), сукупностей (або класів), відносин між ними, способів їх взаємодії і операцій над об'єктами (або методів). Безперечною перевагою даного підходу є концептуальна близькість до предметної області довільної структури і призначення. Механізм успадкування атрибутів і методів дозволяє будувати похідні поняття на основі базових і таким чином створити модель як завгодно складної предметної області з заданими властивостями.

Ще однією теоретично цікавою і практично важливою властивістю ООП є підтримка механізму обробки подій, які змінюють атрибути об'єктів і моделюють їх взаємодія в предметної області. Переміщаючись по ієрархії класів від загальних понять предметної області до конкретніших (або від більш складних до більш простих) і навпаки, програміст отримує можливість змінювати ступінь абстрактності або конкретності погляду на модельований їм реальний світ.

Використання раніше розроблених (можливо, іншими колективами програмістів) бібліотек об'єктів і методів дозволяє значно заощадити трудовитрати при виробництві програмного забезпечення, особливо типового.

Об'єкти, класи і методи можуть бути поліморфними, що робить реалізоване програмне забезпечення більш гнучким і універсальним.

Складність адекватної (несуперечливої і повної) формалізації об'єктної теорії породжує труднощі тестування і верифікації створеного програмного забезпечення. Мабуть, ця обставина є одним з найбільш істотних недоліків ООП.

Найбільш відомим прикладом об'єктно-орієнтованої мови програмування є мова C++, яка розвинулась з імперативної мови C. Її прямим нащадком і логічним продовженням є мова C#, яка досліджується в якості зразка. Інші приклади об'єктно-орієнтованих мов програмування: Visual Basic, Java, Eiffel, Oberon.

Перехід від структурно-процедурного підходу до програмування до об'єктно-орієнтованого, подібно переходу від низькорівневих мов програмування до мов високого рівня, вимагає значних витрат на навчання. Природно, що платою є підвищення продуктивності купа програмістів при проектуванні і реалізації програмного забезпечення. Інша перевага ООП (об’єктно-орієнтовне програмуваня) перед імперативним підходом – вищий відсоток повторного використання вже розробленого програмного коду. При цьому, на відміну від попередніх підходів до програмування, ООП вимагає глибокого розуміння основних принципів, або, інакше, концепцій, на яких він базується(рис. 3.6).

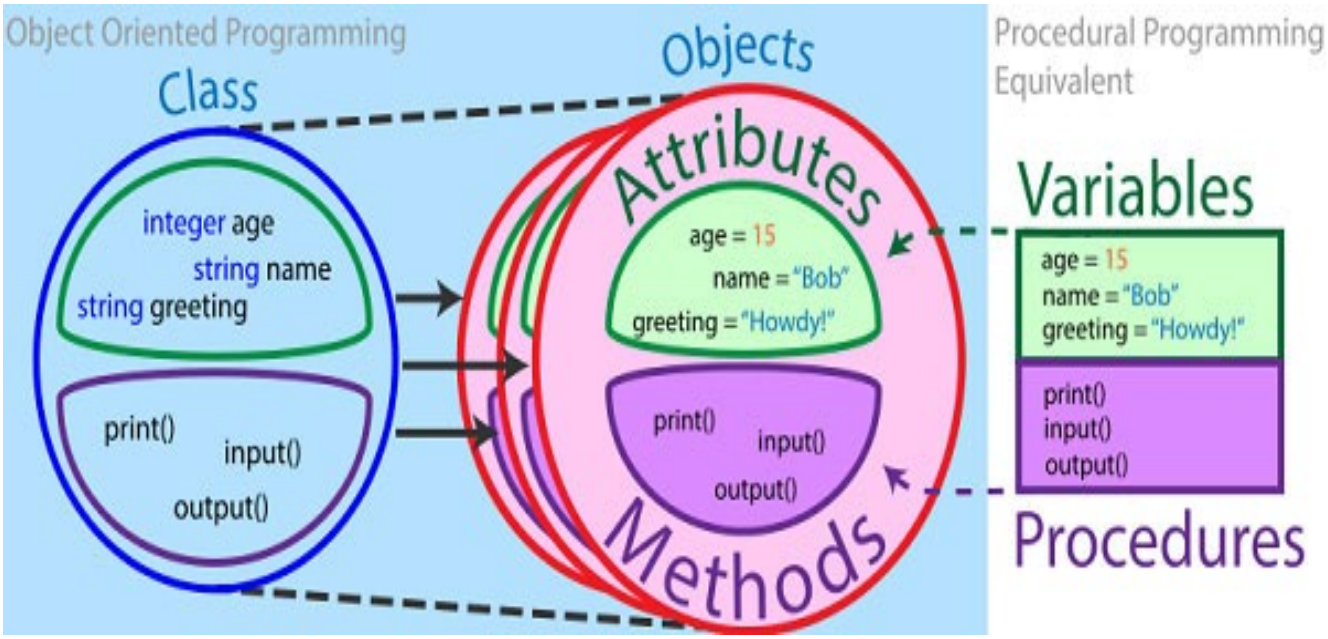



Рисунок 3.6 – Базова схема понять ООП

До числа основних понять ООП зазвичай відносять абстракцію даних, спадкування, інкапсуляцію і поліморфізм.

Виділяють інші фундаментальні принципи ООП. Так, спадкування конкретних атрибутів об'єктів і функцій оперування об'єктами засноване на ієрархії.

Інкапсуляція означає «приховування» властивостей і методів всередині об'єкта.

Поліморфізм, як і в функціональному програмуванні, розуміється як наявність функцій з можливістю обробки даних змінного твань.

Розглянемо більш докладно такий фундаментальний принцип ООП, як абстракція. У розділах математики, які досліджують моделювання процесу створення програм, під абстракцією прийнято розуміти довільне вираження мови програмування, яке є відмінним від ідентифікатора.

Важливою операцією, є операція обчислення значення виразу або команди, тобто операція означивания (зокрема, функція обчислення значення явно використовується при побудові семантики мови програмування). У зв'язку з цим

важливо встановити, що є значенням абстракції. Будемо вважати, що значення функції або змінної може бути присвоєно абстракції і є значенням останньої.

В об'єктно-орієнтованому програмуванні кожен об'єкт являє собою принципово динамічну сутність, тобто змінюється в залежності від часу (а також від впливу зовнішніх по відношенню до нього факторів). Інакше кажучи, об'єкт володіє тим чи іншим способом поведінки. Відносно абстракції як об'єкта поводження полягає в додатку функції до аргументу.

Уже відзначено, що концепція абстракції в об'єктно-орієнтованому програмуванні адекватно моделюється за допомогою лямбда-числення. Точніше кажучи, операція абстракції в повній мірі є моделлю однойменного поняття ООП.

Інший фундаментальною складовою концепції об'єктно-орієнтованого програмування є інтуїтивно ясне поняття спадкування. У неформальній постановці під спадкуванням розуміється властивість того чи іншого об'єкта, який є похідним від якогось базового об'єкта, зберігати поведінку (а саме, атрибути і операції над ними), характерне для батьківського об'єкта.

З точки зору мов програмування поняття спадкування означає застосовність всіх або лише деяких властивостей і (або) методів базового (або батьківського) класу для всіх класів, похідних від нього. Крім того, збереження властивостей і (або) методів базового класу має забезпечуватися і для всіх конкретизацій (тобто конкретних об'єктів) будь-якого похідного класу.

У процесі розробки використано найвідоміші бібліотеки.

Узагальнені типи є у багатьох бібліотеках базових класів .NET 2.0, але простір імен System.Collections.Generic буквально наповнений ними (що цілком відповідає його назві). Подібно до свого "родича" без узагальнень (System.Collections), простір імен System.Collections. Generic містить безліч типів класу і інтерфейсу, що дозволяє вкладати елементи в різні контейнери. Не дивно, що узагальнені інтерфейси імітують відповідні неузагальнені типи з простору імен System.Collections.

LINQ (Language-Integrated Query) представляє просту та зручну мову запитів до джерела даних. Як джерело даних може бути об'єкт, що реалізує інтерфейс IEnumerable (наприклад, стандартні колекції, масиви), набір даних DataSet, документ XML. Але незалежно від типу джерела LINQ дозволяє застосувати до всіх той самий підхід для вибірки даних.

Існує кілька різновидів LINQ:

LINQ to Objects: застосовується для роботи з масивами та колекціями

LINQ to Entities: використовується при зверненні до баз даних через технологію Entity Framework

LINQ to XML: застосовується під час роботи з файлами XML

LINQ to DataSet: застосовується під час роботи з об'єктом DataSet

Parallel LINQ (PLINQ): використовується для виконання паралельних запитів.

Windows Forms — інтерфейс програмування програм (API), який відповідає за графічний інтерфейс користувача і є частиною Microsoft .NET Framework. Цей інтерфейс спрощує доступ до елементів інтерфейсу Microsoft Windows за рахунок створення обгортки для існуючого Win32 API в керованому коді. Причому керований код класи, що реалізують API для Windows Forms, не залежать від мови розробки. Тобто програміст однаково може використовувати Windows Forms як із написанні ПЗ на C#, З++, і VB.Net, J# та інших.

### 3.3 Реалізація адаптивного тестування на основі моделі Раша

За допомогою моделі Раша можна на основі тестувань визначити рівні складностей тестових завдань. Як наслідок, питання, які були задіяні в тесті, можна розмістити на одній шкалі. Але для шкалування всієї сукупності питань із бази тестів не можна просто застосувати модель Раша. Адже для одного тесту здійснюється лише вибірка окремої кількості питань. Необхідною умовою для застосування моделі Раша є те, що всі вибрані завдання повинні бути запропоновані кожному із студентів. В результаті ми отримаємо щільну матрицю результатів. Для шкалування наступної (відмінної від попередньої) вибірки

питань на ту ж саму шкалу отримати логіти рівнів складності недостатньо, тому що при наступному тестуванні рівні знань студентів можуть відрізнятись і навіть склад групи дещо може різнитись від складу протестованих у попередньому тесті. Тому нами було запропоновано метод калібрування на основі трьох опорних завдань із попереднього тесту. Таким чином, у результаті скінченної кількості поточних тестувань можна звести на одну шкалу всі питання з бази. Кількість таких експериментальних тестувань залежить від вибірок. Процес застосування даного методу зображено на схемі 3.7

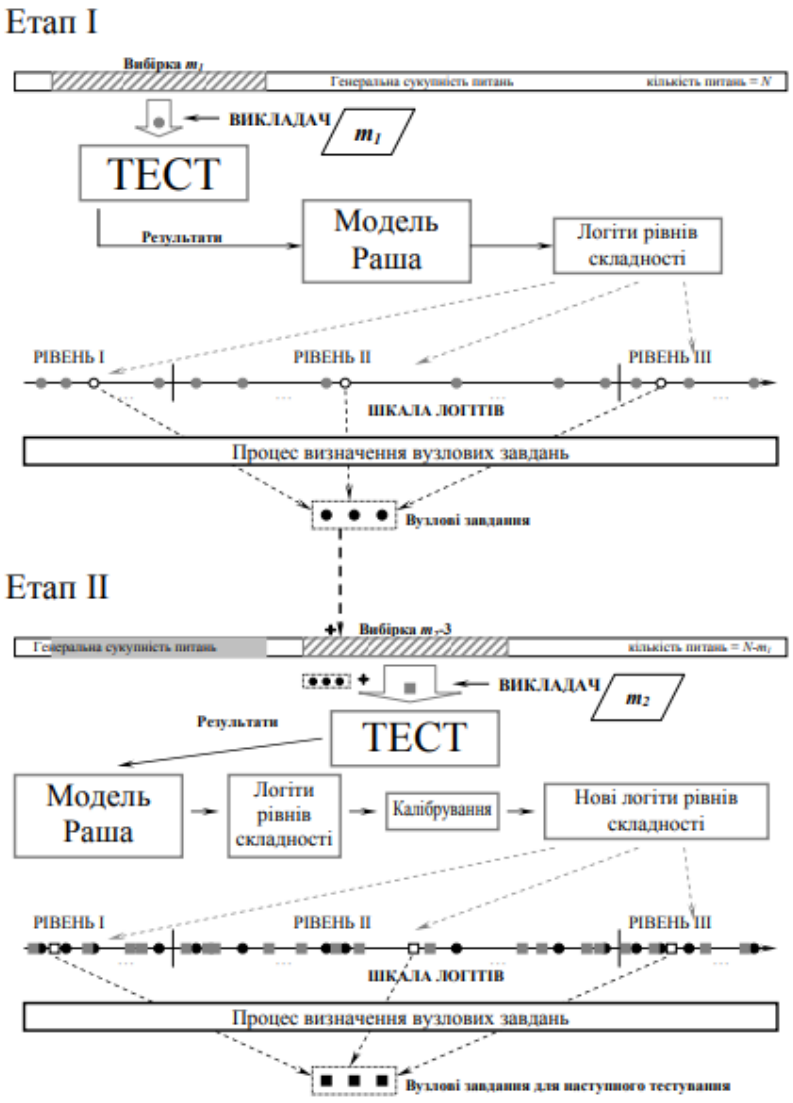



Рисунок 3.7 – Процес шкалування тестових завдань генеральної вибірки (I-II Етап)

Таким чином було сформульовано постановку задачі для практичної реалізації методу визначення міри складності тестового завдання на основі однопараметричної моделі Раша. Для реалізації цього методу було створено програмний модуль до існуючої системи дистанційного навчання за допомогою засобів PHP, який виконує контроль над подачею тестових питань, опрацьовує результати тестувань за допомогою методів, описаних вище, ставить у відповідність ідентифікатору питання значення логіта рівня складності. Це дозволяє розмістити питання на одній метричній шкалі. Алгоритм функціонування такого модуля можна подати у вигляді блок-схеми, яка зображена на рисунку 3.8

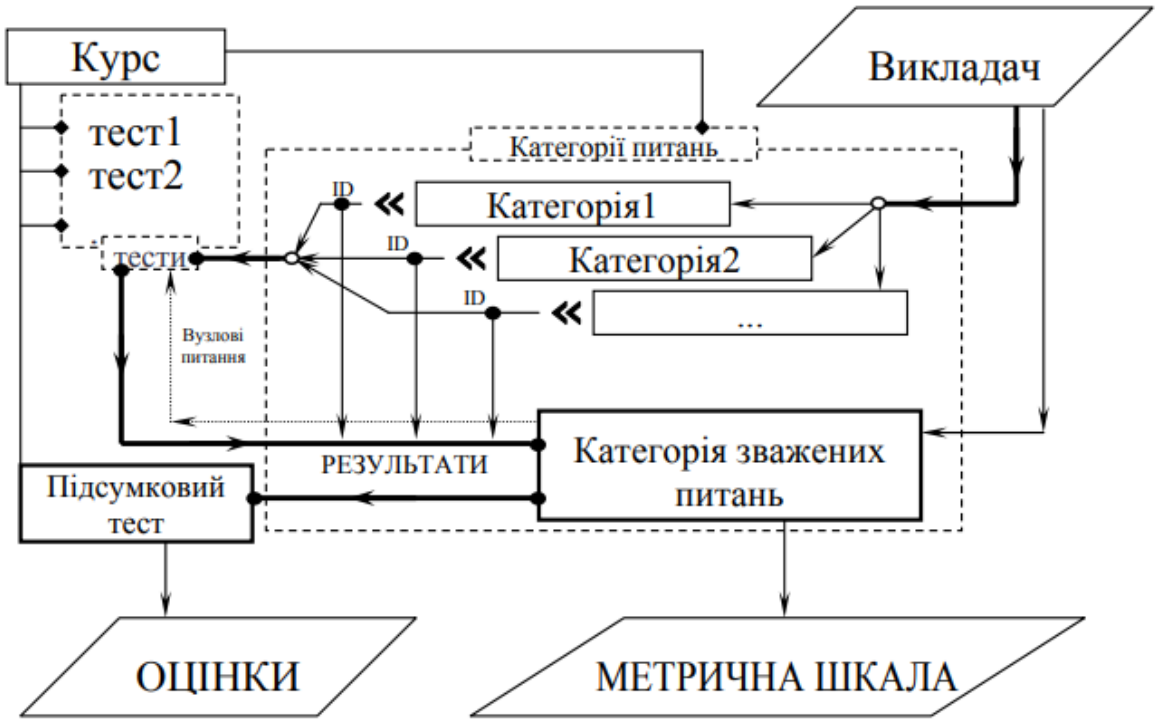



Рисунок 3.8 – Блок-схема програми

На перших стадіях функціонування дистанційного курсу формується перший тест для поточного тестування. Для тестового контролю в даному курсі

викладач формує категорії завдань, із яких будуть вибиратися завдання для тестувань. Для першого тесту викладач задає кількість випадково відібраних завдань з цих категорій . На рисунку видно, що для тесту відібрано два завдання із категорії 1 і набір з десяти випадкових завдань з цієї ж категорії. Після цього формується нова категорія «зважених» завдань, в котру ці завдання автоматично переносяться з наявних вже категорій. Відбувається організоване тестування. Після закінчення ми отримуємо щільну первинну матрицю результатів. Вона опрацьовується за допомогою моделі Раша. В результаті ми отримуємо масив логітів рівнів складності цих завдань.

При створенні наступних тестів викладач вибирає завдання із групи «незважених», що залишились в наявних категоріях. Таким чином, протягом поточних тестувань вичерпується вся генеральна сукупність тестових завдань, передбачених для даного курсу. Для того, щоб звести на одну шкалу логіти рівнів складності завдань, задіяних в другому тестуванні, перед тестуванням відбувається процес визначення вузлових завдань.

В даному експерименті це значення логітів, що набувають від’ємного, близького до нуля, додатного значень відповідно. Фактично, тепер ці завдання відповідають трьом рівням складності. За допомогою результатів відповідей на ці ж питання в другому тестуванні ми можемо визначити поправку для визначення логітів, які розмістяться на потрібній нам метричній шкалі. Процес вибору і переміщення завдань між категоріями аналогічний до попереднього. В експерименті нами було передбачено, що протягом поточних тестувань будуть зважені всі завдання з курсу і при підсумковому тестуванні можуть бути враховані ваги питань. Таким чином, ми вирішуємо проблему неадекватності визначення рівня знань студента. Маючи логіти, які відповідають завданням на метричній шкалі, такий метод надає можливість оцінити знання студента, враховуючи складність завдань, на які він відповідав(рис. 3.9).

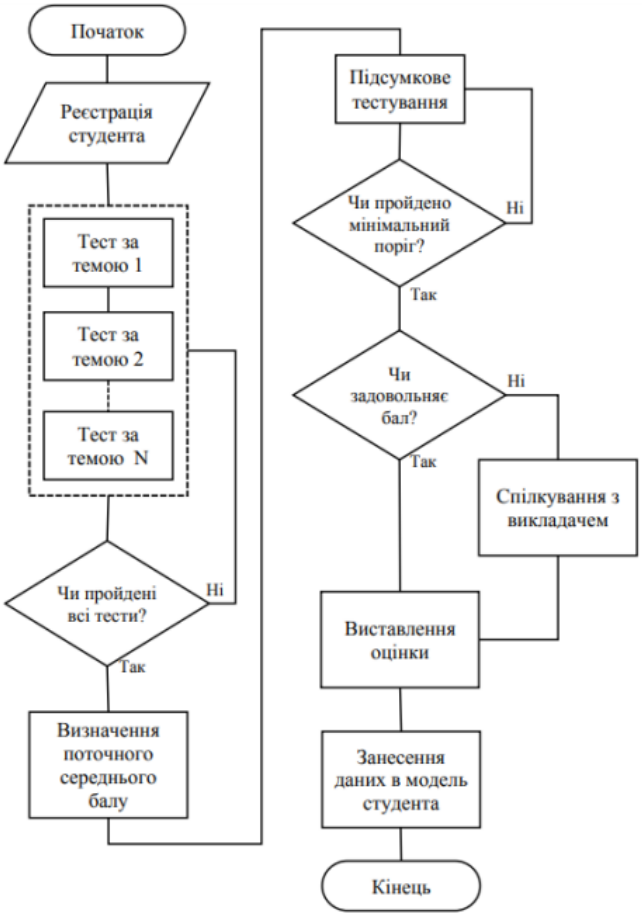



Рисунок 3.9 – Блок-схема покращеного алгоритму адаптивного тестування

В даному експерименті це значення логітів, що набувають від’ємного, близького до нуля, додатного значень відповідно. Фактично, тепер ці завдання відповідають трьом рівням складності. За допомогою результатів відповідей на ці ж питання в другому тестуванні ми можемо визначити поправку для визначення логітів, які розмістяться на потрібній нам метричній шкалі. Процес вибору і переміщення завдань між категоріями аналогічний до попереднього. В експерименті нами було передбачено, що протягом поточних тестувань будуть зважені всі завдання з курсу і при підсумковому тестуванні можуть бути враховані ваги питань. Таким чином, ми вирішуємо проблему неадекватності визначення рівня знань студента. Маючи логіти, які відповідають завданням на

метричній шкалі, такий метод надає можливість оцінити знання студента, враховуючи складність завдань, на які він відповідав.

Можна зробити узагальнення всіх переваг комп'ютерного адаптивного тестування:

1. КАТ уникає використання невідповідних питань. Питання, які занадто легкі або занадто важкі для учасників тесту, можуть викликати небажане поводження. Вони в значній мірі усунуті.

2. КАТ може бути коротшим від традиційного тестування.

3. КАТ може бути швидшим та інформативнішим. Неправильно вибране питання буде мати незначний вплив на тестові результати.

4. КАТ може бути чудовим засобом для правильного оцінювання учасника тесту і підбирання питань, відповідних до рівня знань.

5. КАТ дозволяє кожному учаснику тесту працювати індивідуально.

6. КАТ не потребує спеціального навчання учасників тесту.

7. КАТ пропонує багато варіантів для вибору індивідуального часу на відповідь конкретного питання.

8. КАТ зменшує в учасників тесту такі психологічні фактори, як втома і переживання.

9. КАТ може зменшити час тестування на більше ніж 50 % при підтримці того ж самого рівня надійності прийняття правильного рішення.

### 3.4 Тестування розробленого програмного модуля гейміфікації

Термін "навантажувальне тестування" використовується по-різному у професійній спільноті тестування програмного забезпечення. Під тестом навантаження зазвичай розуміється практика моделювання очікуваного

використання програми шляхом імітації одночасного доступу до програми декількох користувачів. Таким чином, це тестування найбільш актуальне для розрахованих на багато користувачів систем; часто побудованих з використанням моделі клієнт / сервер, веб-серверів у тому числі. Однак інші типи програмних систем можуть бути протестовані таким чином. Найбільш точне тестування навантаження імітує фактичне використання, на відміну від тестування з використанням теоретичного або аналітичного моделювання.

Навантаження тестування дозволяє виміряти якість обслуговування (QOS) вашого веб-сайту на основі фактичної поведінки клієнтів. Майже всі інструменти навантажувального тестування та фреймворки наслідують класичну парадигму навантажувального тестування: коли клієнти відвідують ваш веб-сайт, реєстратор сценаріїв записує спілкування, а потім створює відповідні сценарії взаємодії.

Генератор навантаження намагається відтворити записані сценарії, які можуть бути змінені з іншими параметрами тесту перед відтворенням. У процедурі відтворення, як апаратна, так і програмна статистика буде відстежуватися і збиратися провідником, ця статистика включає ЦП, пам'ять, дискове введення-виведення фізичних серверів, а також час відгуку, пропускну здатність тестованої системи (SUT) і т.д. нарешті, вся ця статистика буде проаналізована та буде сформовано звіт про навантажувальне тестування.

Тестування навантаження та продуктивності аналізує програмне забезпечення, призначене для розрахованої на багато користувачів аудиторії, піддаючи програмне забезпечення різній кількості віртуальних і живих користувачів, одночасно відстежуючи вимірювання продуктивності при цих різних навантаженнях. Тестування навантаження та продуктивності зазвичай проводиться в тестовому середовищі, ідентичному виробничому, перш ніж програмна система буде запущена.

Тестування стабільності – це тип дисфункції тестування програмного забезпечення, що виконується для вимірювання ефективності та здатності програмного додатка безперервно функціонувати протягом тривалого періоду

часу. Метою тестування стабільності є перевірка того, чи дає програмне забезпечення збій через нормальне використання будь-якої миті часу, перевіряючи його повний діапазон використання.

Тестування стабільності проводиться для перевірки ефективності розробленого продукту поза нормальними робочими можливостями, часто до критичної точки. Більш важливе значення має обробка помилок, надійність програмного забезпечення, надійність і масштабованість продукту за великого навантаження, ніж перевірка поведінки системи за умов.

Тестування стабільності оцінює проблеми стабільності. Це тестування в першу чергу, призначене для максимального навантаження на програмний компонент.

Було вирішено протестувати спеціальну навчальну програму-вікторину з гейміфікаціїї, яка дозволяє тренуватись в опануванні певних дисциплін для покращення їх засвоєння (рис. 3.10-3.13).


Рисунок 3.10 – Скріншот роботи вікторини

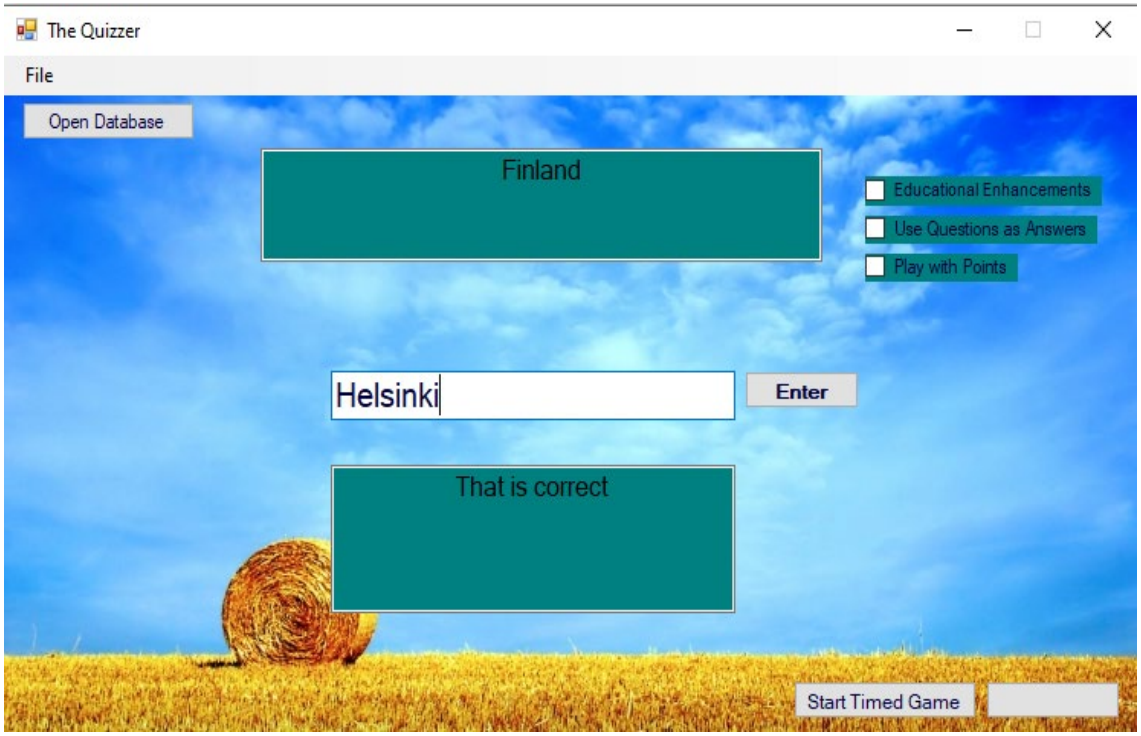



Рисунок 3.11 – Скріншот роботи вікторини (після завершення гри)

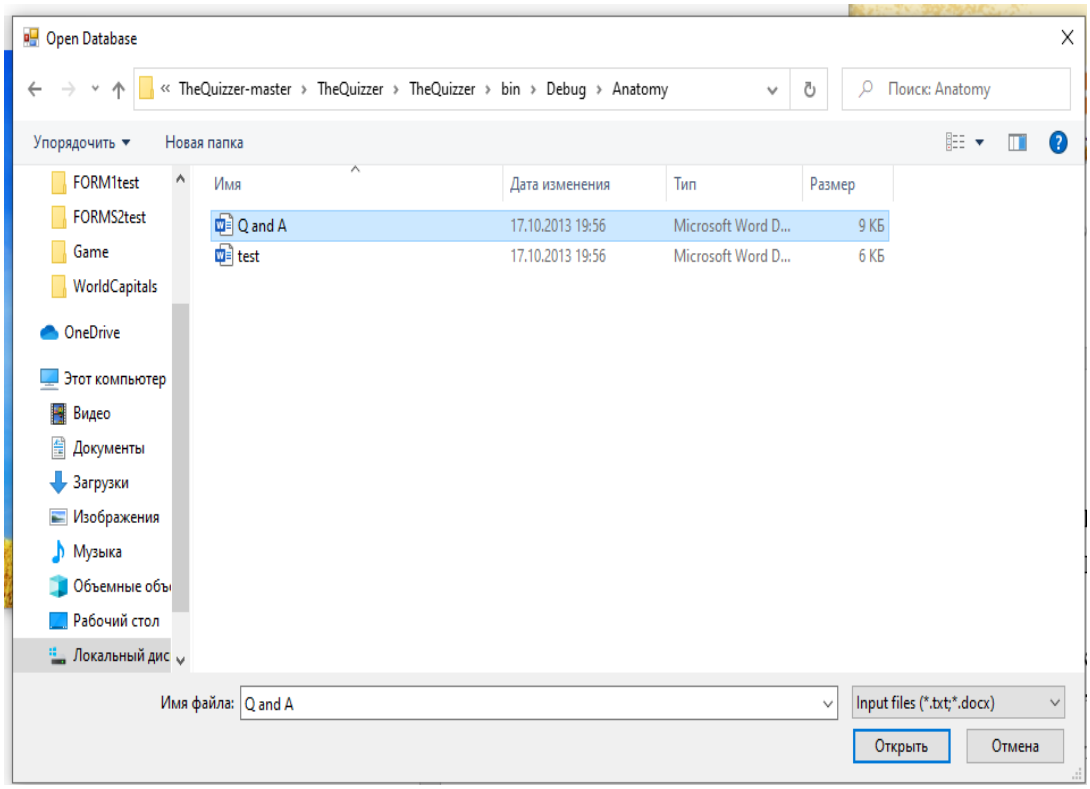



Рисунок 3.12 – Скріншот роботи вікторини (вибір в базі даних )

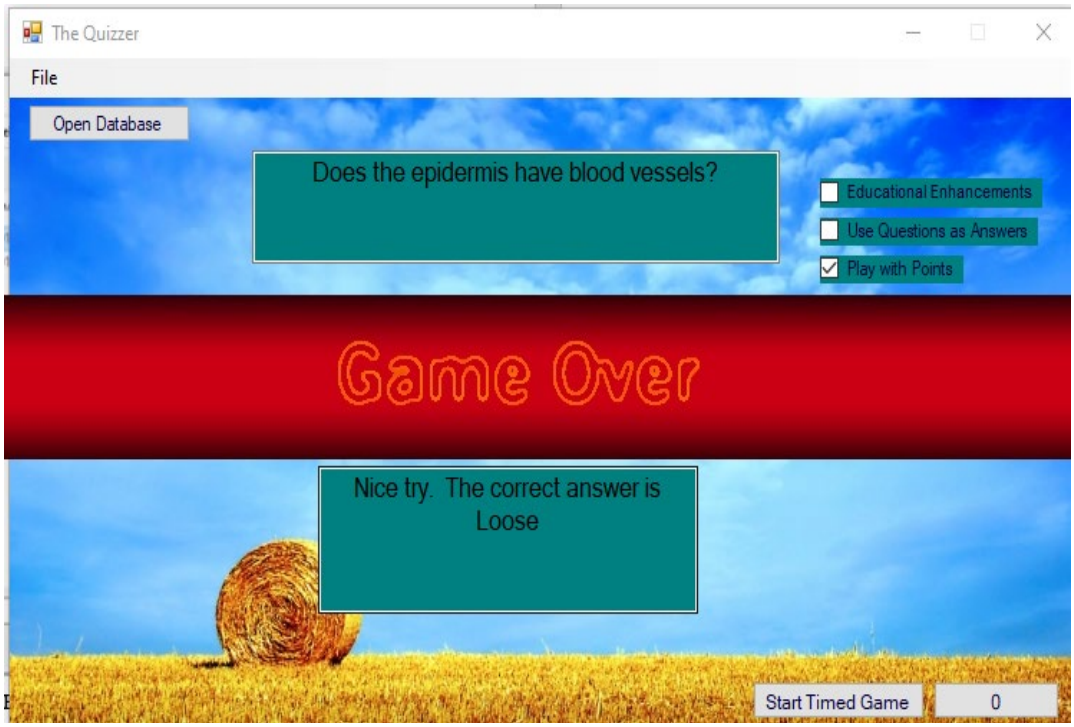



Рисунок 3.13 – Завершення вікторини при неправильній грі

Порівняно з системами-аналогами, дана система показала набагато меншу кількість багів та набагато якісніший рівень освіченості учасників після проходження вікторин.

## 3.4 Порівняльна характеристика ефективності розробленої системи геймифікації

Для візуалізації порівняльної характеристики розробленої системи геймифікації порівняно з аналогами було проведено колективну експертизу з залученням десяти експертів (студентів) та оцінюванням роботи з чотирма подібними системами, а саме – даною системою та трьома її аналогами. Варто зупинитися докладніше на методі колективної експертизи.

Методи колективної експертної оцінки засновані на виявленні узагальненої оцінки експертної групи шляхом аналізу та обробки індивідуальних незалежних оцінок експертів, що входять до складу групи.

В основі застосування методів колективної експертної оцінки лежить гіпотеза щодо наявності у експертів умінь оцінити з достатнім ступенем вірогідності: важливість і знання проблеми фактора, параметра, напряму розвитку, ознаки тощо; час здійснення тієї чи іншої події; значення параметрів, які прогнозуються; доцільність вибору одного з альтернативних шляхів розвитку об'єкта прогнозування і т. ін.

Методи колективної експертної оцінки за ознакою способу отримання інформації від експертів умовно можна розподілити на дві великі групи: методи групової експертизи і методи анкетування.

Найбільш часто використовуються такі методи групової експертизи: експертних комісій, колективної генерації ідей (метод "мозкової атаки"), оперативних ігор.

Інший поширений метод колективної експертної оцінки – метод експертних комісій, коли організація роботи експертів будується за принципом круглого столу. Під час спільної роботи експерти пропонують свої варіанти прогнозу, разом обґрунтовуючи і обговорюючи їх. По можливості, вони намагаються виробити єдину точку зору відносно прогнозу. Проте дошкульним місцем цього методу є те, що в основі вироблення єдиної думки лежить логіка компромісу, котра може привести до надмірного впливу окремих експертів (через їхній авторитет) на міркування решти членів експертної групи.

Метод генерації ідей ("мозкових атак") заснований на стимулюванні творчої діяльності експертів шляхом сумісного обговорення конкретної проблеми. При цьому обговорення регламентується певними правилами: забороняється оцінка ідей, що висуваються; обмежується час одного виступу; припускаються багаторазові виступи кожного експерта; пріоритет щодо виступу має експерт, який розвиває попередню ідею; обов'язково фіксуються всі ідеї; оцінка ідей, що висунуті, здійснюється на наступних етапах.

Відомі три різновиди методу колективної генерації ідей ("мозкової атаки"): метод прямої групової експертизи ("пряма мозкова атака "), метод "групової згоди" і метод "оперативної творчості".

Метод прямої групової експертизи заснований на гіпотезі про те, що серед ідей, запропонованих експертами, є принаймні декілька добрих. Порядок проведення прямої групової експертизи такий: перед групою експертів різних спеціальностей ставиться проблема, яка може мати декілька варіантів рішень. Мета методу полягає в збиранні ідей щодо поставленої проблеми. Потім в декілька етапів проводиться їх аналіз, відбираються найбільш раціональні і виключаються непридатні. Найбільш часто цей метод застосовується на рівні зовнішньополітичних і стратегічних рішень, а також при воєнно-економічному і воєнно-технічному прогнозуванні.

Метод "групової згоди" заснований на проведенні групової експертизи з метою визначення згоди й єдності поглядів щодо прогнозованого питання. У цьому процесі беруть участь, як правило, шість експертів. Встановлено, що збільшення кількості експертів більше шести веде до збільшення часу та ускладнює одержання згоди, не збільшуючи суттєво кількості запропонованих ідей. При зменшенні кількості експертів можливо, що будуть враховані не всі обставини та знижений ступінь точності прогнозу.

Було обрано метод прямої колективної експертизи для оцінки роботи додатків та її результати оформлено у таблиці 3.2 та на рис.3.11.

Таблиця 3.2 – Результати колективної оцінки систем геймифікації

| | Середня оцінка за зручністю роботи | Середня оцінка за відсутністю багів | Середня оцінка за ефективністю вивчення дисциплін | Середня оцінка за графічним інтерфейсом користувача |
|---|---|---|---|---|
| Розроблена система | 4,8 | 4,76 | 4,92 | 4,87 |
| Аналог 1 | 4,77 | 4,78 | 4,9 | 4,8 |
| Аналог 2 | 4,5 | 4,7 | 4,83 | 4,83 |
| Аналог 3 | 4,79 | 4,71 | 4,9 | 4,56 |

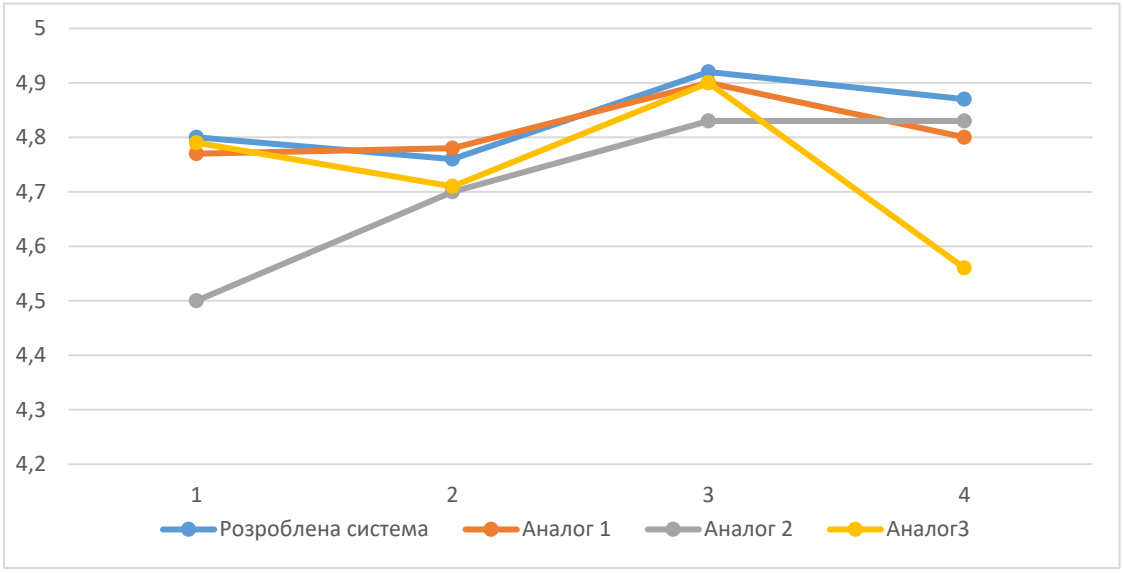



Рисунок 3.11 – Порівняльна характеристика систем геймифікації

Отже, можна зробити висновок, що розроблена система геймифікації є значно ефективнішою за аналоги.

### 3.5 Проведення групової експертизи щодо успішності впровадження нейромережевої складової за допомогою методу Дельфі

Метод Дельфі - це колективний експертний метод, який використовується для збору та узгодження експертних оцінок, думок та прогнозів з метою прийняття важливих рішень. Цей метод базується на взаємодії групи експертів, які виступають анонімно та незалежно один від одного.

У методі Дельфі зазвичай існує модератор, який керує процесом та координує взаємодію експертів. Початково, експерти отримують питання або проблему, про яку потрібно прийняти рішення або дати прогноз. Кожен експерт надає свою відповідь або оцінку. Після цього, відповіді анонімізуються та агрегуються. Агреговані результати надсилаються експертам для повторного рецензування і перегляду.

Процес повторюється декілька разів, з попередніми результатами враховують під час наступних раундів. Мета - досягнути згоди та зближення

експертних думок. Цей метод дозволяє вирішувати складні проблеми, які потребують оцінки від експертів з різних областей знань.

Метод Дельфі ефективно використовується в ситуаціях невизначеності, коли інформація обмежена або коли потрібно об'єктивно оцінити можливі ризики чи перспективи. Він застосовується в різних галузях, таких як наукові дослідження, прогнозування ринкових трендів, стратегічне планування та управління проектами. Метод Дельфі дозволяє зменшити вплив індивідуальних біасів, забезпечує більш об'єктивні результати та сприяє зростанню якості прийнятих рішень.

Оцінювати якість нейромережевого модуля будемо за єдиним впровадженим нейромережевим параметром – успішність генерації вітань в кінці вікторини. Оцінювання здійснювалось 10 експертами по розробленій системі і трьом аналогам. Результат показав, що розроблена система має найвищу групову оцінку порівняно з аналогами у 5,792 бали (табл. 3.3).

Таблиця 3.3 – Визначення якості розробки нейромережевого модуля додатку

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 6,2 | 5,9 | 0,7 | 5,8 | 2,7 | 1,2 | 7 | 3,6 | 3,9 | 6 | 4,631 |
| 0,8 | 5,6 | 3,6 | 1,9 | 2,7 | 1,1 | 6,2 | 4 | 3,7 | 7 | 3,963 |
| 7,9 | 1,1 | 1,1 | 8,3 | 2,8 | 2,3 | 1,9 | 9,2 | 9,9 | 3,3 | 5,343 |
| 2 | 2,8 | 4,9 | 1,4 | 5,7 | 6 | 7,2 | 9,5 | 4,5 | 9,3 | 5,792 |

## 3.6 Висновок до розділу 3

Здійснено вибір мови програмування, після проведення порівняльного аналізу трьох мов об`єктно-орієнтованого програмування, що всі відповідають вимогам до виконання роботи. Обрано оптимальне середовище розробки системи, після детального дослідження всіх його можливостей та переваг, що залучені чи прямо використовуються у створенні необхідного додатку. Описано усі базові принципи розробки гейміфікованого додатку, створено прототип, що відповідає поставленим вимогам, протестовано його можливості.

# 4 ЕКОНОМІЧНА ЧАСТИНА

Економічна частина є завершальним розділом магістерської кваліфікаційної роботи, в якому розробляються остаточні висновки щодо економічної ефективності запропонованої розробки. В даному розділі розглянемо основні питання конкурентоспроможності продукту та комерційного потенціалу розробки.

Магістерська кваліфікаційна робота на тему «Інформаційна технологія гейміфікації процесц тестування під час навчання» відноситься до науково-технічних робіт, які орієнтовані на виведення на ринок (або рішення про виведення науково-технічної розробки на ринок може бути прийнято у процесі проведення самої роботи), тобто коли відбувається так звана комерціалізація науково-технічної розробки. Але для цього потрібно знайти потенційного інвестора, який би взявся за реалізацію цього проекту і переконати його в економічній доцільності такого кроку.

Для наведеного випадку нами мають бути виконані такі етапи робіт:

1) проведено комерційний аудит науково-технічної розробки, тобто встановлення її науково-технічного рівня та комерційного потенціалу;

2) розраховано витрати на здійснення науково-технічної розробки;

3) розрахована економічна ефективність науково-технічної розробки у випадку її впровадження і комерціалізації потенційним інвестором і проведено обґрунтування економічної доцільності комерціалізації потенційним інвестором.

## 4.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту є оцінювання науково-технічного рівня та рівня комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності, тобто під час виконання магістерської кваліфікаційної роботи.

Для проведення комерційного і технологічного аудиту залучимо 3-х незалежних експертів. У нашому випадку такими експертами будуть провідні викладачі випускової та споріднених кафедр.

Результати оцінювання комерційного потенціалу розробки заносимо до таблиці 4.1.

Таблиця 4.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

| Бали (за 5-ти бальною шкалою) | | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| Технічна здійсненність концепції | | | | | |
| 1 | Достовірність концепції не підтверджена | Концепція підтверджена експертними висновками | Концепція підтверджена розрахунками | Концепція перевірена на практиці | Перевірено працездатність продукту в реальних умовах |
| Ринкові переваги (недоліки) | | | | | |
| 2 | Багато аналогів на малому ринку | Мало аналогів на малому ринку | Кілька аналогів на великому ринку | Один аналог на великому ринку | Продукт не має аналогів на великому ринку |
| 3 | Ціна продукту значно вища за ціни аналогів | Ціна продукту дещо вища за ціни аналогів | Ціна продукту приблизно дорівнює цінам аналогів | Ціна продукту дещо нижче за ціни аналогів | Ціна продукту значно нижче за ціни аналогів |
| 4 | Технічні та споживчі властивості продукту значно гірші, ніж в аналогів | Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів | Технічні та споживчі властивості продукту на рівні аналогів | Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів | Технічні та споживчі властивості продукту значно кращі, ніж в аналогів |
| 5 | Експлуатаційні витрати значно вищі, ніж в аналогів | Експлуатаційні витрати дещо вищі, ніж в аналогів | Експлуатаційні витрати на рівні експлуатаційних витрат аналогів | Експлуатаційні витрати трохи нижчі, ніж в аналогів | Експлуатаційні витрати значно нижчі, ніж в аналогів |

Продовження таблиці 4.1

| | | | | | |
|---|---|---|---|---|---|
| Ринкові перспективи | | | | | |
| 6 | Ринок малий і не має позитивної динаміки | Ринок малий, але має позитивну динаміку | Середній ринок з позитивною динамікою | Великий стабільний ринок | Великий ринок з позитивною динамікою |
| 7 | Активна конкуренція великих компаній на ринку | Активна конкуренція | Помірна конкуренція | Незначна конкуренція | Конкурентів немає |
| Практична здійсненність | | | | | |
| 8 | Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї | Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців | Необхідне незначне навчання фахівців та збільшення їх штату | Необхідне незначне навчання фахівців | Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї |
| 9 | Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні | Потрібні незначні фінансові ресурси. Джерела фінансування відсутні | Потрібні значні фінансові ресурси. Джерела фінансування є | Потрібні незначні фінансові ресурси. Джерела фінансування є | Не потребує додаткового фінансування |
| 10 | Необхідна розробка нових матеріалів | Потрібні матеріали, що використовуються у військово промисловому комплексі | Потрібні дорогі матеріали | Потрібні досяжні та дешеві матеріали | Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві |
| 11 | Термін реалізації ідеї більший за 10 років | Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років | Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років | Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років | Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років |
| 12 | Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту | Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу | Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу | Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту | Відсутні будь- які регламентні обмеження на виробництво та реалізацію продукту |

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці.

Таблиця 4.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки

| Критерії | Експерти | | |
|---|---|---|---|
| | Експерт 1 | Експерт 2 | Експерт 3 |
| | Бали, виставлені експертами | | |
| Технічна здійсненність концепції | 3 | 2 | 3 |
| Ринкові переваги (наявність аналогів) | 3 | 3 | 4 |
| Ринкові переваги (ціна продукту) | 4 | 2 | 2 |
| Ринкові переваги (технічні властивості) | 3 | 3 | 3 |
| Ринкові переваги (експлуатаційні витрати) | 2 | 4 | 3 |
| Ринкові перспективи (розмір ринку) | 3 | 3 | 4 |
| Ринкові перспективи (конкуренція) | 4 | 2 | 3 |
| Практична здійсненність (наявність фахівців) | 3 | 2 | 2 |
| Практична здійсненність (наявність фінансів) | 2 | 2 | 3 |
| Практична здійсненність (необхідність нових матеріалів) | 3 | 3 | 3 |
| Практична здійсненність (термін реалізації) | 4 | 4 | 3 |
| Практична здійсненність (розробка документів) | 3 | 3 | 2 |
| Сума балів | 37 | 33 | 35 |
| Середньоарифметична сума балів $\overline{СБ}$ | 35 | | |

За результатами розрахунків, наведених в таблиці 4.2, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в табл. 4.3

Таблиця 4.3 – Науково-технічні рівні та комерційні потенціали розробки

| Середньоарифметична сума балів, розрахована на основі висновків. | Рівень комерційного потенціалу розробки |
|---|---|
| 0 – 10 | Низький |
| 11 – 20 | Нижче середнього |
| 21 – 30 | Середній |
| 31 – 40 | Вище середнього |
| 41 – 50 | Високий |

Оскільки середньоарифметична сума балів складає 35, то рівень комерційного потенціалу розробки вище середнього, тому дана розробка є реальною для подальшої її реалізації та впровадження.

Можливі декілька шляхів реалізації розробки.

### 4.1.1 Розрахунок витрат на здійснення науково-дослідної роботи

Витрати, пов’язані з проведенням науково-дослідної роботи на тему «Інформаційна технологія гейміфікації процесу тестування під час навчання», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

### 4.1.2 Витрати на оплату праці

*Основна заробітна плата дослідників*

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Витрати на основну заробітну плату дослідників розраховують відповідно до посадових окладів працівників, за формулою:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \quad (4.1)$$

де $k$ – кількість посад дослідників залучених до процесу досліджень;

$M_{ni}$ – місячний посадовий оклад конкретного дослідника, грн;

$t_i$ – число днів роботи конкретного дослідника, дн.;

$T_p$ – середнє число робочих днів в місяці, $T_p$=21 дні.

Зведемо сумарні розрахунки до таблиці 4.4.

Таблиця 4.4 – Витрати на заробітну плату дослідників

| Найменування посади | Місячний посадовий оклад, грн | Оплата за робочий день, грн | Число днів роботи | Витрати на заробітну плату, грн |
|---|---|---|---|---|
| Керівник проекту | 35000 | 1750 | 6 | 10500 |
| Розробник | 30000 | 1500 | 25 | 37500 |
| Всього | | | | 48000 |

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (4.2)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати.

На даному підприємстві додаткова заробітна плата начисляється в розмірі 11% від основної заробітної плати.

$$З_{дод}\ 0{,}1 \cdot (З_о + З_р) = 0{,}1 \cdot (48000 + 0) = 4800 грн. \quad (4.2.1)$$

**4.1.3 Відрахування на соціальні заходи**

Нарахування на заробітну плату $Н_{зп}$ дослідників та робітників, які брали участь у виконанні даного етапу роботи, розраховуються за формулою :

$$З_{дод} = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%}, \quad (4.3)$$

де $Н_{зп}$ – норма нарахування на заробітну плату.

Дана діяльність відноситься до бюджетної сфери, тому ставка єдиного внеску на загальнообов'язкове державне соціальне страхування буде складати 22%, тоді:

$$Зн = (48000+0+4800) \cdot 22 / 100\% = 11616\ грн \quad (4.3.1)$$

### 4.1.4 Амортизація обладнання, програмних засобів та приміщень

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12} \quad (4.4)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

Таблиця 4.5 – Амортизаційні відрахування по кожному виду обладнання

| Найменування | Балансова вартість, грн. | Строк корисного використання, років | Термін використання, місяців | Сума амортизації, грн. |
|---|---|---|---|---|
| Ноутбук | 34000 | 5 | 1,2 | 680 |
| Всього | 680 | | | |

### 4.1.5 Паливо та енергія для науково-виробничих цілей

Витрати на силову електроенергію ($B_e$) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i} \quad (4.5)$$

де $W_{yi}$ – встановлена потужність обладнання на визначеному етапі розробки, кВт;

$t_i$ – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_e$ = 7,50 грн;

$K_{впі}$ – коефіцієнт, що враховує використання потужності, $K_{впі} < 1$;

$\eta_i$ – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

Таблиця 4.6 – Витрати на електроенергію

| Найменування обладнання | Потужність, кВт | Тривалість годин роботи |
|---|---|---|
| Ноутбук | 0,13 | 170 |

$$Ве = \sum \frac{W_i \cdot t_i \cdot Ц_е \cdot К_{впі}}{ККД} = \frac{0{,}13 \cdot 170 \cdot 7{,}5 \cdot 0{,}85}{0{,}88} = 167 \text{ грн.} \quad (4.5.1)$$

### 4.1.6 Сировина та матеріали

Витрати за статтею «Сировина та матеріали» відсутні.

### 4.1.7 Програмне забезпечення розробки

Витрати за статтею «Програмне забезпчення» відсутні.

### 4.1.8 Службові відрядження

Витрати за статтею «лужбові відрядження» відсутні.

### 4.1.9 Робота сторонніх організацій

Витрати за статтею «Робота сторонніх організацій» відсутні.

### 4.1.6 Інші витрати

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%} \quad (4.6)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів} = 90\%$.

$$Ів = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%} = (48000 + 0) \cdot \frac{90}{100} = 43200 \text{ грн.}, \quad (4.6.1)$$

### 4.1.7 Накладні (загальновиробничі) витрати

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов’язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов’язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов’язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$В_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%} \quad (4.7)$$

де $H_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати».

$$\text{Внзв} = \left(З_о + З_р\right) \cdot \frac{Н_{нзв}}{100\%} = (48000 + 0) \cdot \frac{150}{100} = 72000 \text{ грн.}, \quad (4.7.1)$$

Витрати на проведення науково-дослідної роботи на тему «Інформаційна технологія гейміфікації процесу тестування під час навчання» розраховуємо як суму всіх попередніх статей витрат за формулою:

$$В_{заг} = З_о + З_р + З_{дод} + З_н + А_{обл} + В_е +$$

$$+ I_в + В_{нзв} = 48000+0+4800+11616+680+167+43200+72000=180463 \text{ грн.} \quad (4.7.2)$$

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{B_{заг}}{\eta} \quad (4.8)$$

де $\eta$ - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо $\eta$=0,5.

$$\text{ЗВ} = \frac{В_{заг}}{\eta} = \frac{180463}{0,5} = 360926 \text{ грн.}, \quad (4.8.2)$$

## 4.2 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

Розрахуємо можливе збільшення чистого прибутку у потенційного інвестора для кожного із років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки за формулою:

$$\Delta П_i = (\pm \Delta Ц_0 \cdot N + Ц_0 \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot \left(1 - \frac{\vartheta}{100}\right), \quad (4.9)$$

де $\pm \Delta Ц_0$ – зміна основного якісного показника від впровадження результатів науково-технічної розробки в аналізованому році;

$N$ – основний кількісний показник, який визначає величину попиту на аналогічні чи подібні розробки у році до впровадження результатів нової науково-технічної діяльності;

$Ц_0$ – основний якісний показник, який визначає ціну реалізації нової науково-технічної розробки в аналізованому році;

$\Delta N$ – зміна основного кількісного показника від впровадження результатів науково-технічної розробки в аналізованому році;

$\lambda$ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість;

$\rho$ – коефіцієнт, який враховує рентабельність інноваційного продукту (послуги), рекомендується приймати 0,2...0,5;

$\vartheta$ – ставка податку на прибуток.

Ставка податку на додану вартість в 2023 році залишилась на рівні 20% , а коефіцієнт λ=0,8333. Ставка податку на прибуток складає 18%.

Очікуваний термін життєвого циклу розробки 1 рік, тому:

$$\Delta П = ((3900 - 3500) \cdot 7500 - (7500 - 7500) \cdot 3500) \cdot 0{,}8333 \cdot 0{,}3 \cdot \left(1 - \frac{18}{100}\right) = 667473 \text{ грн.}, \quad (4.9.1)$$

Далі розрахуємо приведену вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \quad (4.10)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

*Т* – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

$\tau$ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні;

*t* – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t} = \frac{667473}{(1+0,1)^1} = 606794 \text{ грн.}, \quad (4.10.1)$$

Далі розрахуємо величину початкових інвестицій, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки. Для цього можна використати формулу:

$$PV = k_{інв} \cdot ЗВ, \quad (4.11)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науко-во-технічної розробки та її комерціалізацію.

*ЗВ* – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, грн.

$$PV = k_{інв} \cdot ЗВ = 1,5 \cdot 360926 = 541389 \text{ грн.} \quad (4.11.1)$$

Тоді абсолютний економічний ефект або чистий приведений дохід для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$Е_{абс} = ПП - PV, \quad (4.12)$$

де *ПП* – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, грн;
*PV* – теперішня вартість початкових інвестицій, грн.

$$\text{Е}_{\text{абс}} = \text{ПП} - PV = 606794 - 541389 \ = 65405 \text{ грн.} \qquad (4.12.1)$$

Внутрішня економічна дохідність інвестицій , які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки, розраховується за формулою:

$$\text{Е}_{\text{в}} = \sqrt[\text{Т}_{\text{ж}}]{1 + \frac{\text{Е}_{\text{абс}}}{PV}}, \qquad (4.12)$$

де $\text{Е}_{\text{абс}}$ – абсолютний економічний ефект вкладених інвестицій, грн;
*PV* – теперішня вартість початкових інвестицій, грн;
$\text{Т}_{\text{ж}}$– життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, роки.

$$\text{Е}_{\text{в}} = \sqrt[\text{Т}_{\text{ж}}]{1 + \frac{\text{Еабс}}{PV}} = \sqrt[1]{1 + \frac{65405}{541389}} = 1{,}058. \qquad (4.12.1)$$

Далі визначимо бар’єрну ставку дисконтування, тобто мінімальну внутрішню економічну дохідність інвестицій, нижче якої кошти у впровадження науково-технічної розробки та її комерціалізацію вкладатися не будуть.
Мінімальна внутрішня економічна дохідність вкладених інвестицій визначається за формулою:

$$\tau_{min} = d + f, \qquad (4.13)$$

де $d$ – середньозважена ставка за депозитними операціями в комерційних банках;
$f$ – показник, що характеризує ризикованість вкладення інвестицій.

$$\tau_{\text{мін}} = d + f = 0{,}15 + 0{,}3 = 0{,}45 \quad (4.13.1)$$

$\tau_{\text{мін}}$ =0.45<1.058, що свідчить про те, що внутрішня економічна дохідність інвестицій, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково-дослідну роботу доцільно.

Далі розрахуємо період окупності інвестицій, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{\text{Е}_\text{в}}, \quad (4.14)$$

де $\text{Е}_\text{в}$ – внутрішня економічна дохідність вкладених інвестицій.

$$\text{Т}_\text{о} = \frac{1}{\text{Е}_\text{в}} = \frac{1}{1{,}058} = 0{,}95 \text{ року.} \quad (4.14.1)$$

Термін окупності складає 1 рік, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження цієї розробки та виведення її на ринок.

### 4.3 Висновок до розділу 4

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Інформаційна технологія гейміфікації процесу тестування під час навчання» становить 45 балів, що, свідчить про комерційну важливість

проведення даних досліджень (рівень комерційного потенціалу розробки високий).

Також термін окупності становить 0.95 р., що менше 2-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок

# ВИСНОВКИ

Під час написання магістерської кваліфікаційної роботи було підтверджено актуальність предметної області з експертного оцінювання, виявлено мету, предмет, об’єкт та основні задачі дослідження.

Було обгрунтовано доцільність створення додатку гейміфікації процесу тестування під час навчання.Розглянуто та проаналізовано предметну область систем гейміфікації тестування, досліджено їх основні особливості та принципи. Проаналізовано ряд програм-аналогів, які забезпечують навчання у процесі гри, містять рейтинги та призи, а також заохочують та мотивують до вивчення. У результаті аналізу виявлено ряд недоліків подібних систем, серед них: недостатня універсальність додатку, необхідність обов’язкової авторизації, загальна складність інтерфейсу. Здійснено постановку задачі дослідження та сформульовано основні вимоги до розроблюваного програмного забезпечення для гейміфікації вивчення різних дисциплін.

Спроектовано простий алгоритм роботи додатку гейміфікації процесу тестування, що враховує всі необхідні елементи, які мають бути залучені до навчального процесу, а також окреслено ряд методик, що задовольняють всі необхідні стандарти навчання, для проведення тестування майбутнього додатку,який буде створено згідно зі всіми поставленими вимогами та деталями до технічного завдання. Побудовано декілька наочних UML-діаграм, що відображають елементи та можливості майбутнього додатку, для кращого розуміння структури програми та візуалізації потреб користувача, який буде брати участь у навчальному процесі з використанням технології геймифікації та додатку, що ці технології вводить в навчальний процес.

Описано принцип роботи алгоритму адаптивного тестування та моделі Раша. Вдосконалено алгоритм за рахунок нового модуля.

Здійснено вибір мови програмування, Проведено аналіз сучасних об’єктно-орієнтованих мов програмування, та обрано мову C# та середовище Visual Studio 2017 для проектування додатку. Протестовано різні режими роботи і підтверджено його коректність.. Обрано оптимальне середовище розробки системи, після детального дослідження всіх його можливостей та переваг, що залучені чи прямо використовуються у створенні необхідного додатку. Описано усі базові принципи розробки гейміфікованого додатку, створено прототип, що відповідає поставленим вимогам, протестовано його можливості та оцінено результати, отримані дані збігаються з вимогами до проекту, чим доведено його коректність. візуалізована порівняльна характеристика ефективності розробленої системи гейміфікації. Проведена експертиза щодо успішного впровадження нейромережевої складової на онові методі Делфі та методі Раша.

Проаналізовано віхи подальшого дослідження та сформовано задачі на удосконалення інтелектуальної технології гейміфікації процесу тестування під час навчання.

Статистика тестування показала, що розроблена програма має значний успіх в порівнянні з аналогами.

Було проведено дослідження, в якому комерційний потенціал розробки інформаційної технології гейміфікації процесу тестування оцінюється на високому рівні, досягаючи 44,3 бала. Технічно розробка виявилася значно переважною над існуючими аналогами, маючи узагальнений коефіцієнт якості приблизно в 3,55 рази. Термін окупності цієї розробки становить 1,56 р., що відповідає вимогам комерційної привабливості і може зацікавити потенційних інвесторів.

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. А. І. Вдовиченко. Підхід щодо гейміфікації процесу тестування під час навчання./ Вдовиченко А. І., Арсенюк І. Р.// Молодь в науці: дослідження, проблеми, перспективи (МН-2024), 15.10.23 – 10.05.24 : збірник матеріалів. – Вінниця: ВНТУ, 2024. – Режим доступу: https://conferences.vntu.edu.ua/index.php/mn/mn2024/paper/viewFile/19808/16407
2. Геймифікація для мотивації – [Електронний ресурс] Режим доступу https://indigo.co.ua/ua/blog/geymifikaciya-dlya-motivacii-kogda-missiya-vypolnima
3. Геймифікація освітнього процесу – [Електронний ресурс] Режим доступу https://cyberleninka.ru/article/n/geymifikatsiya-osvitnogo-protsesu-pid-chas-vivchennya-distsiplin-prirodnicho-matematichnogo-tsiklu-uchnyami-zzso
4. Duolingo – [Електронний ресурс] Режим доступу https://englishtest.duolingo.com/ru
5. Goalbookapp – [Електронний ресурс] Режим доступу https://goalbookapp.com/
6. Coursehero – [Електронний ресурс] Режим доступу https://www.coursehero.com/
7. Methodology_for_developing – [Електронний ресурс] Режим доступу https://www.researchgate.net/publication/323818035_Methodology_for_devel

oping_gamification-based_learning_programming_language_framework

8. gamification – [Електронний ресурс] Режим доступу https://www.valamis.com/hub/gamification
9. Геймифікація – [Електронний ресурс] Режим доступу https://journals.sagepub.com/doi/pdf/10.1177/1046878114563660
10. Using-Gamification – [Електронний ресурс] Режим доступу https://www.cognizant.com/InsightsWhitepapers/Using-Gamification-to-Build-a-Passionate-and-Quality-Driven-Software-Development-Team.pdf
11. Top learning gamification companies – [Електронний ресурс] Режим доступу https://elearningindustry.com/top-elearning-gamification-companies
12. Interaction design [Електронний ресурс] Режим доступу https://www.interaction-design.org/literature/topics/gamification
13. what-is-gamification – [Електронний ресурс] Режим доступу https://www.biworldwide.com/gamification/what-is-gamification/
14. Benefits of learning gamification – [Електронний ресурс] Режим доступу https://insights.learnlight.com/en/articles/5-benefits-of-gamification-in-learning/
15. Pointagram – [Електронний ресурс] Режим доступу https://www.pointagram.com/?gclid=Cj0KCQiA3NX_BRDQARIsALA3fIIoXGzf-yW57wlxTgvb6qHEfFsWxxLl2sLeJD-20d79pxqwOfsy5KcaAj1bEALw_wcB
16. 10 successful examples of gamification[Електронний ресурс] Режим доступу https://potion.social/en/blog/10-amazingly-successful-examples-of-gamification/
17. The Multiplayer Classroom: Designing Coursework as a Game, 1st Edition by Lee Sheldon
18. Karl M. Kapp - The Gamification of Learning and Instruction: Game-based Methods and Strategies for Training and Education, 1st Edition
19. Jane McGonigal - Reality Is Broken: Why Games Make Us Better and How They Can Change the World, Reprint Edition

20. Федорук П.І. Адаптивні тести: статистичні методи аналізу результатів тестового контролю знань // Математичні машини і системи. – 2007. – № 3,4. – С. 122-138.
21. Федорук П.І. Моделі і методи діагностики знань з використанням адаптивних тестів // УСиМ. – 2007. – № 5. – С. 68-76.
22.. У чому відмінність веб-додатків від мобільних додатків [Електронний ресурс]. Режим доступу – https://brander.ua/blog/u-chomu-vidminnist-vebdodatkivvid-mobilnykh-dodatkiv. Computerized Adaptive Testing: Theory and Practice. – Dordrecht, The Netherlands: Kluwer, 2000. – 323 p.
23. Вивчення адаптивного тестування// Пишний В. А. URL:https://la.kpi.ua/itstream/12345789/53738/1/Pyshnyi
24. Інструменти адаптивного навчання в CMS UKU// URL:https://ceit-blog.ucu.edu.ua/ed-tech/adaptyvni-instrumenty-navchannya-v-cms-ucu/

# ДОДАТКИ

## Додаток А (обов'язковий)
## Протокол перевірки МКР на наявність текстових запозичень

Назва роботи: Інформаційна технологія гейміфікації процесу тестування під час навчання

Тип роботи: магістерська кваліфікаційна робота

(БДР, МКР)

Підрозділ кафедра комп'ютерних наук, ФІІТА

(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 82,69% Схожість 17,31%

**Аналіз звіту подібності** (відмітити потрібне):

✓ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

□ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи Вдовиченко А.І.

Керівник роботи Арсенюк І.Р.

Опис прийнятого рішення

Магістерську кваліфікаційну роботу допущено до захисту

Особа, відповідальна за перевірку Озеранський В.С.

Опис прийнятого рішення

Магістерську кваліфікаційну роботу допущено до захисту

Особа, відповідальна за перевірку _____ Озеранський В.С.

## ДОДАТОК Б (довідниковий)

## ІНСТРУКЦІЯ КОРИСТУВАЧА

Для запуску додатку необхідно відкрити файл Quizzer.exe та обрати файл вікторини для завантаження в файлову систему. Далі почати проходити вікторину до повного завершення гри, можливо обрати один з режимів.

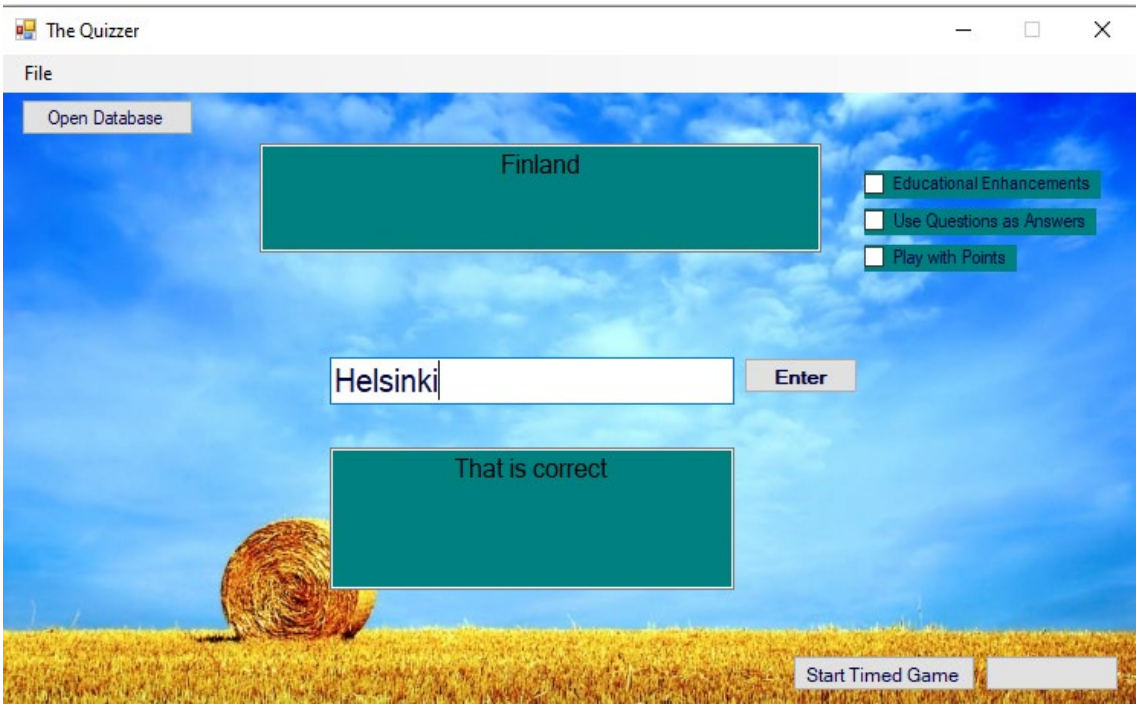



Рисунок Б.1 – Проходження вікторини


Рисунок Б.2 – Отримання вітальної листівки після проходження

## ДОДАТОК В (обов'язковий)

## ЛІСТИНГ ДОДАТКУ

```
using System;
using System.IO;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Media;
using System.Threading;

using Novacode;
//using DocumentFormat.OpenXml;
//using DocumentFormat.OpenXml.Packaging;
//using DocumentFormat.OpenXml.Wordprocessing;
```

```
namespace TheQuizzer
{
   public partial class Form1 : Form
   {
      private static string SEP =
System.IO.Path.DirectorySeparatorChar.ToString();
      private const int EDUCATIONAL_FREQUENCY = 10;
      private const int TIME_LIMIT = 60;
      private const float MAX_QUESTION_FONT_SIZE = 20;
      private const float MIN_QUESTION_FONT_SIZE = 12;
      private const float FONT_SHRINK_SPEED = 1.0f;
      private int numCorrect = 0, numIncorrect = 0;
      private int educationalIndex1 = -1, educationalIndex2 = -1;
      private Random random = new Random();
      private List<int> unaskedIndices = new List<int>();
      private List<Entry> entryList = new List<Entry>();
      private List<int> incorrectEntryIndexes = new List<int>();
      private int currentEntryIndex;
      private string lastOpenPath;
      private bool educationalEnhancements = false;
      private bool useQuestionsAsAnswers = false;
      private bool playWithPoints = false;
      private bool currentQuestionElement1;
      private int points = 0;
      private int countdown = TIME_LIMIT;
      private bool enterKeyDown;
      private string hsFilePath;
      HighScore hs;
      SoundPlayer yep = new SoundPlayer("zsrc" + SEP + "yep.wav");
```

```
SoundPlayer nope = new SoundPlayer("zsrc" + SEP + "nope.wav");
SoundPlayer tick = new SoundPlayer("zsrc" + SEP + "tick.wav");
SoundPlayer ding = new SoundPlayer("zsrc" + SEP + "ding.wav");
List<SoundPlayer> yay = new List<SoundPlayer>();

public Form1()
{
   InitializeComponent();

   for (int i = 1; File.Exists("zsrc" + SEP + "yay" + i + ".wav"); i++)
   {
      SoundPlayer p = new SoundPlayer("zsrc" + SEP + "yay" + i + ".wav");
      yay.Add(p);
   }
   hs = new HighScore();
   hs.FormClosed += new FormClosedEventHandler(hs_FormClosed);
}

void hs_FormClosed(object sender, FormClosedEventArgs e)
{
   checkHighScores();
   hs.Dispose();
   hs = new HighScore();
   hs.FormClosed += new FormClosedEventHandler(hs_FormClosed);
}

private void submitAnswer()
{
   textBox2.SelectAll();
   // lame-o Easter egg
```

```
            if (isCorrectAnswer(textBox2.Text, "Greg is cool"))
            {
               yep.Play();
               textBox3.Text = "You betcha";
               return;
            }
            if (isCorrectAnswer(textBox2.Text, "Hi Greg") ||
isCorrectAnswer(textBox2.Text, "Hi, Greg"))
            {
               yep.Play();
               textBox3.Text = "Hi";
               return;
            }
            bool correct = false;

            if (currentQuestionElement1 && isCorrectAnswer(textBox2.Text,
entryList[currentEntryIndex].getElementTwo()))
                  correct = true;
            else if (!currentQuestionElement1 && isCorrectAnswer(textBox2.Text,
entryList[currentEntryIndex].getElementOne()))
                  correct = true;

            if (correct)
            {
               yep.Stop();
               nope.Stop();
               yep.Play();
               textBox3.Text = "That is correct";
               points += 40;
               numCorrect++;
```

```
            educationalIndex2 = -1;
        }
        else
        {
            yep.Stop();
            nope.Stop();
            nope.Play();
            if (currentQuestionElement1)
            {
                textBox3.Text = "Nice try.  The correct answer is\r\n" +
entryList[currentEntryIndex].getElementTwo()[0];
            }
            else //current question is element2
            {
                textBox3.Text = "Nice try.  The correct answer is\r\n" +
entryList[currentEntryIndex].getElementOne()[0];
            }
            points -= 10;

            if (playWithPoints && points < 0)
            {

                points = 0;
                if (!timer1.Enabled)
                {
                    button2.Text = ":(";
                    button2.Update();
                    textBox3.Update();
                    Splash gameOverScreen = new Splash();
```

```
                    gameOverScreen.BackgroundImage =
System.Drawing.Image.FromFile("zsrc" + SEP + "gameover.jpg");
                    gameOverScreen.Width =
gameOverScreen.BackgroundImage.Width;
                    gameOverScreen.Height =
gameOverScreen.BackgroundImage.Height;
                    gameOverScreen.timer1.Interval = 1500;
                    gameOverScreen.Show();
                    Thread.Sleep(1500);
                }
            }
            numIncorrect++;
            educationalIndex2 = currentEntryIndex;
            if (!incorrectEntryIndexes.Contains(currentEntryIndex))
            {
                incorrectEntryIndexes.Add(currentEntryIndex);
            }
        }

        if (playWithPoints)
        {
            button2.Text = points.ToString();
        }

        generateNextEntry();


    }

    private void generateNextEntry()
```

```
        {
            //choose an index
            if (educationalEnhancements && educationalIndex1 != -1)
            {
                currentEntryIndex = educationalIndex1;
            }
            else
            {
                if (educationalEnhancements && incorrectEntryIndexes.Count != 0
&& random.Next(EDUCATIONAL_FREQUENCY) == 0)
                {
                    textBox1.ForeColor = System.Drawing.Color.DarkBlue;
                    currentEntryIndex =
incorrectEntryIndexes[random.Next(incorrectEntryIndexes.Count)];
                }
                else
                {
                    textBox1.ForeColor = System.Drawing.Color.Black;
                    int entryIndex =
unaskedIndices[random.Next(unaskedIndices.Count)];
                    currentEntryIndex = entryIndex;
                    unaskedIndices.Remove(entryIndex);
                    if (unaskedIndices.Count == 0)
                    {
                        for (int i = 0; i < entryList.Count; i++)
                        {
                            unaskedIndices.Add(i);
                        }
                    }
                }
```

```
    }
    educationalIndex1 = educationalIndex2;
    //choose to use either element1 or element2
    if (useQuestionsAsAnswers && random.Next(2) == 0)
    {
        currentQuestionElement1 = false;
        textBox1.Text = entryList[currentEntryIndex].getElementTwo()[0];
    }
    else
    {
        currentQuestionElement1 = true;
        textBox1.Text = entryList[currentEntryIndex].getElementOne()[0];
    }
    if (!questionboxTimer.Enabled)
    {
        beginFancyTextInBox1();
    }
}

private bool isCorrectAnswer(string givenAnswer, string[] answers)
{
    foreach (string answer in answers)
    {
        if (isCorrectAnswer(givenAnswer, answer)) // compare ignoring case
        {
            return true;
        }
    }
    return false;
}
```

```
        private bool isCorrectAnswer(string givenAnswer, string actualAnswer)
        {
            givenAnswer = givenAnswer.Replace(" ", "");
            actualAnswer = actualAnswer.Replace(" ", "");

            if (String.Compare(givenAnswer, actualAnswer, true) == 0) // compare
ignoring case
            {
                return true;
            }
            if (String.Compare(actualAnswer, "yes", true) == 0)
            {
            // if adding more values, remember to not have spaces.
                string[] yesVals = { "yeah", "yesh", "yea", "yep", "booyeah",
                                     "hellyeah", "fuckyeah", "wellduh", "wellduhh",
                                     "heckyeah", "affirmative"
                                      }; // apologies for the language. Gotta satisfy the end
users.
                foreach (string validAnswer in yesVals)
                {
                    if (String.Compare(givenAnswer, validAnswer) == 0)
                    {
                        return true;
                    }
                }
            }
            if (String.Compare(actualAnswer, "no", true) == 0)
            {
                string[] noVals = { "nope", "nah", "lolno", "noo",
```

```
                    "nooo", "noooo", "nooooo", "noooooo",
                    "nooooooo", "noooooooo", "nooooooooo",
                    "noooooooooo", "hellno", "heckno",
                    "negative"};
    foreach (string validAnswer in noVals)
    {
        if (String.Compare(givenAnswer, validAnswer) == 0)
        {
            return true;
        }
    }
}
if(givenAnswer.EndsWith("duh", true, null))
{
    givenAnswer = givenAnswer.Substring(0, givenAnswer.Length – 3);
}
else if (givenAnswer.EndsWith("duhh", true, null))
{
    givenAnswer = givenAnswer.Substring(0, givenAnswer.Length – 4);
}
if (givenAnswer.EndsWith("stupid", true, null))
{
    givenAnswer = givenAnswer.Substring(0, givenAnswer.Length – 6);
}
if (givenAnswer.EndsWith("lol", true, null))
{
    givenAnswer = givenAnswer.Substring(0, givenAnswer.Length – 3);
}
```

```
            if (String.Compare(givenAnswer, actualAnswer, true) == 0) // compare ignoring case
            {
               return true;
            }

            // strip away ending 's' from both given and actual answers to negate possible plural forms
            if (givenAnswer.EndsWith("s", true, null))
            {
               givenAnswer = givenAnswer.Substring(0, givenAnswer.Length – 1);
            }
            if (actualAnswer.EndsWith("s", true, null))
            {
               actualAnswer = actualAnswer.Substring(0, actualAnswer.Length -1);
            }
            if (String.Compare(givenAnswer, actualAnswer, true) == 0) // compare ignoring case
            {
               return true;
            }

            //TODO: deal with plural forms, answers ending with 'duh'

            return false;
         }

         private void openDatabaseToolStripMenuItem_Click(object sender, EventArgs e)
         {
```

```
            tick.Play();
            openDatabase();
        }

        private void button4_Click(object sender, EventArgs e)
        {
            tick.Play();
            openDatabase();
        }

        private void openDatabase()
        {
            OpenFileDialog fDialog = new OpenFileDialog();
            fDialog.Title = "Open Database";
            fDialog.Filter = "Input files (*.txt;*.docx)|*.txt;*.docx";
            fDialog.Multiselect = true;
            if (lastOpenPath == null)
            {

//MessageBox.Show(Path.GetDirectoryName(Application.ExecutablePath).Replace("
\\", "/") + "/Anatomy");
                fDialog.InitialDirectory                                    =
Path.GetDirectoryName(Application.ExecutablePath) + SEP + "Anatomy";
            }
            else
            {
                fDialog.InitialDirectory = lastOpenPath;
            }
            if (fDialog.ShowDialog() == DialogResult.OK)
            {
```

```
            textBox1.Text = "Loading...";
            textBox1.Font = changeFontSize(textBox1.Font,
MIN_QUESTION_FONT_SIZE);
            textBox1.Update();
            textBox1.Font = changeFontSize(textBox1.Font,
MAX_QUESTION_FONT_SIZE);
            hsFilePath = fDialog.FileName.Substring(0,
fDialog.FileName.LastIndexOf('.')) + ".HighScore";

            lastOpenPath = fDialog.FileName.Substring(0,
fDialog.FileName.LastIndexOf(SEP));
            string[] filesToOpen = fDialog.FileNames;
            foreach (string s in filesToOpen)
            {
               if (s.EndsWith("txt"))
               {
                  AddEntriesFromTxt(s);
               }
               else if (s.EndsWith("docx"))
               {
                  AddEntriesFromDocx(s);
               }
            }
            if (entryList.Count > 1)
            {
               hsFilePath = null;
            }
         }
      }
```

```
        private void AddEntriesFromTxt(string fileName)
        {
            TextReader tr = new StreamReader(fileName);
            string line;
            while ( (line = tr.ReadLine()) != null)
            {
                //check if line is valid: exactly 1 equals sign
                if (line.IndexOf('=') == line.LastIndexOf('=') && line.IndexOf('=') != -
1)
                {
                    string firstStuff = line.Substring(0, line.IndexOf('='));
                    string secondStuff = line.Substring(line.IndexOf('=') + 1);

                    AddEntries(firstStuff, secondStuff);

                    string justFileName = fileName.Substring(0,
fileName.LastIndexOf('.'));
                    justFileName =
justFileName.Substring(justFileName.LastIndexOf(SEP) + 1);
                    textBox1.Text = justFileName + " – Added";
                }
            }
        }

        private void AddEntriesFromDocx(string fileName)
        {
            /*
            // The following method was adapted from some copypasta from
http://stackoverflow.com/questions/11240933/extract-table-from-docx
            StringBuilder result = new StringBuilder();
```

```
            MessageBox.Show(fileName);
        WordprocessingDocument wordProcessingDoc = null;
        try
        {
            wordProcessingDoc = WordprocessingDocument.Open(fileName,
true);
        }
        catch (IOException)
        {
            MessageBox.Show("Looks like another process (probably Microsoft
Word) has document " + fileName + " open – close it and try again.");
            return;
        }

        IEnumerable<Paragraph> paragraphElement =
wordProcessingDoc.MainDocumentPart.Document.Descendants<Paragraph>();

        foreach (OpenXmlElement section in
wordProcessingDoc.MainDocumentPart.Document.Body.Elements<OpenXmlEleme
nt>())
        {
            if (section.GetType().Name == "Table")
            {
                Table tab = (Table)section;
                string[] text = new string[2];
                foreach (TableRow row in tab.Descendants<TableRow>())
                {
                    int columnIndex = 0;
                    foreach (TableCell cell in row.Descendants<TableCell>())
                    {
```

```
                    if (columnIndex > 1) break;

                    text[columnIndex] = cell.InnerText.Trim();
                    columnIndex++;
                }
                if (text[0] != null && text[1] != null && text[0].Length != 0 &&
text[1].Length != 0)
                {
                    string firstStuff = text[0];
                    string secondStuff = text[1];
                    // treat anything aftere a # as a comment
                    if (!secondStuff.StartsWith("."))
                    {
                        int index = secondStuff.IndexOf("//");
                        if (index >= 0)
                        {
                            secondStuff = secondStuff.Substring(0, index);
                        }
                        AddEntries(firstStuff, secondStuff);
                    }
                }
            }
        }

        wordProcessingDoc.Close();
        */
        using (DocX document = DocX.Load(fileName))
        {
            Table t = document.Tables[0];
```

```
    foreach (Row r in t.Rows)
    {
        string firstStuff = "", secondStuff = "";
        for (int i = 0; i < 2; i++)
        {
            string text = "";
            foreach (Paragraph p in r.Cells[i].Paragraphs)
            {
                if (p == null) break;
                text += p.Text;
            }
            text = text.Trim();
            if (i == 0) firstStuff = text;
            else secondStuff = text;
        }
        if (firstStuff.Length != 0 && secondStuff.Length != 0)
        {
            // treat anything after a # as a comment
            if (!secondStuff.StartsWith("."))
            {
                int index = secondStuff.IndexOf("//");
                if (index >= 0)
                {
                    secondStuff = secondStuff.Substring(0, index);
                }
                AddEntries(firstStuff, secondStuff);
            }
        }
    }
}
```

```
            string justFileName = fileName.Substring(0, fileName.LastIndexOf('.'));
            justFileName = justFileName.Substring(justFileName.LastIndexOf(SEP)
+ 1);

            textBox1.Text = justFileName + " – Added";
        }

        private void AddEntries(string firstStuff, string secondStuff)
        {
            // TODO: implement questions with multiple answers, possibly plural
structures
            /* Get the first elements */
            string[] firstElements = firstStuff.Split('|');

            /* Get the second elements */

            string[] secondElements = secondStuff.Split('|');

            entryList.Add(new Entry(firstElements, secondElements));
            unaskedIndices.Add(entryList.Count – 1);

        }

        private void textBox2_Leave(object sender, EventArgs e)
        {
            textBox2.Focus();
        }

        private void button1_Click(object sender, EventArgs e)
        {
```

```
            activateButton();
        }

        private void textBox1_TextChanged(object sender, EventArgs e)
        {
            beginFancyTextInBox1();
        }

        private void beginFancyTextInBox1()
        {
            textBox1.Font = changeFontSize(textBox1.Font,
MAX_QUESTION_FONT_SIZE);
            questionboxTimer.Enabled = true;
        }

        private System.Drawing.Font changeFontSize(System.Drawing.Font font,
float size)
        {
            return new System.Drawing.Font(font.Name, size, font.Style, font.Unit,
font.GdiCharSet, font.GdiVerticalFont );
        }

        private void questionboxTimer_Tick(object sender, EventArgs e)
        {
            if (textBox1.Font.Size > MIN_QUESTION_FONT_SIZE)
            {
                textBox1.Font = changeFontSize(textBox1.Font, textBox1.Font.Size –
FONT_SHRINK_SPEED);
            }
            else
```

```
            {
                textBox1.Font = changeFontSize(textBox1.Font, MIN_QUESTION_FONT_SIZE);
                questionboxTimer.Enabled = false;
            }
        }

        private void checkBox1_CheckedChanged(object sender, EventArgs e)
        {
            tick.Play();
            if (checkBox1.Checked)
            {
                educationalEnhancements = true;
            }
            else
            {
                educationalEnhancements = false;
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            string text = "You have answered " + (numCorrect + numIncorrect) + " questions.\r\n" + numCorrect + " answers were correct.";
            if (incorrectEntryIndexes.Count != 0)
            {
                text += "\r\nYou answered the following incorrectly:\r\n";
            }
            foreach(int i in incorrectEntryIndexes)
            {
```

```
            text += "\r\n" + entryList[i].getElementOne()[0];
        }
        MessageBox.Show(text);
    }

    private void checkBox2_CheckedChanged(object sender, EventArgs e)
    {
        tick.Play();
        if (checkBox2.Checked)
        {
            useQuestionsAsAnswers = true;
        }
        else
        {
            useQuestionsAsAnswers = false;
        }
    }

    private void Form1_Load(object sender, EventArgs e)
    {
        Splash splash = new Splash();
        splash.Show();
    }

    private void refreshDatabaseToolStripMenuItem_Click(object sender,
EventArgs e)
    {
        tick.Play();
        numCorrect = 0;
        numIncorrect = 0;
```

```
            points = 0;
            entryList.Clear();
            unaskedIndices.Clear();
            incorrectEntryIndexes.Clear();
            textBox1.Text = "";
            textBox2.Text = "";
            textBox3.Text = "";
            button1.Font = new System.Drawing.Font(button1.Font,
System.Drawing.FontStyle.Bold);
            button1.Text = "Start";
            button2.Text = points.ToString();
            openDatabase();
        }

        private void textBox2_Click(object sender, EventArgs e)
        {
            textBox2.SelectAll();
        }

        private void checkBox3_CheckedChanged(object sender, EventArgs e)
        {
            if (!timer1.Enabled)
            {
                tick.Play();
            }
            if (checkBox3.Checked)
            {
                playWithPoints = true;
                points = 0;
                button2.Text = points.ToString();
```

```
    }
    else
    {
        playWithPoints = false;
        button2.Text = "";
    }
}

private void button3_Click(object sender, EventArgs e)
{
    textBox2.SelectAll();
    if (button3.Text == "Start Timed Game" && entryList.Count != 0)
    {
        educationalEnhancements = false;
        useQuestionsAsAnswers = false;
        playWithPoints = true;

        points = 0;
        button2.Text = points.ToString();
        checkBox1.Checked = false;
        checkBox1.Enabled = false;
        checkBox2.Checked = false;
        checkBox2.Enabled = false;
        checkBox3.Checked = true;
        checkBox3.Enabled = false;
        countdown = TIME_LIMIT;
        button3.Text = countdown.ToString();
        generateNextEntry();
        yep.Play();
        if (button1.Text == "Start")
```

```
            {
                button1.Text = "Enter";
            }
            timer1.Enabled = true;
        }
        else
        {
            button3.Text = "Start Timed Game";
            timer1.Enabled = false;
        }
    }

    public void checkHighScores()
    {
        TextReader tr = new StreamReader(hsFilePath);
        List<string> highscores = new List<string>();
        string line = tr.ReadLine();
        while (line != null)
        {
            highscores.Add(line);
            line = tr.ReadLine();
        }
        tr.Close();
        bool inserted = false;
        for (int i = 0; i < highscores.Count; i++ )
        {
            if (this.points >
Convert.ToInt32(highscores[i].Substring(highscores[i].LastIndexOf('=') + 1)))
            {
                string name = hs.GetName();
```

```
                highscores.Insert(i, name + "=" + points);
                inserted = true;
                break;
            }
        }
        if (highscores.Count < 5 && inserted == false)
        {
            string name = hs.GetName();
            highscores.Add(name + "=" + points);
        }
        if (highscores.Count > 5)
        {
            highscores.RemoveAt(highscores.Count – 1);
        }
        File.Delete(hsFilePath);
        File.Create(hsFilePath).Close();
        TextWriter tw = new StreamWriter(hsFilePath);
        foreach (string h in highscores)
        {
            tw.WriteLine(h);
        }
        tw.Close();
        String messageBoxString = "High Scores:\n";
        foreach(string s in highscores)
        {
            messageBoxString += s.Substring(0, s.IndexOf('=')).Replace("=", "") +
" – ";
            messageBoxString += s.Substring(s.IndexOf('=') + 1).Replace("=", "")
+ "\n";
        }
```

```
            MessageBox.Show(messageBoxString);
        }

        private void timer1_Tick(object sender, EventArgs e)
        {
            countdown--;
            if (countdown <= 0)
            {
                bool awarded = false;
                for (int i = yay.Count – 1; i >= 0; i--)
                {
                    if (points >= 50 * (i+1))
                    {
                        Splash award = new Splash();
                        try
                        {
                            System.Drawing.Image img =
System.Drawing.Image.FromFile("zsrc" + SEP + "yay" + (i + 1) + ".jpg");
                            award.BackgroundImage = img;
                            award.Height = img.Height;
                            award.Width = img.Width;
                            award.timer1.Interval = 4000;
                            award.Show();
                            award.PlaySound(yay[i]);
                            awarded = true;
                            break;
                        }
                        catch (FileNotFoundException) {  }
                    }
                }
```

```
            if(awarded == false)
            {
               try
               {
                  Splash emptySplash = new Splash();
                  System.Drawing.Image                    img                    =
System.Drawing.Image.FromFile("zsrc" + SEP + "gameover.jpg");
                  emptySplash.BackgroundImage = img;
                  emptySplash.Width = img.Width;
                  emptySplash.Height = img.Height;
                  emptySplash.timer1.Interval = 1000;
                  emptySplash.Show();
                  emptySplash.PlaySound(ding);

               }
               catch (FileNotFoundException) { }
            }
            button3.Text = "Start Timed Game";
            timer1.Enabled = false;

            checkBox1.Enabled = true;
            checkBox2.Enabled = true;
            checkBox3.Enabled = true;
            if (hsFilePath != null)
            {
               bool refreshHighScores = false;
               if (!File.Exists(hsFilePath))
               {
                  File.Create(hsFilePath).Close();
               }
```

```
            TextReader tr = new StreamReader(hsFilePath);
            string line;
            List<string> highscores = new List<string>();
            line = tr.ReadLine();
            while (line != null)
            {
               highscores.Add(line);
               line = tr.ReadLine();
            }
            tr.Close();
            foreach (string s in highscores)
            {
               if (this.points > Convert.ToInt32(s.Substring(s.LastIndexOf('=') +
1)))
               {
                  refreshHighScores = true;
               }
            }
            if (highscores.Count < 5 && refreshHighScores == false)
            {
               refreshHighScores = true;
            }
            if (refreshHighScores)
            {
               hs.Show();
            }
         }
      }
      else
      {
```

```
            button3.Text = countdown.ToString();
        }
    }

    private void exitToolStripMenuItem_Click(object sender, EventArgs e)
    {
        this.Close();
    }

    private void activateButton()
    {
        if (entryList.Count == 0 || enterKeyDown)
        {
            return;
        }

        if (button1.Text == "Start")
        {
            yep.Play();
            button1.Font = new System.Drawing.Font(button1.Font,
System.Drawing.FontStyle.Bold);
            button1.Text = "Enter";
            generateNextEntry();
        }
        else
        {
            submitAnswer();
        }
    }
```

```
        private void textBox2_KeyDown(object sender, KeyEventArgs e)
        {
            if (!enterKeyDown && e.KeyData == Keys.Enter)
            {
                activateButton();
                enterKeyDown = true;
                e.SuppressKeyPress = true;
            }

        }

        private void textBox2_KeyUp(object sender, KeyEventArgs e)
        {
            enterKeyDown = false;
            e.Handled = true;
            e.SuppressKeyPress = true;
        }

        private void viewHighScoresToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            if (hsFilePath != null)
            {
                TextReader tr = new StreamReader(hsFilePath);
                List<string> highscores = new List<string>();
                string line = tr.ReadLine();
                while (line != null)
                {
                    highscores.Add(line);
                    line = tr.ReadLine();
```

ДОДАТОК Г **(обов'язковий)**

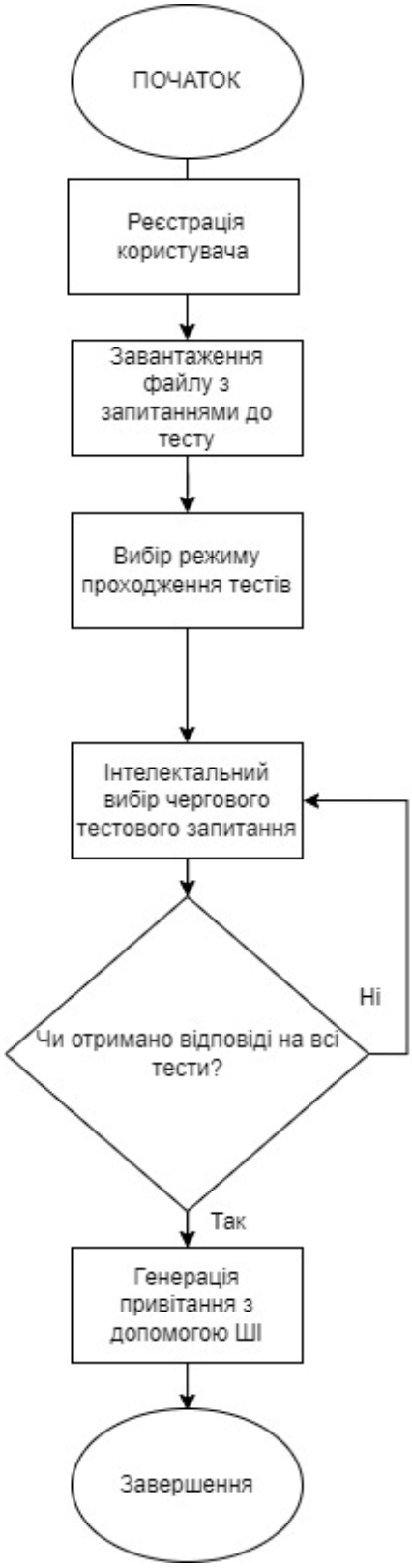
**ІЛЮСТРАТИВНА ЧАСТИНА**

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ГЕЙМІФІКАЦІЇ ПРОЦЕСУ ТЕСТУВАННЯ ПІД ЧАС НАВЧАННЯ.

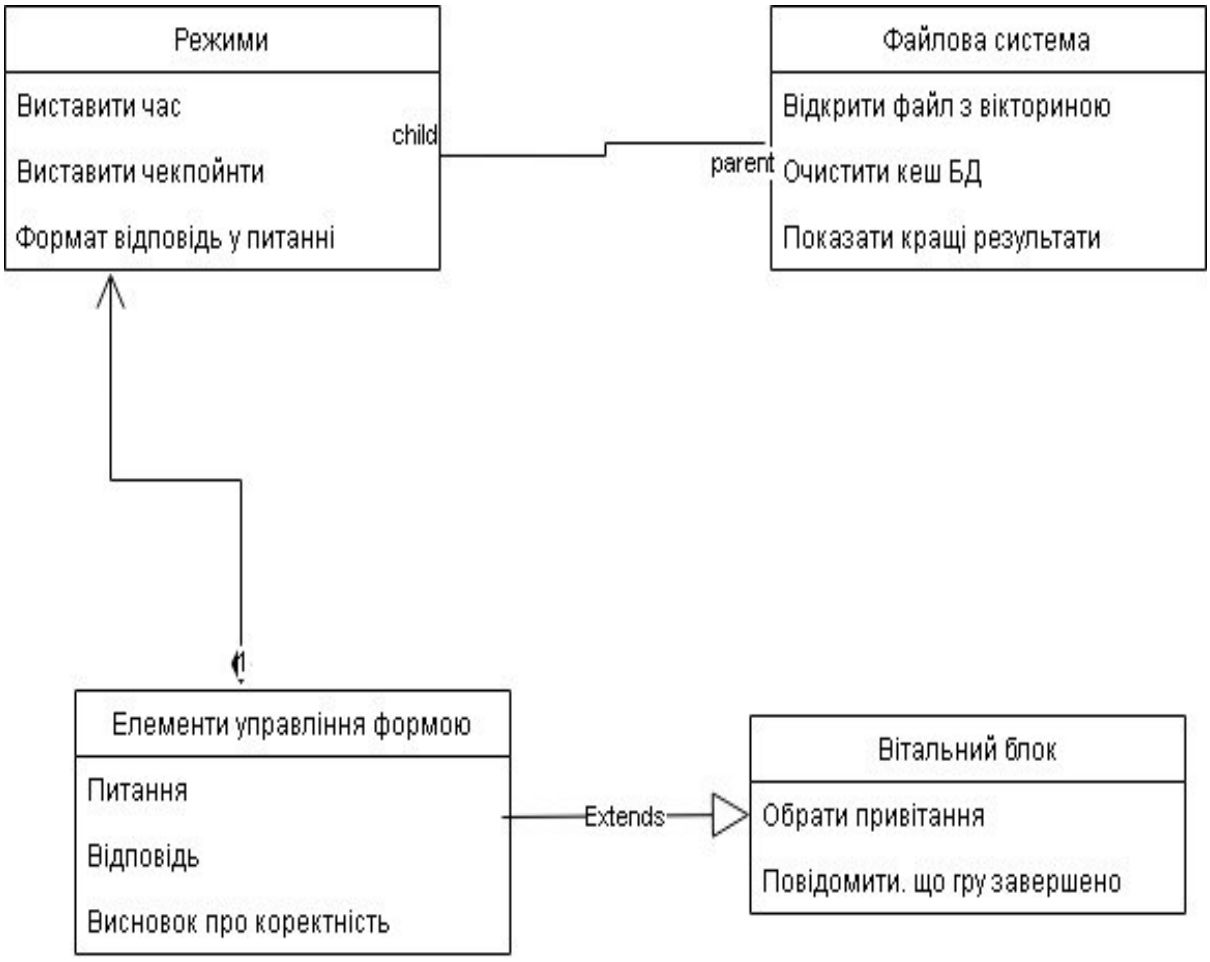
Виконав: студент 2-го курсу, групи 3КН-22м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
_____ Вдовиченко А. І.
(прізвище та ініціали)

Керівник: д.т.н., проф. кафедри КН
_____ Арсенюк І. Р.
(прізвище та ініціали)
«_____»_____2023 р.

Вінниця ВНТУ - 2023 рік

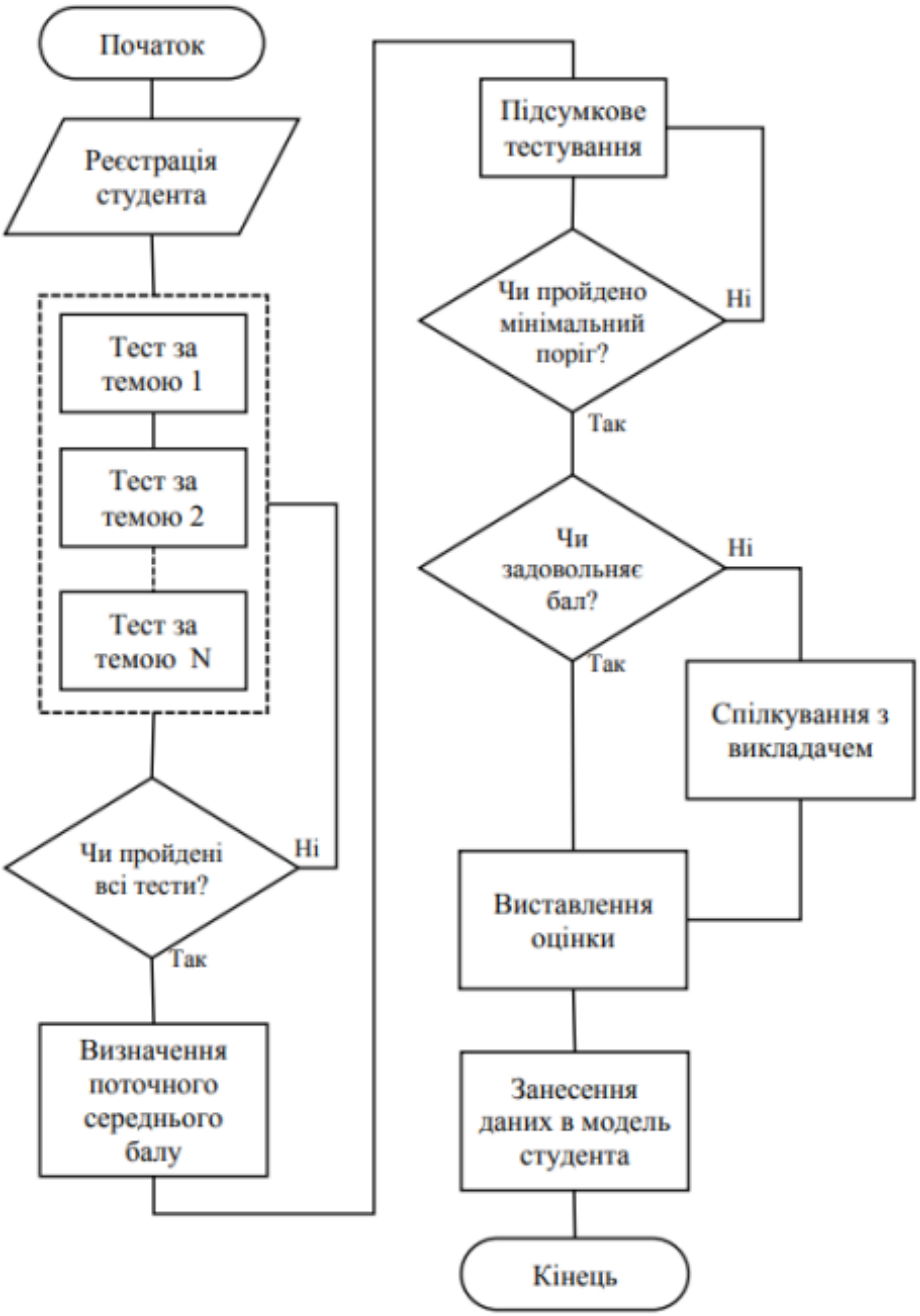



Алгоритм роботи модуля гейміфікації

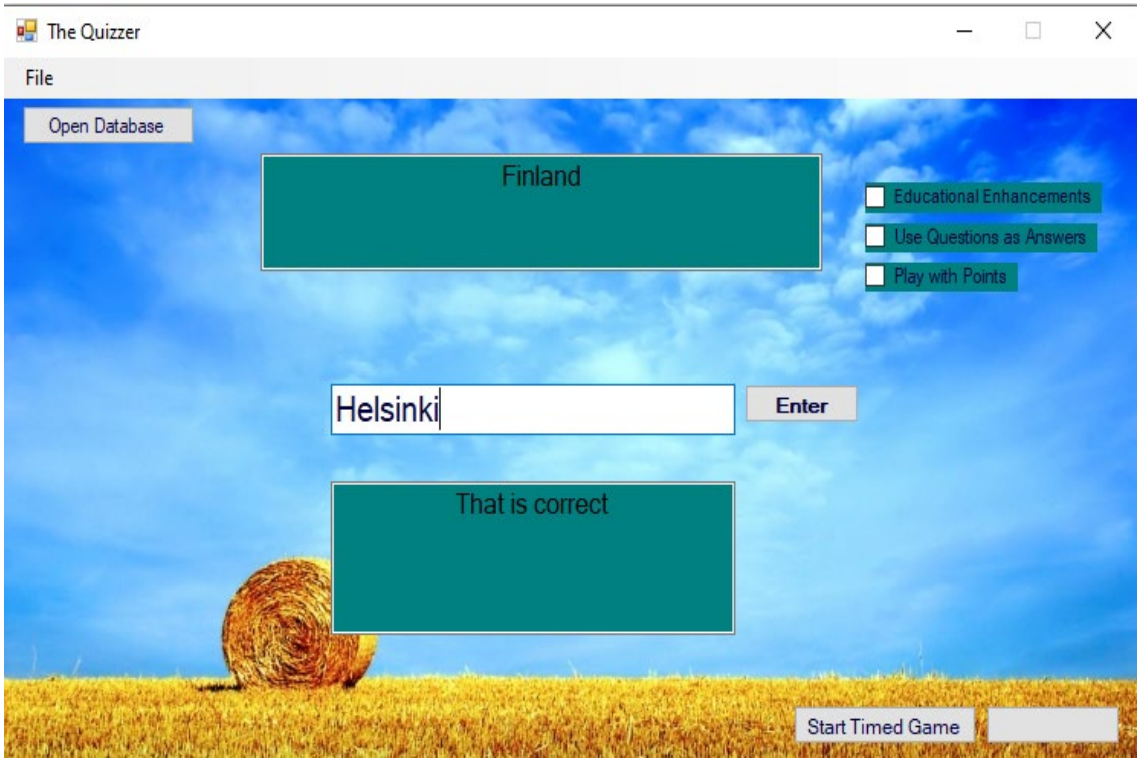



ЮМЛ-діаграма класів додатку


ЮМЛ-діаграма прецедентів додатку

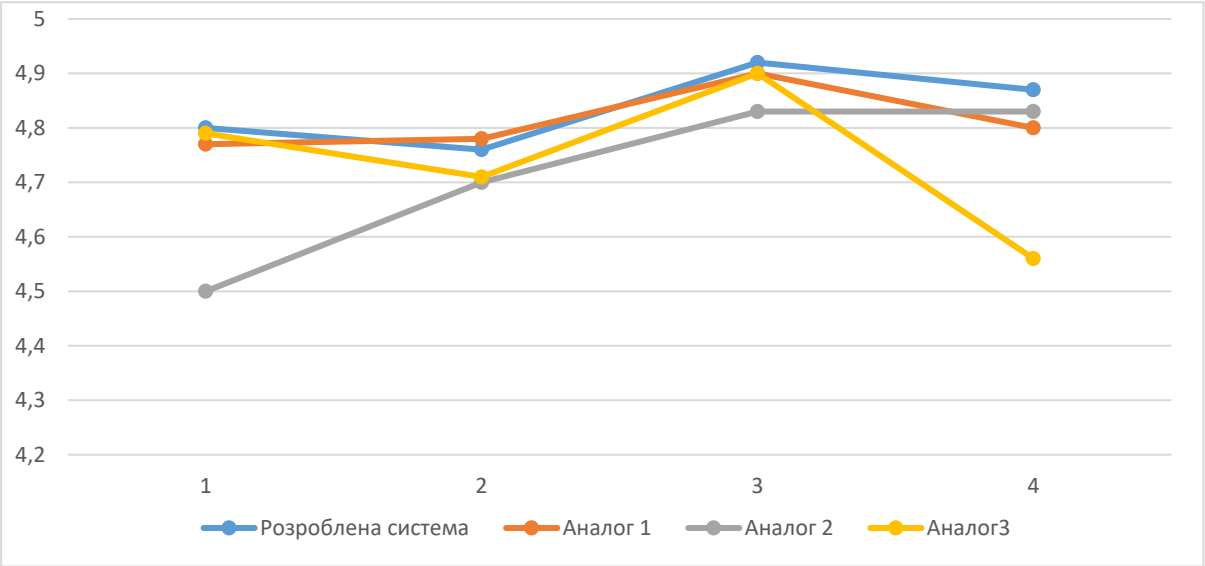



Блок-схема вдосконаленого алгоритму адаптиввного тестування




Скрінншот проходження тесту


Скріншот завершення тесту




Порявняльна характеристика систем гейміфікації