Вінницький національний технічний університет
(повне найменування вищого навчального закладу)
Факультет інтелектуальних інформаційних технологій та автоматики
(повне найменування інституту, назва факультету (відділення))
Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

**Бакалаврська кваліфікаційна робота на тему:**

# ПРОГРАМНИЙ МОДУЛЬ ОЦІНЮВАННЯ ЯКОСТІ КОНТРОЛЬНИХ ЗАВДАНЬ

Виконав: студент 4 курсу, групи 4КН-21б
спеціальності 122 – Комп'ютерні науки
(шифр і назва напряму підготовки, спеціальності)
Гнатюк М.І.
(прізвище та ініціали)
Керівник: к.т.н., ст. викл. кафедри КН
Петришин С.І.
(прізвище та ініціали)
«04» червня 2025 р.
Рецензент: к.т.н., доцент кафедри АІІТ
Гармаш В.В.
(прізвище та ініціали)
«05» червня 2025 р.

**Допущено до захисту**
Завідувач кафедри КН
д.т.н., проф. Яровий А.А.
(прізвище та ініціали)
«06» червня 2025 р.

Вінниця ВНТУ – 2025 рік

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

**ЗАТВЕРДЖУЮ**
Завідувач кафедри КН
проф., д.т.н. Яровий А.А.
"05" травня 2025 року

**ЗАВДАННЯ**
**НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ**

Гнатюку Миколі Івановичу

1. Тема роботи: Програмний модуль оцінювання якості контрольних завдань.
керівник роботи: Петришин Сергій Іванович, к.т.н., ст. викладач
затверджені наказом вищого навчального закладу "20" березня 2025 року № 97

2. Строк подання студентом роботи 30 травня 2025р.

3. Вихідні дані до роботи:
Кількість тестових контрольних завдань – не менше 10шт;
Кількість користувачів тестів – не менше 100шт;
Операційна система Android з рівнем API не нижче 19;
Мова програмування – об'єктно-орієнтована;

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):
вступ, обгрунтування доцільності розробки програмного модуля оцінювання якості контрольних завдань, розробка програмного модуля оцінювання якості контрольних завдань, програмна реалізація модуля оцінювання якості контрольних завдань; висновки; список використаних джерел; додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень):
структура програмного модуля оцінювання якості контрольних завдань; схема алгоритму оцінювання якості контрольних завдань; загальна UML-діаграма класів.

6. Консультанти розділів проєкту (роботи)

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| 1-3 | Петришин С.І. к.т.н., ст. викладач | | |
| | | | |
| | | | |
| | | | |

6. Дата видачі завдання 05 травня 2025 року

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва та зміст етапу | Термін виконання | | Примітка |
|---|---|---|---|---|
| | | початок | закінчення | |
| 1. | Обгрунтування доцільності розробки програмного модуля оцінювання якості контрольних завдань | 05.05.2025 | 08.05.2025 | |
| 2. | Розробка програмного модуля оцінювання якості контрольних завдань | 09.05.2025 | 11.05.2025 | |
| 3. | Програмна реалізація модуля оцінювання якості контрольних завдань | 12.05.2025 | 16.05.2025 | |
| 4. | Розробка інструкції користувача. | 17.05.2025 | 25.05.2025 | |
| 5. | Висновки та аналіз отриманих результатів | 26.05.2025 | 01.06.2025 | |
| 6. | Оформлення матеріалів до захисту БКР | 02.06.2025 | 04.06.25 | |

Студент _____ Гнатюк М.І.
(підпис) (прізвище та ініціал

Керівник проєкту (роботи) _____ Петришин С.І.
(підпис) (прізвище та ініціал

# АНОТАЦІЯ

Гнатюк М.І. Програмний модуль оцінювання якості контрольних завдань. Бакалаврська кваліфікаційна робота складається з 71 сторінок формату А4, на яких є 29 рисунків, 3 таблиці, список використаних джерел містить 20 найменуваннь.

В роботі обґрунтовано доцільність розробки програмного модуля оцінювання якості контрольних завдань. В роботі визначені функціональні і нефункціональні вимоги до додатка оцінювання якості контрольних завдань. Виконано порівняння існуючих додатків оцінювання якості контрольних завдань. Розроблено алгоритм роботи програмного засобу оцінювання якості завдань.

Проведене тестування програми на наявність помилок або дефектів, показало, що додаток повністю придатний для користування.

Ключові слова: тестування, контрольні завдання, оцінювання якості, статистична обробка.

## ABSTRACT

Hnatyuk M. Software module of quality assessment of control tasks. Bachelor's qualification work consists of 71 pages of A4, which has 10 figures, 4 tables, the list of sources used contains 20 names.

The work substantiates the feasibility of developing a software module for assessing the quality of control tasks. The work defines functional and non -functional requirements for the quality assessment of the quality of control tasks. Comparison of existing applications of quality assessment of control tasks has been compared. The algorithm of the software assessment of quality assessment has been developed.

Testing the program for errors or defects, showed that the application is fully suitable for use.



Keywords: testing, control tasks, quality assessment, statistical processing.

# ЗМІСТ

# ВСТУП

**Актуальність теми.** Контрольні завдання є одним із головних інструментів перевірки знань учнів або студентів. Від того, наскільки якісно вони складені, залежить об’єктивність оцінювання, ефективність навчального процесу та рівень підготовки здобувачів освіти. Якісне контрольне завдання має відповідати темі, навчальним цілям і рівню підготовки учнів. Якщо завдання занадто легкі або навпаки — надто складні, це може призвести до неправильних результатів оцінювання та втрати мотивації до навчання.

У сучасних умовах цифровізації освіти все частіше використовуються електронні форми контролю знань, тому виникає потреба у системному підході до оцінювання не лише відповідей учнів, а й самих завдань. Це дозволяє покращити якість навчання, уникати помилок у формулюванні питань, а також робить процес контролю знань більш ефективним і справедливим.

Інформаційні технології мають високий педагогічний потенціал, дають змогу суттєво різноманітити методи організації та реалізації ефективного навчального процесу, створювати інтелектуальні навчальні системи та технології, які включають: цілеспрямовано підготовлені навчальні матеріали, програмно-методичні комплекси, навчальні програми, за допомогою яких контролюють засвоєні знання та набуті навички й уміння [1].

Сьогодні впровадження інформаційних технологій в навчальний процес, організований на базі різних технологій навчання, йде дуже активно. Це викликано, по-перше, тим, що навчання без використання інформаційних технологій не прогресивне, по-друге, різко зріс об’єм інформації, необхідний студентам, та традиційні засоби і методи навчання вже не відповідають вимогам часу. Важливою проблемою інформатизації навчального процесу є не лише використання існуючих сучасних інформаційних ресурсів, а і створення нових, які б задовольняли всім вимогам та потребам конкретного навчального закладу, супроводжувалися методичними вказівками, передбачали обмін досвідом при впровадженні їх педагогами в навчальний процес. Тому одним із атрибутів навчального процесу

стають інтелектуальні технології навчання і контролю тестових завдань. У зв’язку з цим дане бакалаврське дослідження є досить актуальним.

**Метою бакалаврської кваліфікаційної роботи** є розширення функціональних можливостей програмного забезпечення оцінювання якості контрольних завдань.

**Об’єктом дослідження** є процес оцінювання якості контрольних завдань.

**Предметом дослідження** є програмні засоби для оцінювання якості контрольних завдань.

Для досягнення поставленої мети необхідно виконати такі задачі:

- обґрунтувати доцільність розробки програмного модуля оцінювання якості контрольних завдань;
- проаналізувати переваги та недоліки програм-аналогів для оцінювання якості контрольних завдань;
- розробити алгоритм оцінювання якості контрольних завдань;
- здійснити програмну реалізацію модуля оцінювання якості контрольних завдань;
- виконати тестування програмного модуля оцінювання якості контрольних завдань та провести аналіз отриманих результатів.

# 1 ОБГРУНТУВАННЯ ДОЦІЛЬНОСТІ РОЗРОБКИ ПРОГРАМНОГО МОДУЛЯ ОЦІНЮВАННЯ ЯКОСТІ КОНТРОЛЬНИХ ЗАВДАНЬ

## 1.1 Аналіз предметної області оцінювання якості знань

Освіта й наука є надзвичайно важливими інструментами суспільної трансформації [3]. З огляду на світові комунікаційні тенденції в науковій та освітній сферах, у зв’язку з активним впливом мобільних технологій на шляху до розбудови суспільства знань, маючи потужні наукові школи, навчальні й науково-дослідні установи та високий інтелектуальний потенціал, суспільство перебуває на шляху вдосконалення комунікаційних зв’язків в освітній сфері. Тому в процесі аналізу можливих шляхів реалізації програмного модуля оцінювання якості контрольних завдань, було обрано створити мобільний додаток. Адже, на сьогодні мобільні технології створюють підґрунтя для використання нових способів комунікації: соціальної, освітньої, наукової.

Програмний модуль оцінювання якості контрольних завдань за допомого мобільних додатків можна використовувати для:

– дистанційної освіти , адже дистанційна освіта стає лідером навчальних технологій;

– персоналізації навчання, на противагу уніфікованим підходам в освіті, що потребують від усіх суб’єктів навчання однакових результатів;

– персоналізації освітніх програм, а саме врахування індивідуальних психологічних характеристик особистості як основи для персональних освітніх програм. Вони можуть стати тим підґрунтям, завдяки якому з’явиться мотивація навчання і набудуть нового поштовху у розвитку інтелект, творчість та креативність;

– гейміфікація, що передбачає впровадження ігрових технологій у неігрові ситуації, технологія винагород як метод підвищення мотивації навчання та поліпшення його якості, тобто формально освіта гейміфікована, оскільки використовує систему заохочень – позитивні оцінки і перехід до наступного класу

чи курсу як новий level up;

– розширення інтерактивності освітніх ресурсів. Інтерактивні підручники мають докорінно змінити «традиційні» подання і інтерпретацію навчального матеріалу – лінійна побудова курсів та їх текстове представлення не можуть забезпечити багатовимірність сучасного навчального процесу, яка підтримується мультимедіа-технологіями. Наприклад, кольорові фото, аудіо- та відеопідтримка, інтерактивна інфографіка тощо;

– віртуалізація освітнього процесу, передбачає собою організацію освітнього процесу з використанням віртуальних технологій та відеоігр це є унікальною можливістю надати знання про реальний світ через інтерактивне занурення у віртуальний світ [4].

### 1.2 Аналіз існуючих програм-аналогів оцінювання якості знань

Існує певна кількість мобільних додатків, які надають змогу оцінити якість знань користувача, однак не всі з них повною мірою виконують необхідні функції або відповідають конкретним вимогам навчального процесу. Більшість таких додатків орієнтовані на загальні тести, мають обмежений функціонал або не дозволяють проводити глибокий аналіз результатів. Крім того, деякі з них не підтримують індивідуальні налаштування для різних типів завдань, предметів чи освітніх рівнів.

Через це виникає потреба у створенні спеціалізованого програмного продукту з оптимальним набором функцій, який би враховував практичні потреби викладачів, студентів або учнів. Щоб визначити, які саме функції повинні бути реалізовані в новому додатку, було проведено аналіз найбільш відомих аналогів.

У результаті дослідження розглянуто переваги — такі як зручний інтерфейс, швидка перевірка відповідей, можливість самостійного створення тестів — а також виявлено недоліки: відсутність адаптивності, складність у редагуванні завдань, обмежені можливості аналізу статистики тощо.

Основним конкурентом є додаток QuizUp [5] – найбільш простий і цікавий

додаток. Він має інтуїтивно простий дизайн і зрозумілий користувачеві набір функцій (рис. 1.1). Цей додаток буде зручним для тих, хто є фанатом яскравої графіки, і кому не потрібен поглиблений статистичний аналіз зроблених помилок.


Рисунок 1.1 - Додаток QuizUp

Наступним аналогом розробки є додаток Quiz Your Friends [6]. Це популярний і зручний мобільний додаток, основною ідеєю якого є створення та проходження вікторин у дружньому, неформальному середовищі. Користувач може створити власний тест, додати до нього запитання, варіанти відповідей та потім надіслати посилання на нього своїм друзям. Останні, у свою чергу, проходять тест, а результати відображаються у спеціальному списку, що дозволяє порівняти відповіді між користувачами (рис. 1.2).

Особливістю цього додатку є можливість збереження створених вікторин, що дає змогу повертатися до них пізніше, повторно проходити або редагувати при

потребі. Така функціональність підвищує зручність використання та робить додаток привабливим для ширшого кола користувачів — не лише для розваги, а й для навчальних цілей.


Рисунок 1.2 - Додаток Quiz Your Friends

Проте слід зазначити, що Quiz Your Friends орієнтований переважно на неформальне спілкування, і тому має обмежений функціонал з точки зору глибокого аналізу знань чи педагогічного оцінювання. У ньому, наприклад, відсутні розширені можливості статистичного аналізу результатів, систематизації запитань за темами або рівнями складності. Це обмежує його застосування у професійному освітньому середовищі.

DK Quiz [7] – це мобільний додаток, який пропонує понад 100 готових тестів, охоплюючи широкий спектр тем, зокрема історію, науку, географію, культуру, їжу тощо (рис. 1.3). Такий різноманітний контент робить додаток привабливим для користувачів різного віку та з різними інтересами. Кожен тест складається з кількох запитань із варіантами відповідей, що дозволяє не лише перевірити знання, а й

дізнатися щось нове.

Важливою особливістю DK Quiz є гейміфікація процесу тестування: чим швидше користувач дає правильну відповідь, тим більше балів він отримує. Це стимулює не тільки точність, а й швидкість мислення, що додає інтерактивності та азарту до процесу проходження тестів.

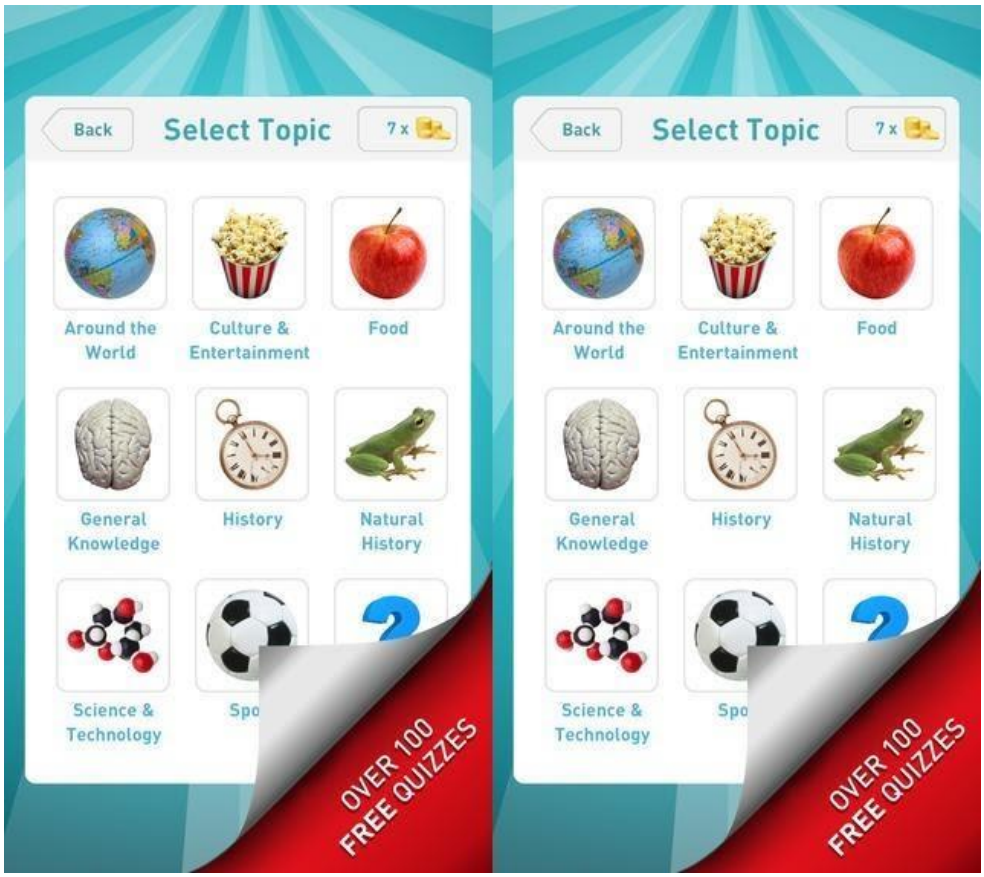


Рисунок 1.3 - Додаток DK Quiz

Крім того, додаток має добре реалізовану систему відображення результатів. Користувач може бачити свою статистику, зокрема відсоток виграшів і програшів, що дозволяє відстежувати прогрес і порівнювати досягнення з попередніми результатами. Така функція є корисною для самостійної оцінки рівня знань.

Однак, незважаючи на загальну привабливість, додаток має і певні недоліки. Зокрема, останні оновлення не містять нових функцій або розширень функціональності. Це може зробити програму менш цікавою для користувачів у довготривалій перспективі, оскільки відсутність регулярного оновлення контенту й

можливостей призводить до зниження мотивації. Також у додатку немає можливості створювати власні тести, аналізувати типові помилки чи формувати тести відповідно до рівня знань конкретного користувача.

Переваги і недоліки аналогів розробки наведено в таблиці 1.1.

Таблиця 1.1 – Порівняння програм аналогів оцінювання якості контрольних завдань

| Назва | Переваги | Недоліки |
|---|---|---|
| QuizUp | – Зручний та інтуїтивно зрозумілий інтерфейс;<br>– Містить простий та зрозумілий статистичний аналіз помилок. | – Немає можливості зробити тест і відправити його друзям;<br>– Має невелику базу вікторин. |
| Quiz Your Friends | – Тест можна зробити і відправити друзям<br>– Вікторини можна зберігати. | – Не містить поглибленого статистичного аналізу;<br>– Має не досить зручний інтерфейс. |
| DK Quiz | – Має загальний рейтинг з вашими друзями, яких ви можете підключити через соціальні мережі;<br>– Має більше 100 тестів, що охоплюють безліч тем. | – Нові оновлення не надають нові функції або функціональні можливості;<br>– Відсутність можливості ділитися результатами вікторини з друзями. |

## 1.3 Аналіз вимог до програмного модуля оцінювання якості контрольних завдань

Програмний модуль буде являти собою мобільний додаток, який повинен складатися з екранів логіну, створення тесту та тестових запитанньь, вибору тесту,

тестових питань і екрану виводу результату. Платформа буде являти собою клієнт-серверне рішення виду мобільний додаток-хмарне сховище, для роботи якого доступ в інтернет буде вимагатися тільки для синхронізації і передачі даних.

Можна визначити наступний набір функціональних вимог до програмного модуля:

- повинно бути забезпечено реєстрацію та авторизацію користувача в додатку;
- додаток повинен забезпечувати створення тесту та тестових запитань;
- додаток повинен дозволити користувачеві редагувати дані тестів;
- реалізувати сторінки тесту з вибором однієї правильної відповіді;
- реалізувати сторінки тесту з вибором декільком правильних відповідей;
- реалізувати сторінки тесту з співставленням термінів до тверджень;
- реалізувати індикатора статусу проходження тесту;
- відображення результату тестування;
- відображення інформації по темам, по яким були допущені помилки.

Нефункціональні вимоги до програмного модуля:

- додаток повинен бути доступний на пристроях з операційною системою Android;
- додаток повинен зберігати анонімні дані користувачів на зовнішньому сервері з базою даних.

### 1.4 Висновок до розділу 1

В даному розділі проаналізовано предметну область, були визначені функціональні і нефункціональні вимоги до додатка оцінювання якості контрольних завдань. Виконано порівняння існуючих додатків оцінювання якості контрольних завдань. В ході аналізу об’єкту проектування було визначено вимоги до програмного модуля оцінювання якості контрольних завдань.

В результаті аналізу систем-аналогів, таких як QuizUp, Quiz Your Friends та DK Quiz, було розглянуто вказані системи та визначено їх недоліки.

# 2 РОЗРОБКА ПРОГРАМНОГО МОДУЛЯ ОЦІНЮВАННЯ ЯКОСТІ КОНТРОЛЬНИХ ЗАВДАНЬ

## 2.1 Аналіз особливостей класифікації мобільних додатків

Перед тим, як починати розробку мобільних додатків, слід чітко визначитися з головними функціями майбутньої розробки. Від того, які функції має виконувати мобільний додаток, залежить тривалість його розробки, рівень складності і, звичайно, ціна. Загалом визначають чотири основні типи мобільних додатків:

1. Корпоративні. Їх мета – спрощення роботи компанії, швидка передача даних між працівниками або отримання корпоративної інформації. Цільовою аудиторією таких мобільних додатків є, насамперед, працівники компанії, а також реальні та потенційні клієнти та партнери. Розробка мобільних додатків такого типу простіша, для їх дизайну використовуються кольори та елементи корпоративного стилю компанії.

2. Контентні. Це мобільні додатки, основна мета яких надавати різного роду інформацію (текстову або у відео чи аудіоформаті). Головним чином, така розробка мобільних додатків здійснюється для засобів масової інформації, радіо, телеканалів чи порталів.

3. Сервісні. Їх мета випливає з назви, а саме – надавати певні сервісні послуги, тобто виконувати завдання, які ставить користувач, у режимі реального часу. Розробка мобільних додатків такого типу є складною, адже вони повинні працювати, як годинник: починаючи від калькулятора або будильника і закінчуючи програмами для роботи з великими обсягами тексту або графіки.

4. Ігрові. Основне завдання таких мобільних додатків – розважати, але це не означає, що воно єдине. Отже, розробка мобільних додатків такого типу ускладняється необхідністю вдало розмістити контекстну або пряму рекламу.

## 2.2 Аналіз підходів до розробки мобільних додатків

На даний момент існує три методи розробки мобільних додатків [8]:

- розробка нативних додатків;

- розробка кросплатформених або гібридних додатків;

- розробка веб-додатків.

Розглянемо преваги та недоліки методів розробки мобільних додатків.

Нативні додатки [9] завантажуються в App Store [10] або в Google Play [11], а потім встановлюються безпосередньо на пристрій. Оскільки користувальницький інтерфейс розроблений для конкретної ОС, вони забезпечують максимальну зручність використання програми. Якщо цільова аудиторія розділена між користувачами пристроїв Android [12] і iOS, потрібно буде створити два окремих продукту.

Переваги:

- швидкий, чуйний і зручний інтерфейс, який відповідає стандартам платформи;

- високий рівень безпеки;

- повноцінний призначений для користувача інтерфейс;

- раціональне використання пам'яті пристрою завдяки ефективному розвитку;

- доступно в App Store і Google Play;

- можливість працювати без доступу в інтернет;

- легко збирає призначені для користувача дані для оцінки продуктивності і поліпшення продукту.

Недоліки:

- трудомістка і дорога розробка. Необхідно створити два окремих додатки з різними базами коду для Android і iOS [13];

- потрібне схвалення з боку App Store і Google Play;

- у кожного продукту свій цикл випуску і оновлень;

- оновлення будуть вимагатися регулярно, щоб гарантувати актуальність та сумісність мобільного додатка з новими версіями ОС.

Гібридні додатки [14] створені з використанням таких технологій, як JavaScript, React, Dart, HTML, CSS, і підтримуються гібридним середовищем розробки. Використання спеціальних плагінів дозволяє отримати доступ практично до всіх функцій нативной платформи.

Ці додатки також встановлюються на пристрій користувача і доступні в магазинах додатків. Основна відмінність полягає в тому, що гібридні додатки повинні працювати через WebView [15] і розміщуватися у власному додатку, яке служить зовнішньою оболонкою.

Одним з ключових переваг гібридних додатків є використання єдиної бази коду для всіх платформ.

Переваги:

- можливість використання нативного функціоналу пристрою;

- простіший процес випуску оновлення;

- можливість роботи без підключення до Інтернету;

- узгодженість дизайну та навігації для різних платформ. Спрощує навігацію в додатку для користувачів на будь-якому пристрої.

Недоліки:

- потенційні проблеми з продуктивністю;

- можливість виникнення проблем з анімованими елементами;

- неможливо реалізувати складні функції;

- гібридне середовище розробки не є гнучким, що ускладнює адаптацію до останніх оновлень;

- потенційні проблеми інтеграції (налаштування сповіщень, налаштування сховища, тощо).

Веб-додаток - це програма, яка використовує веб-технології та працює в браузері [16]. По суті, це веб-сайт, який виглядає як додаток і виконує завдання через Інтернет.

Такі продукти не мають широкі функціональні можливості власних додатків, оскільки вони не встановлюються безпосередньо на пристрій і не використовують його апаратне забезпечення. При цьому вони можуть працювати на будь-якому

пристрої: ноутбуці, планшеті, розумних годиннику або навіть дисплеї розумного холодильника. Головна вимога до будь-якого пристрою - наявність браузера.

Переваги:

- швидка розробка додатків з використанням таких технологій, як CSS, HTML, JavaScript;

- не потрібно схвалення в App Store або Google Play. Якщо ви порівняєте веб-додатки з мобільними додатками, перевага першого полягає в тому, що після створення продукт відразу ж готовий до використання;

- сумісність з будь-якою платформою;

- миттєві оновлення, тому у користувачів завжди є доступ до останньої версії продукту;

- економія пам'яті пристрою. Веб-додатки завантажуються в ваш браузер, а не встановлюються на пристрій.

Недоліки:

- працює лише з Інтернетом, не може використовуватися, якщо немає мережі;

- недоступний в App Store або Google Play, що ускладнює пошук, зменшує кількість потенційних користувачів та збільшує витрати на рекламу;

- відсутність безпеки через роботу в браузері;

- несумісний зі старими версіями браузера;

- обмежений доступ до власних функцій платформи: камери, мікрофона, контактів, тощо.

## 2.3 Реалізація функціональних можливостей програмного модуля оцінювання якості контрольних завдань

Перед тим як почати розробку програмного модуля оцінювання якості контрольних завданьзнань, потрібно визначити які функціональні можливості він має реалізовувати, які вхідні і вихідні дані мають бути, тощо [26]. Для наглядної демонстрації функціональних можливостей програмного модуля створена діаграма прецедентів інформаційної технології (рис. 2.1)

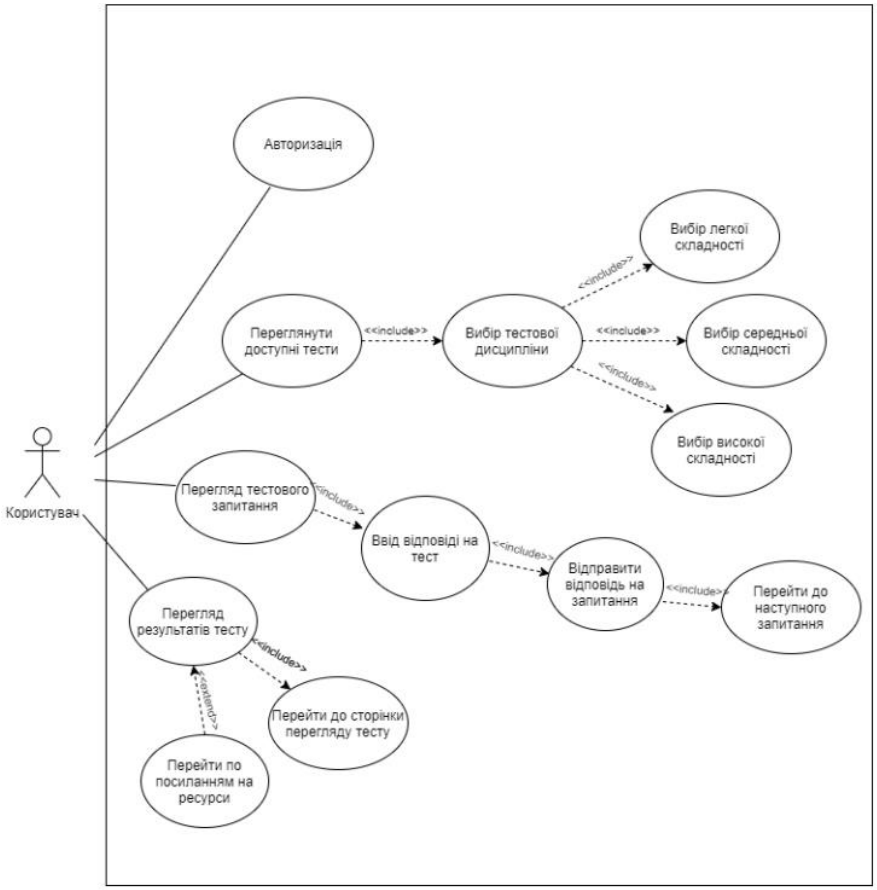


Рисунок 2.1 — Діаграма прецедентів програмного модуля оцінювання якості контрольних завдань

Додаток можна розділити на чотири типи екранів: сторінки авторизації, вибору тесту, питання і перегляду результатів. Сторінка авторизації може представляти з себе екран з двома полями і кнопкою відправки даних. Після вводу необхідних даних відбувається перехід на сторінку вибору тесту. Сторінка вибору тесту це список тестів у вигляді карточок по доступним категоріям. Інформація при них приходить від сервера, у відповідь на відповідний запит. Кожен тест має 3 доступних рівні складності.

Інформація що приходить від сервера впливає на типи питань і самі питання, які зустрічаються під час тестування. Після вибору користувача посилається запит на генерацію тесту. Якщо операція успішна, сервер повертає інформацію про ідентифікатор загенерованого тесту, який потім використовується для запитів на тестові питання, і клієнт переходить на відповідний екран з ініціацією запиту на перше питання. У разі невдачі, сервер повідомляє про це застосунок, і користувачу

виводиться відповідне повідомлення про це.

Інтерфейс сторінки тесту може відрізнятися в залежності від типу тесту, але деякі атрибути можуть залишатися незмінними. Серед них панель статусу тесту, яка представляє з себе деяку кількість індикаторів, вишикуваних в один ряд. Якщо пункт котрий користувач обрав засвітився зеленим, на питання було дано правильну відповідь, якщо червоним, то відповідно неправильну.

## 2.4 Розробка алгоритму роботи програмного модуля оцінювання якості контрольних завдань

Загальний алгоритм роботи програмного модуля оцінювання якості контрольних завдань наведено на рисунку 2.2.

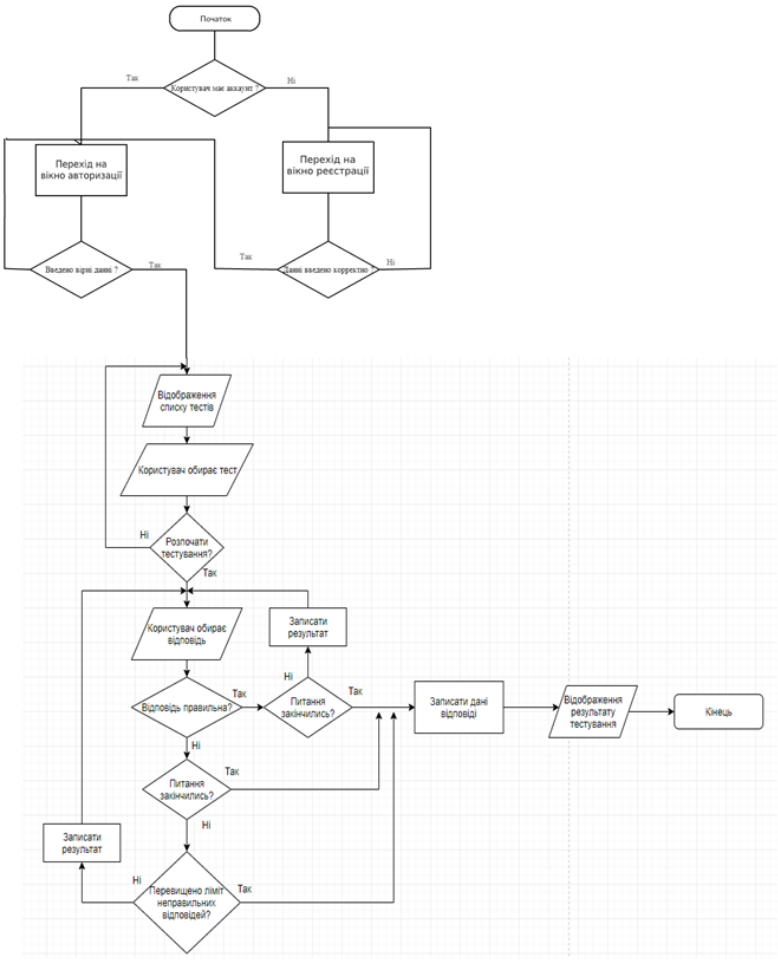


Рисунок 2.2 – Загальний алгоритм роботи програмного модуля оцінювання якості контрольних завдань

Проект складається з окремих структурних елементів і знаходиться в директорії lib (рис. 2.3)

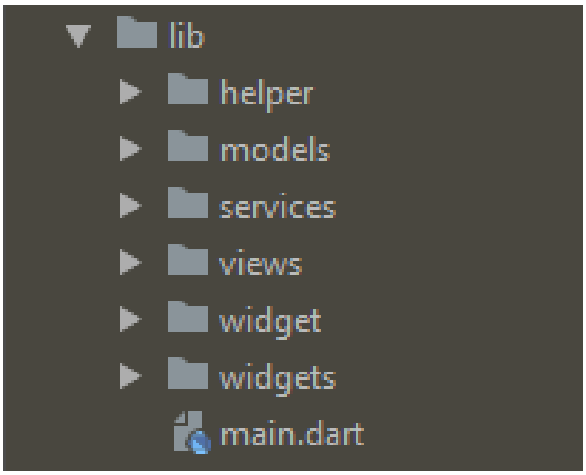


Рисунок 2.3 — Структура файлів проекту

Кожен модуль програми розміщений в своїй директорії, відповідно до його функціонального призначення:

helper – модулі побудовані відповідно до архітектури BLoC, що відповідають за просте і передбачуване управління станом окремих частин програми.

widgets – описи класів виключних ситуацій, які використовуються в застосунку.

models – класи сутностей, якими оперує застосунок. Також існує спосіб створення об’єктів за допомогою JSON.

services– відповідають за ініціацію запитів до серверу. Містять всі необхідні методи для будь-яких запитів.

views – віджети які знаходяться в корені дерева кожної з сторінок. Тут проходить основна взаємодія з bloc і описується користувацький інтерфейс.

main – вхідна точка як для Dart, так і для Flutter застосунків.

Під час розробки програмного модуля оцінювання якості контрольних завдань була спроектована і реалізована така схема системи мобільного застосунку (рис. 2.4).

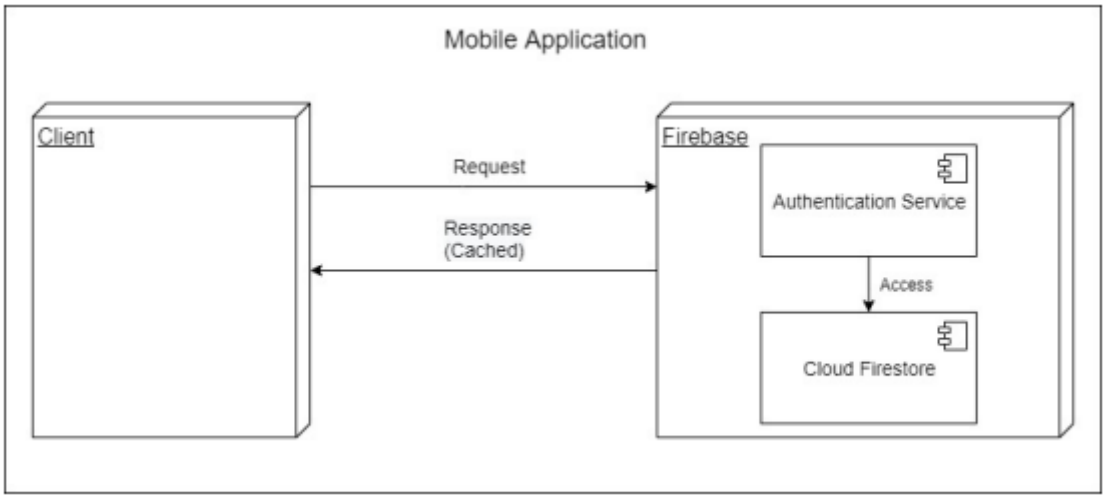


Рисунок 2.4 – Принцип роботи клієнт-серверної частини додатка

Клієнт-серверний застосунок, при використані котрого користувач отримує дані з одного із сервісів Firebase, назва якого Cloud Firestore [19] системи та технології системи та технології системи та технології системи та технології системи та технології системи та технології].

Користувач може отримати доступ до Cloud Firestore, тільки якщо його дані провалідовані правилами сервісу, в яких зазначено, що тільки клієнт отримавший дозвіл Authentication Service, може зчитувати та змінювати дані, що пов’язані з його обліковим записом. При відсутності інтернет підключення, застосунок використовує кеш дані, що зберігаються в пам’яті мобільного пристрою. Всі запити на зміни, що виникають при відсутності інтернет підключення зберігаються в кеш пам’яті, і потім, при його появі посилаються до сервісу Firebase.

## 2.5 Розробка UML-діаграми класів програмного модуля оцінювання якості контрольних завдань

З точки зору мови програмування клас об'єктів можна розглядати як тип даного, а окремий об'єкт - як змінну цього типу. Визначення програмістом власних класів об'єктів для конкретного набору задач повинно дозволити описувати окремі завдання в термінах самого класу завдань (при відповідному виборі імен типів та імен об'єктів, їх параметрів і виконуваних дій). Таким чином, об'єктно-орієнтований

підхід передбачає, що при розробці програми повинні бути визначені класи які у програмі об'єктів і побудовано їх опису, потім створені екземпляри цих об'єктів і визначено взаємодію між ними [17].

Під час проектування мобільного додатку було розроблено діаграму класів (рис. 2.5).

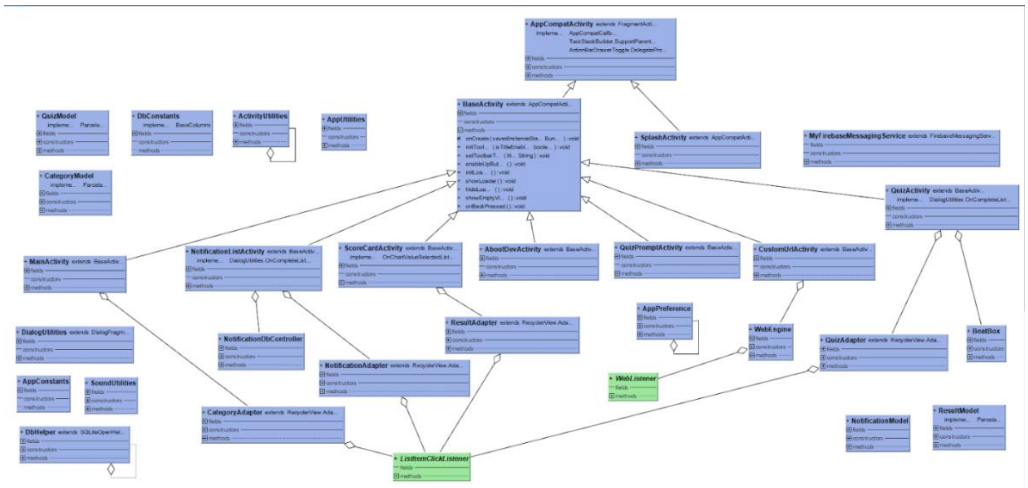

Рисунок 2.5 – Діаграма класів додатку оцінювання якості контрольних завдань

Activity є класом, який по суті представляє окремий екран (сторінку) додатків або їхніх візуальний інтерфейс. Окремі activity, які вже безпосередньо використовуються в додатку, є спадкоємцями цього класу. Додаток може мати одну activity, а може і кілька. Кожна окрема activity задає окреме вікно для відображення.

На діаграмі класів можна побачити що були розроблені такі Activity:

- AboutDevActivity;
- BaseActivity;
- CustomUrlActivity;
- MainActivity;
- NotificatinListActivity;
- QuizActivity;
- QuizPromtActivity;
- ScoreCardActivity;
- SplashActivity.

Кожен Activity наслідується від базового BaseActivity

## 2.6 Висновок до розділу 2

В даному розділі було виконано аналіз особливостей класифікації мобільних додатків. Описано переваги та недоліки популярних програмних засобів у своїх категоріях. У розділі було визначено які функціональні можливості має реалізовувати програмний модуль оцінювання якості контрольних завдань.

В ході дослідження було розроблено алгоритм роботи програмного засобу оцінювання якості завдань. Обгрунтовано використання клієнт-серверної технології для реалізації програмного модуля оцінювання якості контрольних завдань

Розроблено UML-діаграму класів програмного модуля оцінювання якості контрольних завдань

# 3 ПРОГРАМНА РЕАЛІЗАЦІЯ МОДУЛЯ ОЦІНЮВАННЯ ЯКОСТІ КОНТРОЛЬНИХ ЗАВДАНЬ

## 3.1 Обґрунтування вибору мови програмування та середовища розробки

Створення програмного забезпечення здійснюється за допомогою використання інтегрованих середовищ розробки (IDE). Інтегроване середовище розробки — комплексне програмне рішення для розробки програмного забезпечення. Зазвичай, складається з редактора початкового коду, інструментів для автоматизації складання та відлагодження програм. Більшість сучасних середовищ розробки мають можливість автодоповнення коду.

Деякі середовища розробки містять компілятор, інтерпретатор або ж обидва (наприклад NetBeans та Eclipse), інші не містять жодного з них (SharpDevelop та Lazarus). В IDE автоматизований процес компіляції, збірки та запуску додатків, що значно спрощує роботу розробнику програмного забезпечення і довзволяє розробнику початківцю без значних зусиль створювати свої перші додатки. Інтегроване середовище розробки зазвичай має декілька основних влаастивостей які спрощують розробку програмного забезпечення. Підсвічення синтаксису та нумерація рядків значним чином допомогають зоєрінтуватися в програмного коді. Майже кожне IDE має автоматичний відладник додатків. Також дуже важливою є інтеграція з системою контролю версій. Ця функція використовується, коли над проектом працює не один розробник а ціла команда розробників. Ці системи дозволяють на відстані розробникам один і той самий код разом і не переписувати правки один одного.

Для створення програми на базі операційної системи Android, можуть бути обрані різні середовища розробки, такі як, Eclipse, IntelliJ IDEA, Android Studio і т.д.

1. Eclipse - це безкоштовна середовище розробки від некомерційної організації Eclipse Foundation. По суті справи, сама програма - це основа, до якої підключаються різні модулі [13]. Наприклад, Java Development Tools (Для створення додатків на Java), C / C ++ Development Tools (для розробки програм на мові C або

C++) і т. д. Завдяки активному розвитку, а також підтримки з боку компанії і сторонніх розробників, на даний момент можна виділити наступні переваги середовища Eclipse:

- відмінна продуктивність на слабких машинах;

- велике число доповнень (наприклад, для роботи з сервером, базою даних і т.д.);

- можливість підключення модулів;

- можливість групової розробки.

Середовище розробки Eclipse мала велику популярність кілька років тому і вважалася монополістом на ринку IDE для Android. Однак у зв'язку з виходом Android Studio, компанія Google перестала підтримувати Eclipse як основну середовище для розробки додатків під Android.

2. IntelliJ IDEA. Розробкою даного середовища програмування займається компанія JetBrains [20]. Як і Eclipse, це середовище розробки дає можливість створювати програми на декількох мовах програмування. Головний недолік - це наявність платної версії.

3. Android Studio - це інтегроване середовище розробки (IDE) для роботи з платформою Android, анонсована 16 травня 2013 року на конференції Google I / O. На даний момент Android Studio – це офіційна середовище розробки під Android. Середовище Android Studio, засноване на програмному забезпеченні IntelliJ IDEA від компанії JetBrains і є офіційним засобом розробки Android-додатків [24]. Дане середовище розробки є кроссплатформене і доступне на операційних системах Windows, OS X і Linux. даною середовищем підтримується офіційна мова програмування для розробки додатків для платформи Android - Java, до того ж 17 травня 2017 року на Google IO було оголошено про підтримку мови Kotlin в Android Studio 3.0, який заслужив багато похвальних відгуків від досвідчених розробників.

Android SDK є безкомпромісно повним середовищем розробки мобільних додатків для пристроїв з ОС Android [11].

Кожен раз, коли Google випускає нову версію Android або оновлення, також випускається відповідний SDK, який розробники повинні завантажити та

встановити, він включає в собі всі інструменти, необхідні для кодування програм з нуля і навіть для їх тестування.

Інтерфейс Android Studio представляє собою панель інструментів в верхньої частини екрану, дерево проекту і консоль для виведення. Розробнику доступний текстовий редактор для роботи з програмним кодом, а для побудови графічного інтерфейсу додатку є можливість використовувати конструктор інтерфейсу. Файли, що відповідають за візуальне відображення компонентів, є XML-файли, які часто зручно редагувати вручну, тому робота з ними доступна так само і в текстовому режимі. При редагуванні файлів інтерфейсу в режимі конструктора, праворуч стає доступна панель, що відображає властивості обраних об'єктів і їх значення, а також ієрархію і приналежність об'єктів.

## 3.2 Аналіз засобів розробки програмного модуля оцінювання якості контрольних завдань

Android SDK - це набір інструментів, бібліотек та універсальний засіб для розробки мобільних додатків операційної системи Android. Відмінною рисою від звичайних редакторів для написання кодів є наявність широких функціональних можливостей, що дозволяють запускати тестування і налагодження вихідних кодів, оцінювати роботу програми в режимі сумісності з різними версіями ОС Android і спостерігати результат в реальному часі. Підтримує велику кількість мобільних пристроїв, серед яких виділяють: мобільні телефони, планшетні комп'ютери, розумні окуляри, сучасні автомобілі з бортовими комп'ютерами на ОС Android, телевізори з розширеним функціоналом, особливі види наручних годинників і багато інших мобільних гаджетів, габаритні технічні пристосування.

До складу SDK включені різні засоби розробки, в тому числі відладчик, набір бібліотек, телефонний емулятор [28] на базі двигуна QEMU, набір документації, прикладів додатків і посібників. Середовище Android SDK може бути запущена на комп'ютерах, що використовують ОС Linux, Mac OS X 10.5.8 і новіше, Windows 7 і новіше.

Кожний набір засобів розробки додатків унікальний [25], щоб краще зрозуміти це і вибрати той, який підходить найкраще, потрібно проаналізувати ці засоби розробки, на рисунку 3.1 відображений перелік програмних рішень для розробки додатків з відкритим вихідним кодом.

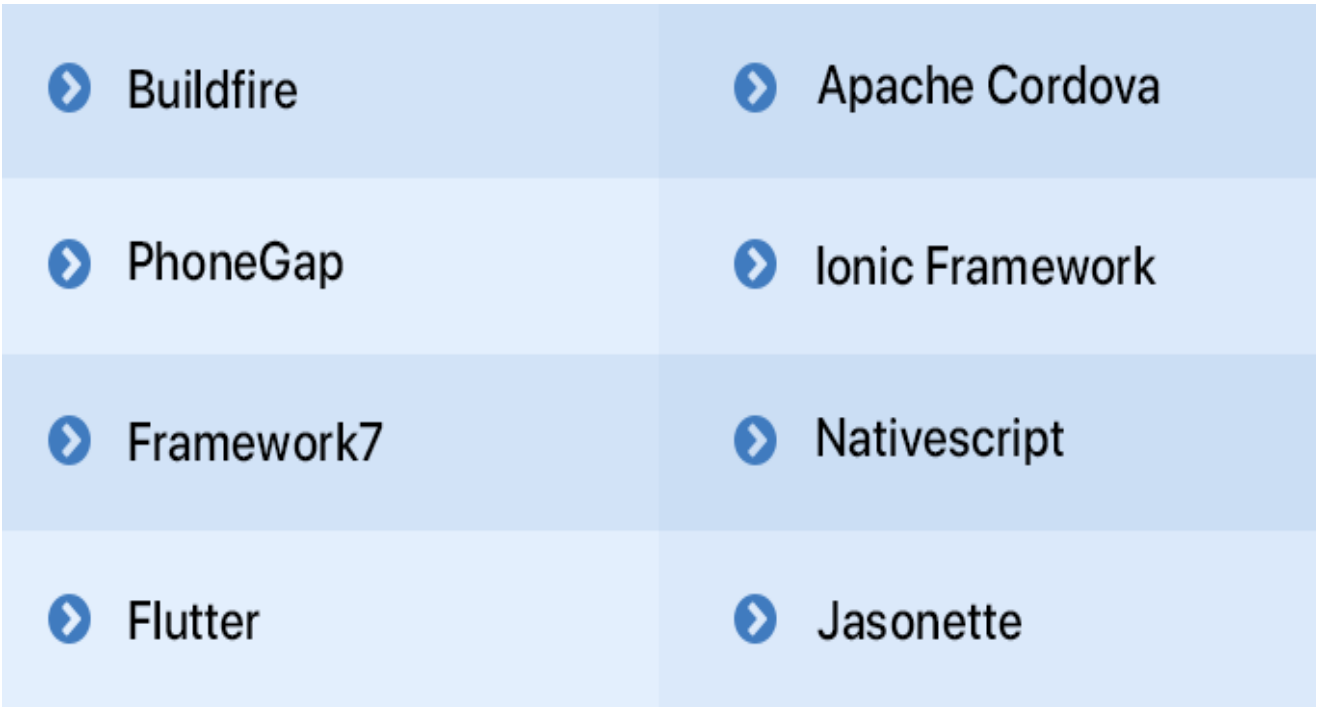


Рисунок 3.1 - Програмні рішення для розробки додатків з відкритим вихідним кодом

На таблиці 3.1 зображено порівняння програмних рішень для розробки додатків з відкритим вихідним кодом, розглянемо кожне з цих рішень для того щоб визначити яке з них буде оптимальним для розробки програмного модуля оцінювання якості контрольних завдань.

Buildfire - це повністю гнучкий до налаштування, безкоштовний, простий та інтуїтивно зрозумілий засіб для створення додатків для Android, iOS та прогресивних веб-програм [10]. Програмне забезпечення забезпечує елегантний інтерфейс (рис. 3.2) з функцією перетягування «drag-and-drop UI», і ця платформа підтримує більше 10 000 програм.

Основні особливості платформи Buildfire.

Buildfire - це спеціальне програмне забезпечення для розробки додатків як для досвідчених розробників додатків так і для початківців.

Це полегшує функцію блокування BuildFire для забезпечення безпеки та відповідності програм.

Додаток обмежує доступ користувачів до вибраного вмісту.

Функція BuildFire Connect забезпечує надійний досвід, інтегруючи або розширюючи будь-який елемент у вашому додатку.

Додаток можна поєднувати з будь-якими сторонніми API або заздалегідь інтегрованими інтеграціями.

Таблиця 3.1 - Порівняння програмних рішень для розробки додатків

| Програмні рішення для розробки додатків | Підтримувані пристрої | Розгортання програмного забезпечення | Особливі можливості |
|---|---|---|---|
| Buildfire | Windows, Mac, Linux, Web-based | Open API, On-Premise | • Контроль доступу / дозволів<br>• Інструменти для спільної роботи<br>• Тестування сумісності<br>• Моделювання даних<br>• Звітність та аналітика<br>• Управління версіями<br>• Розробка веб-додатків |
| Apache Cordova | Windows, Mac, Linux | Open API | • Контроль доступу / дозволів<br>• Інструменти для спільної роботи<br>• Управління розгортанням програмного забезпечення<br>• Звітність та аналітика<br>• Тестування сумісності<br>• Управління версіями |

Таблиця 3.1 - Продовження

| | | | |
|---|---|---|---|
| PhoneGap | Windows, Mac, Linux | Open API, Cloud Hosted | • Контроль доступу / дозволів<br>• Сумісне тестування<br>• Управління розгортанням забезпечення<br>• Розробка веб-додатків<br>• Управління версіями |
| Ionic Framework | Windows, Mac, Linux | Open API, Cloud Hosted | • Контроль доступу / дозволів<br>• Інструменти співпраці<br>• Моделювання даних<br>• Звітування та аналітика<br>• Розробка програмного забезпечення |
| Framework7 | Windows, Mac, Linux | Open API | • Контроль доступу / дозволів<br>• Рефакторинг коду<br>• Розробка без коду<br>• Інструменти співпраці<br>• Налагодження програми<br>• Розробка програмного забезпечення<br>• Розробка веб-додатків |
| NativeScript | Windows, Mac, Linux | Open API, Cloud Hosted | • Контроль доступу / дозволів<br>• Рефакторинг коду<br>• Розробка без коду<br>• Тестування сумісності<br>• Розробка програмного забезпечення<br>• Управління версіями<br>• Розробка веб-додатків |
| Jasonette | Windows, Mac, Linux | Open API | • Контроль доступу / дозволів<br>• Рефакторинг коду<br>• Розробка без коду<br>• Інструменти співпраці<br>• Налагодження програми<br>• Управління версіями |

Таблиці 3.1 - Продовження

| Програмні рішення для розробки додатків | Підтримувані пристрої | Розгортання програмного забезпечення | Особливі можливості |
|---|---|---|---|
| Flutter | Windows, Mac, Linux | Open API, Cloud Hosted | • Контроль доступу / дозволів<br>• Рефакторинг коду<br>• Інструменти співпраці<br>• Тестування сумісності<br>• Налагодження програми<br>• Управління розгортанням<br>• Управління версіями<br>• Розробка веб-додатків<br>• Сторінки Landng / веб-форми |

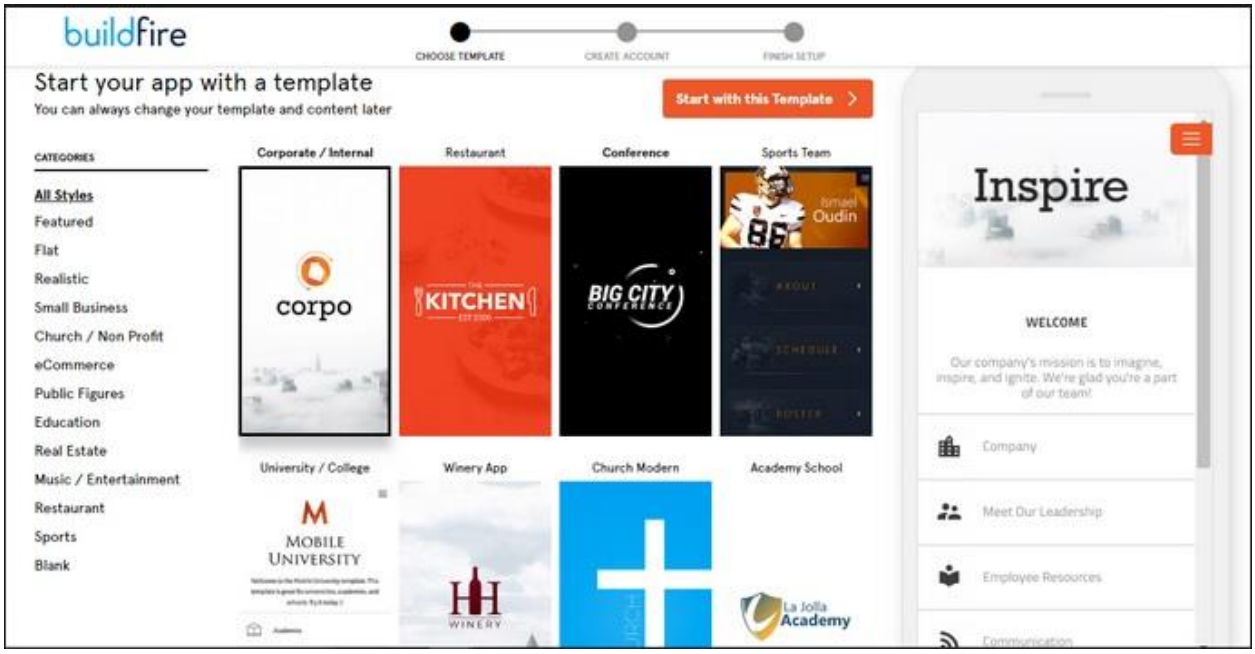


Рисунок 3.2 - Інтерфейс Buildfire

Apache Cordova - мобільне середовище розробки додатків, спочатку розроблена Nitobi, дозволяє програмістам створювати додатки для мобільних пристроїв за допомогою CSS, HTML5 і JavaScript, замість того, щоб використовувати конкретні платформи API, такі як Android, IOS або Windows Phone [11]. Це забезпечується за рахунок перетворення з CSS, HTML і JavaScript в код,

який будь-яка платформа сприймає як елемент web. Це розширює можливості HTML і JavaScript для роботи з різними пристроями. В результаті застосування є гібридними, вся генерація макета здійснюються за допомогою web-view замість основної структури призначеної для користувача інтерфейсу платформи це відображено на рисунку 3.3 - який представляє принцип роботи додатку. Apache Cordova має доступ до API базового функціоналу пристрою, такого як файлова система, камера і інше. Це програмне забезпечення з відкритим вихідним кодом використовується в інших програмах, таких як Appery.io або Intel XDK.

Основні особливості платформи Buildfire.

Apache Cordova спрощує створення мобільних додатків з використанням HTML, CSS і JS і орієнтований на кілька платформ за допомогою однієї бази коду.

Програмне середовище може бути розширена за рахунок власних модулів, щоб розробники могли додавати в додаток додаткові функції.

Інтерфейс командного рядка дозволяє користувачам створювати нові проекти, будувати їх на різних платформах і запускати на різних пристроях або в емуляторах.

Набір основних компонентів Cordova використовується для створення основи додатки, щоб можна було витратити достатньо часу на реалізацію завдання розробки.

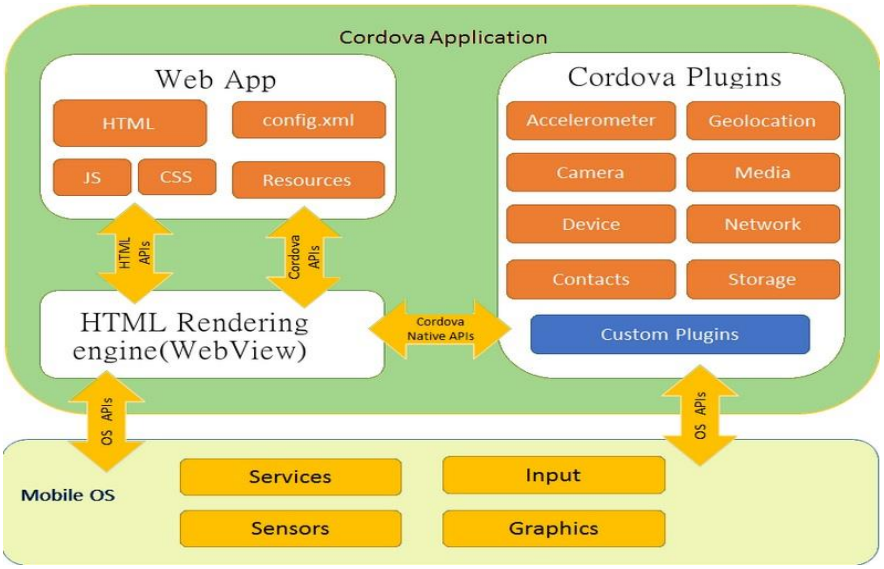


Рисунок 3.3 - принцип роботи додатку Apache Cordovав

PhoneGap - безкоштовний open-source фреймворк для створення мобільних додатків [12], створений Nitobi Software інтерфейс якого зображено на рисунку 2.4. Цей фреймворк дозволяє створити додатки для мобільних пристроїв використовуючи JavaScript, HTML5 та CSS3, без необхідності знання нативних мов програмування (наприклад, Objective-C), під всі мобільні операційні системи такі як iOS, Android, і т.п. Готовий додаток компілюється у вигляді встановлюваних пакетів для кожної мобільної операційної системи.

Основні особливості платформи PhoneGap.

PhoneGap – це засіб створення кроссплатформенних додатків за допомогою PhoneGap включає в себе продуктивність створення додатків на платформах iOS, Android, WebOS, Tizen.

Економічне програмне забезпечення, що забезпечує кращий доступ до власних API.

Забезпечує надійну бекенд-підтримку і гнучкість в розробці.

Бібліотеки призначеного для користувача інтерфейсу в PhoneGap допомагають поліпшити користувальницький інтерфейс в системі.

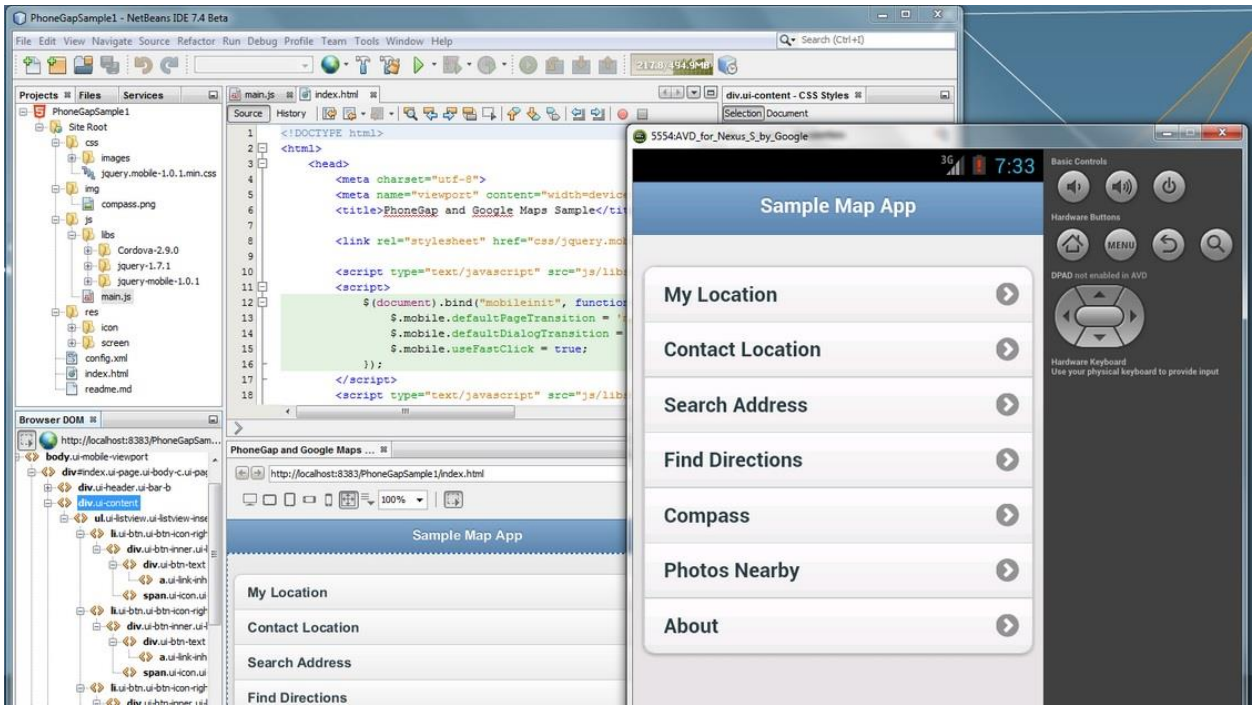


Рисунок 3.4 - Інтерфейс PhoneGap

Ionic Framework - це SDK з відкритим кодом для розробки гібридних

мобільних додатків, створений у 2013 році. Оригінальна версія була випущена в 2013 році і побудована на базі AngularJS та Apache Cordova. Однак останній випуск був перероблений як набір веб-компонентів, що дозволило розробнику вибрати будь-яку структуру інтерфейсу користувача, наприклад Angular, React або Vue.js, приклад вихідного коду інтерфейсу користувача та його відображення зображено на рисунку 3.5. Цей фреймворк також дозволяє використовувати іонічні компоненти, які взагалі не мають інтерфейсу користувача [13]. Ionic надає інструменти та послуги для розробки гібридних мобільних, настільних та прогресивних веб-програм на основі сучасних технологій та практик веб-розробки, використовуючи веб-технології, такі як CSS, HTML5 та Saas [19]. Мобільні програми можна створювати за допомогою цих веб-технологій, а потім розповсюджувати їх у власних магазинах додатків, щоб встановлювати їх на пристрої, використовуючи Cordova або Capacitor.

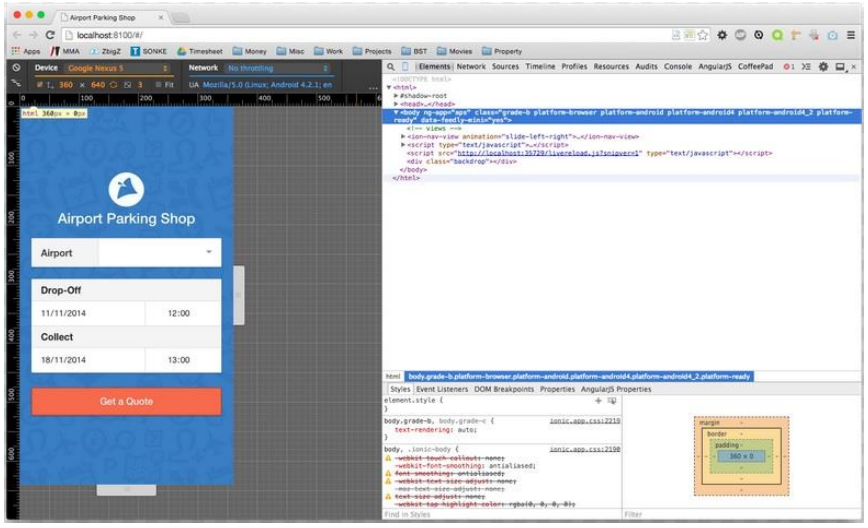


Рисунок 3.5 – Інтерфейс вихідного коду інтерфейсу користувача та його зображення яке розроблене за допомогою Ionic Framework

Основні особливості платформи Ionic Framework:

Ionic - це модуль "npm", і для в становлення потрібен Node.js.

Додатки Ionic працюють із поєднанням веб-коду та власного коду, і вони забезпечують повний доступ до власних функціональних можливостей за допомогою інтерфейсу програми.

Ionic дозволяє створювати додатки на базі Кордови та розгортати їх за допомогою спрощеного інструменту командного рядка „Ionic”.

Програмне забезпечення включає інтерактивні парадигми, типографіку, мобільні компоненти та розширювану базову тему.

Framework7 - це безкоштовна платформа з відкритим вихідним кодом для мобільного HTML, інтерфейс якого зображено на рисунку 3.6. Він використовується для розробки гібридних мобільних додатків або веб-додатків для пристроїв iOS і Android.

Основні особливості платформи Framework7 [14]:

Програмне забезпечення забезпечує зручне управління переглядом та підтримку навігації.

Вбудована бібліотека FastClick допомагає обробляти затримки кліків для інтерфейсу дотику.

Програмне забезпечення забезпечує простий і звичний синтаксис для користувачів, якщо вони знайомі з JQuery, щоб швидко розпочати роботу.

Це полегшує вбудовану систему макетної сітки та налаштування теми та кольорові схеми.

Гнучкий маршрутизатор допомагає завантажувати сторінки з шаблонів і забезпечує апаратно прискорену анімацію та переходи CSS.

NativeScript - це фреймворк з відкритим вихідним кодом, розроблений компанією Telerik, для розробки додатків на платформах Android та iOS, на рисунку 3.7 зображено інтерфейс мобільного додатку створеного за допомогою NativeScript [15]. Додатки на NativeScript розробляються на платформенезалежних мовах, таких як Javascript або TypeScrip. У NativeScript реалізована повна підтримка фреймворка Angular. Мобільні додатки, побудовані з NativeScript, мають повний доступ до платформ API, так як вони були розроблені в XCode або в Android Studio. Також розробники можуть включати у свої додатки сторонні бібліотеки з такими ресурсами, як Cocoapods, Android Arsenal, Maven та npm.js, без створення додаткових шарів.

Основні особливості платформи NativeScript:

Платформа NativeScript розміщена з багатим набором крос-платформних інструментів для швидкого створення додатків для iOS та Android.

Конструктор тем NativeScript візуально налаштовує вбудовані теми.

Програма підтримує надійну систему стилів на основі CSS.

Програмне забезпечення дозволяє користувачеві напряму обратись кожному API-інтерфейсу власної платформи з коду.

Програмне забезпечення використовує власні елементи управління, а його додатки можна використовувати з технологією читання з екрану.

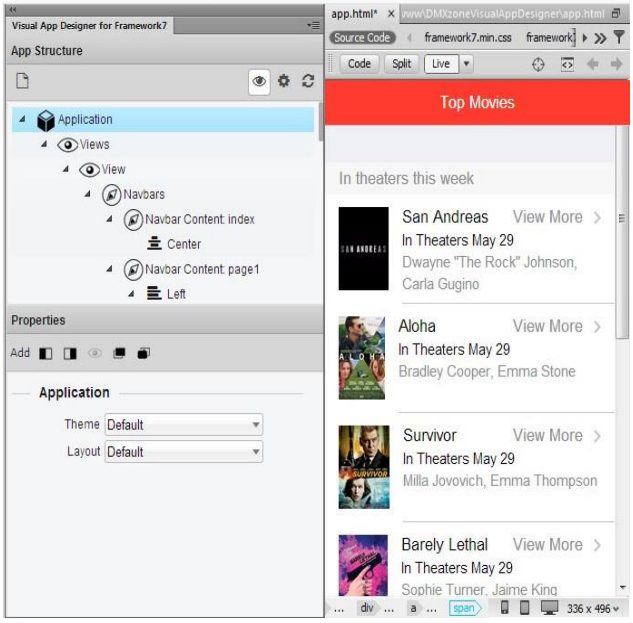


Рисунок 3.6 – Інтерфейс мобільного додатку створеного за допомогою Framework7

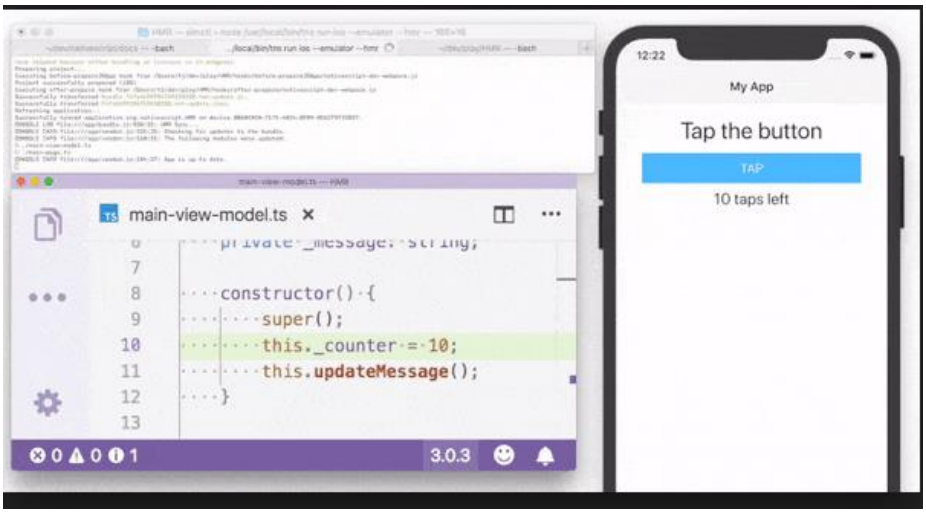


Рисунок 3.7 – Інтерфейс мобільного додатку створеного за допомогою NativeScript

Jasonette - це платформа програмування високого рівня, яка використовує розмітку на основі JSON для розробки додатків (рис. 3.8) для iOS та Android [36]. Основа програми – браузер який використовує JSON [34], він бере файли з сервера та перетворює їх у власні елементи.


Рисунок 3.8 – Інтерфейс мобільного додатку створеного за допомогою Jasonette

Основні особливості платформи Jasonette.

Програмне забезпечення Jasonette працює як веб-браузер та інтерпретує розмітку HTML на веб-сторінках.

Він розширюваний і може бути легко інтегрований з існуючим власним кодом.

Спрощує внесення змін до живе власний додаток всього за кілька хвилин.

Flutter - SDK з відкритим вихідним кодом для створення мобільних додатків від компанії Google [17]. Він використовується для розробки додатків (рис. 3.9) під операційні системи Android та iOS. Також нещодавно з’явилася підтримка Flutter Web для веб браузерів. Це означає наявність можливості використання одної мови програмування для декількох додатків.

Основні особливості платформи Flutter.

На відмінну від інших багатьох відомих сьогодні мобільних платформ, Flutter не використовує Javascript зовсім. В якості мови програмування для Flutter вибрали Dart, який компілюються в бінарний код, завдяки якому досягається швидкість

виконання операцій, яку можна порівняти з нативними технології.

Flutter не використовує нативні компоненти, тому нічого не потрібно розробляти для комунікації з ними. Замість цього інтерфейс будується як в ігрових програмах. Фон, текст, медіа-елементи, кнопки – весь рендер відбувається в Flutter. Навіть враховуючи це, «Hello World» застосунок займає зовсім не багато місця.

Для побудови інтерфейсу користувача використовується декларативний підхід, що схоже на шлях ReactJs, на основі віджетів. Для ще більшого приросту швидкості роботи інтерфейсу віджети малюються за необхідністю – тільки якщо в моделі відбулися зміни (подібно до того як відбувається у світу FrontEnd).

В фреймворк вбудований так званий Hot-reload, якого до сих пір не було в нативних платформах.

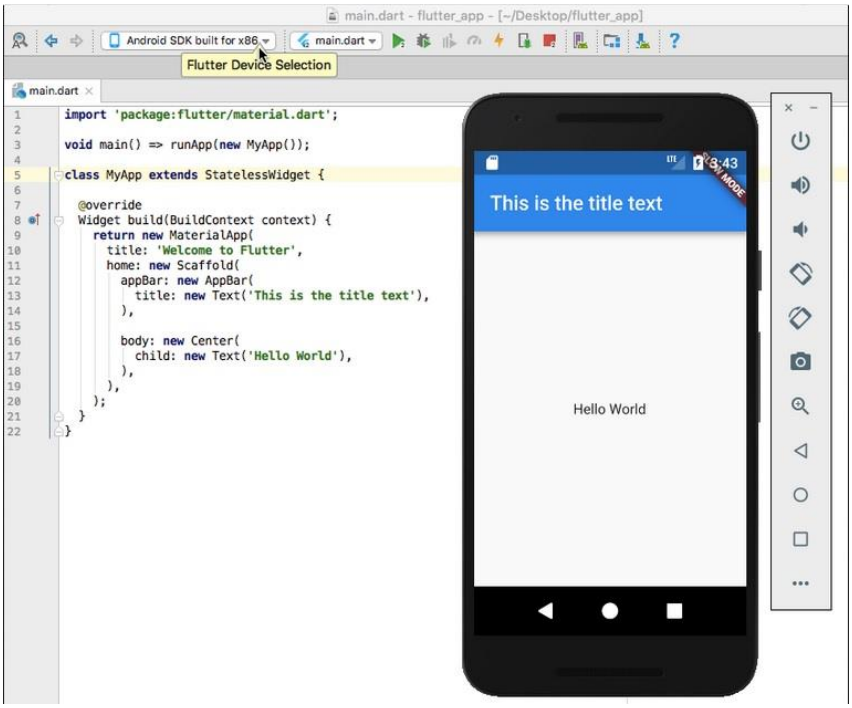


Рисунок 3.9 – Інтерфейс мобільного додатку створеного за допомогою Flutter

### 3.3 Аналіз засобів реалізації взаємодії з зовнішніми хмарним сервісом

BaaS – це модель, що дозволяє розробникам веб-додатків і мобільних програм зв'язати їх застосування з серверним хмарним сховищем і API, що виставляються серверними додатками, а також надає такі функції, як управління користувачами,

повідомлення оповіщень, інтеграція зі службами соціальних мереж [14].

Ідея BaaS в тому, що замість того, щоб самим розробляти і надалі підтримувати серверні сервіси, можна скористатися готовими наборами, які разом формують необхідний універсальний крос-платформний бекенд для будь-якого проекту. Відповідно, не потрібно взаємодіяти з сервером додатка, базою даних, клієнт-серверної бібліотекою, хостингом, необов'язково писати адміністративну панель, продумувати дизайн свого API.

Практичним рішенням вважається Firebase, який реалізує BaaS. Основний сервіс - хмарна СУБД класу NoSQL, що дозволяє розробникам додатків:

Зберігати і синхронізувати дані між декількома клієнтами (рис 3.10);

Підтримувати особливості інтеграції з додатками під операційні системи Android і iOS;

Управління налаштуваннями , створення колекцій.

Зберігання даних в хмарі від Google (рис 3.11), з можливістю локального зберігання даних.

Завдяки реалізованій гнучкій інтеграції засобів платформи Flutter і Firebase: з’являється можливість ефективно використовувати StreamBuilder: отримуючи елементи з бази даних, створюється список віджетів з якими можна взаємодіяти (сортувати / редагувати / видаляти).

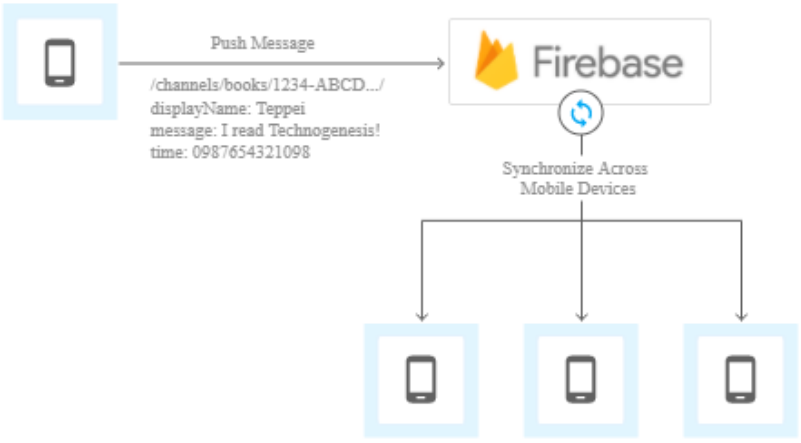


Рисунок 3.10 - Зв'язок між додатком і хмарним сховищем

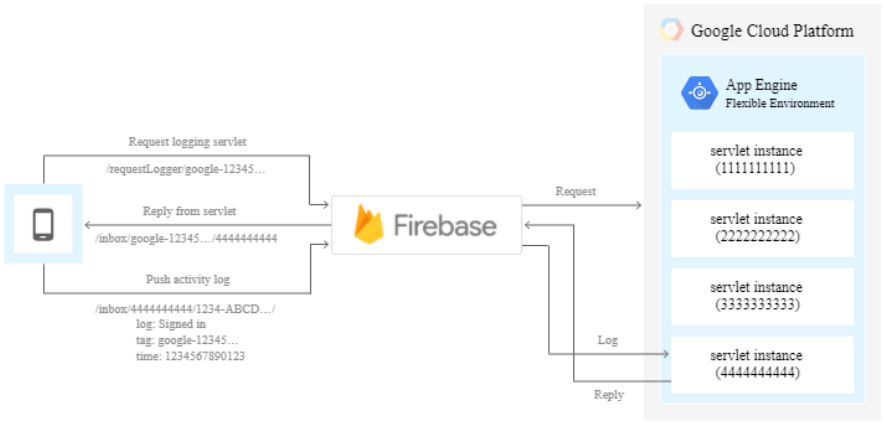


Рисунок 3.11 - Зв'язок між додатком і хмарним сховищем

### 3.4 Розробка засобів для генерації тестів

Сьогодні майже всі спроби запрограмувати формування тестових завдань все більше використовує штучний інтелект, але постає проблема формалізації знань та генерація тестів з її використанням.

Одним з перспективних і порівняно нескладних в реалізації є підхід параметризованих тестів. Суть підходу полягає у поданні різним користувачам шаблонного завдання, яке відрізнятиметься певними параметрами, які генеруються автоматично. Таким чином кожен користувач отримує індивідуальне завдання, а система по певній формулі чи алгоритму, підставляючи параметри отримує вірну відповідь для подальшої перевірки відповіді, введеної студентом. Недоліком підходу є його вузька предметна спрямованість. Так параметризовані тести добре підходять для організації контролю практичних навичок в точних науках, а також програмуванні, проте не підходять для перевірки теоретичних знань, а також контролю в гуманітарних науках.

Деякого поширення в алгоритмізації перевірки засвоєння навчального матеріалу отримав спосіб застосування семантичних мереж для побудови завдань. [18,19]. Основним елементом таких мереж є тріади [18-20]: сутність 1 – відношення – сутність 2. Тест створюється за допомогою опускання однієї з ланок тріади і заданням питання про ланку, яка є відсутньою. Плюсом такого способу є можливість використання знань з предметної області для створення завдань.

Мінусом ж те, що для складання повної семантичної мережі, яка могла б правильно відображати конкретну предметну область потрібні значні витрати. Іншою складністю є лінгвістична придатність та іноді, недоцільність генерованих тестових запитань. Це означає, що на основі семантичної мережі часто трапляються питання про предметну область, для відповіді на які потрібні знання особливостей, які не становлять будь-якої педагогічної цінності у освітньому контексті. Такі проблеми є типовими для класичних моделей знань з використання AI.

Понятійно-тезисна модель формалізації дидактичного тексту розробляється на стику багатьох наукових галузей, серед яких наступні: інженерія знань як напрямок штучного інтелекту; педагогіка, а саме її розділ – дидактика, що розкриває правила викладання; інтерпретація (герменевтика, екзегетика), що вивчає правила тлумачення текстів [6, 7]; лінгвістика і її розділ семантика, що вивчають закономірності природної мови і проблеми, пов’язані зі змістом, значенням, і інтерпретацією лексичних одиниць; інструментом реалізації служать технології розробки web-систем, тощо.

Основною сутністю в понятійно-тезисної моделі - це об’єкт про який в навчальному матеріалі містяться знання, які потрібні користувачу. Для кожної предметної області виділяються свої об’єкти.

Для подання знань використовуються інші структурні елементи – тези, які представляються собою відомість про поняття. Теза – це ознака, яка є істиною для конкретного поняття. З лінгвістичної точки зору, це речення або декілька речень, зяких видалені підмети. В сукупності наведені вище елементи складають з себе ПТ-елементи. Будь-яка теза відповідає лише одному поняттю. Поняття може відповідати довільній кількості тез

Центральним джерелом знань є навчальний матеріал. З нього за допомогою маніпуляцій виводяться конкретні семантичні одиниці. Навчальні матеріали для використання в дистанційній освіті діляться на невеликі частини для кращого розуміння студентами. Такі частини називаються «кадрами», що є дуже важливими для використання в понятійно-тезисної моделі, так як саме з ним виділяються семантичні елементи.

Елементи понятійно-тезисної моделі виводяться з тексту навчального матеріалу. Вони формуються способом осмисленого читання безпосереднього тексту з нескладними маніпуляціями. Викладач виділяє з тексту семантичні елементи і додає їх в базу даних. Кожен фрагмент може бути джерелом довільної кількості тез.

Маючи на увазі те, що тези стосуються лише одного фрагменту матеріалу, з якого вони були створенні, і те що поняття можуть відноситись до багатьох навчальних фрагментів, зв'язок між матеріалом і поняттями влаштовується використовуючи тези. Для створення різних типів завдань використовуються шаблони. Вони дозволяють доповнювати різні предметні області завданнями нових типів. Спосіб побудови тестових завдань на основі понятійно-тезисних елементів описується шаблонами. Функція додавання нових шаблонів надає можливість удосконалення системи не лише на етапі проектування, але й після її розгортання і введення в експлуатацію для налаштування алгоритму створення завдань для різних дисциплін.

### 3.5 Реалізація CRUD-функцій

CRUD-функції (Create, Read, Update, Delete) - це чотири базові функції для роботи з базами даних, що реалізують створення, читання, оновлення (редагування), видалення даних. На діаграмі прецедентів ці функції представлені у варіантах з ключовим словом «взаємодія». Нижче представлений приклад реалізації функції створення поля (рис. 3.12) в хмарному сховищі Firestore. Вхідні параметри: передані дані (data) і шлях до сховища (ім'я колекції, base).

```
addData(data, base) {
       Firestore.instance.runTransaction((Transaction crudTransaction) {
         CollectionReference reference =
            Firestore.instance.collection('user').document('date').colle-
tion(base);
         reference.add(data);
       });
  }
```

Рисунок 3.12 - Реалізація функції створення запису в базі даних

Асинхрона функція getData (рис. 3.13) реалізує отримання (читання) даних на запит користувача. Рухаючись параметр - ім'я колекції або певний документ колекції (base).

```
Future getData(base) async {
    return await Firestore.instance.collection('user').document('date').col-
lection(base).snapshots();
  }
```

Рисунок 3.13 - Реалізація функції читання даних

Функція поновлення даних (рис. 3.14) являє собою запит на зміна конкретного документа колекції за його ідентифікатором , новими даними (newValues) і ім'я колекції (base).

```
updateData(docId, newValues, base) {
    Firestore.instance
        .collection('user').document('date')
        .collection(base).document(docId).updateData(newValues)
        .catchError((e) { print(e); });
  }
```

Рисунок 3.14 - Реалізація функції створення запису в базі даних

Функція видалення даних (рис. 3.15) використовує переданий ідентифікатор документа колекції, для створення запиту на видалення цього документа в Firestore.

```
deleteData(docId, base) {
  Firestore.instance
      .collection('user')
      .document('date')
      .collection(base)
      .document(docId)
      .delete()
      .catchError((e) {
    print(e);
  });
}
```

Рисунок 3.15 - Реалізація функції видалення даних

## 3.6 Розробка інтерфейсу програмного модуля оцінювання якості контрольних завдань

Під час розроблення програмного модуля у вигляді мобільного додатку було дотримано всіх вище наведених правил та спроектовано оптимальний на вигляд і для застосування інтуїтивно зрозумілий інтерфейс. На рисунку 3.16 наведено головне меню вибору тестів. На рисунку 3.17 зображено сторінку проходження тесту, де користувач може обирати певні відповіді.

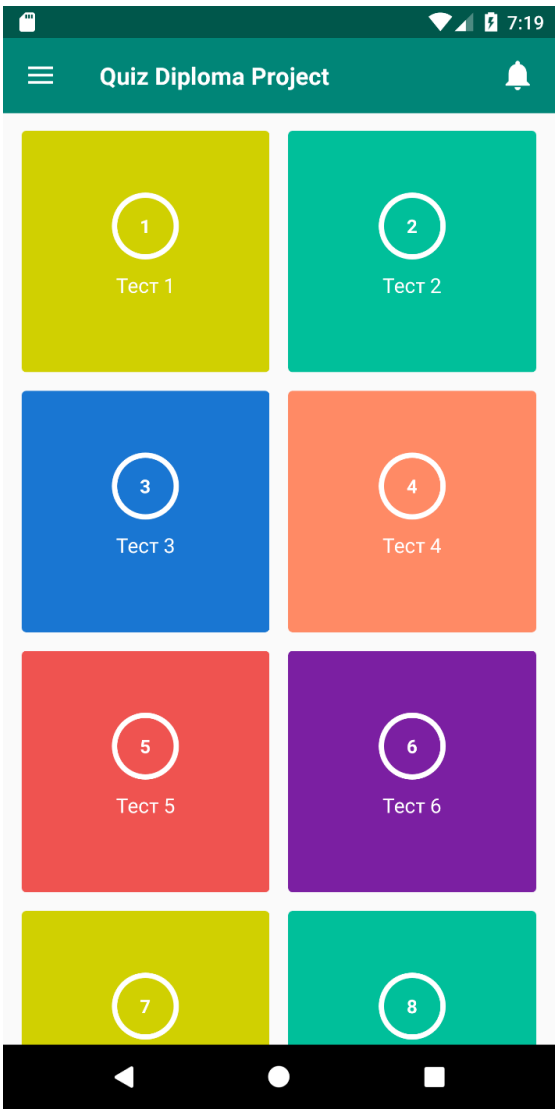


Рисунок 3.16 – Головне меню вибору тестів

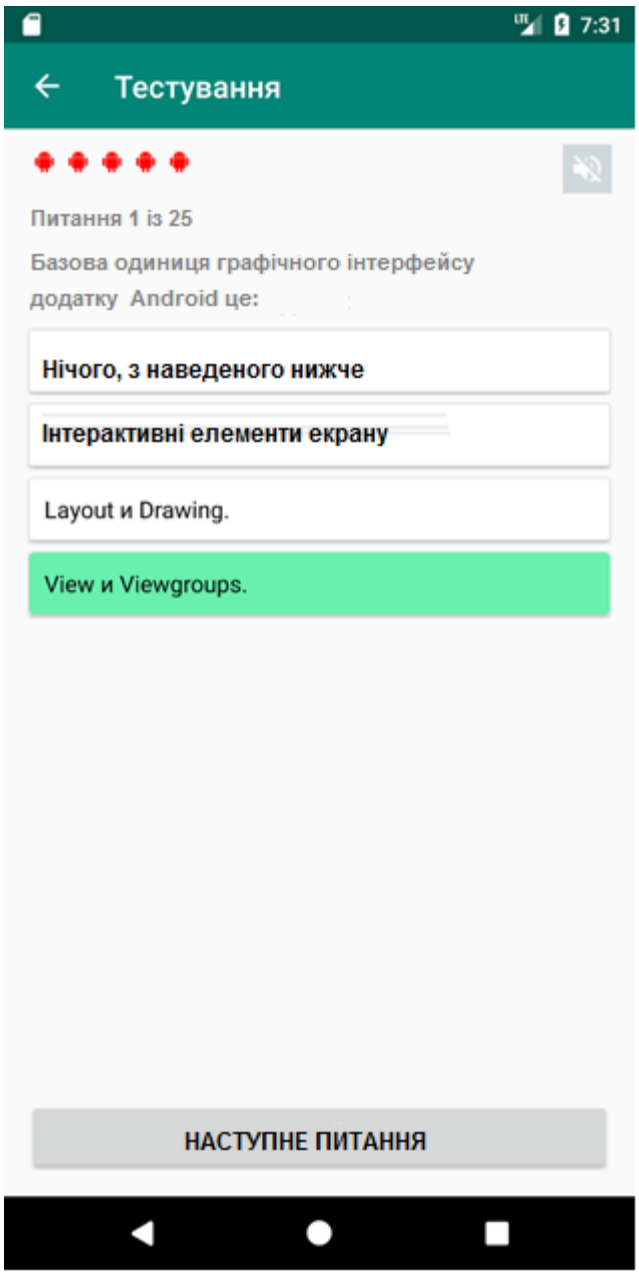


Рисунок 3.17 – Сторінка проходження тесту

### 3.7 Тестування програмного модуля оцінювання якості контрольних завдань

Невід’ємною частиною розробки будь-якого програмного забезпечення є етап тестування, який дозволяє перевірити відповідність створеного продукту до поставлених вимог і очікувань користувачів. Цей процес є ключовим для забезпечення надійності, стабільності та правильності роботи програми. Саме під час тестування виявляються потенційні недоліки, помилки або так звані дефекти, які

можуть призвести до неправильного функціонування окремих модулів або до повного блокування роботи системи.

На практиці тестування дає змогу виявити помилки ще до того, як продукт потрапить до кінцевого користувача, що дозволяє уникнути серйозних наслідків у вигляді втрати даних, збоїв у роботі або негативного досвіду користувача. У разі виявлення дефекту розробник або команда тестувальників фіксує проблему, аналізує її причину, після чого вносяться необхідні зміни до коду з подальшим повторним тестуванням, щоб переконатися в усуненні помилки.

Процес тестування, залежно від методів реалізації, поділяється на два основних типи: ручне тестування та автоматизоване тестування.

Ручне тестування (Manual Testing) передбачає перевірку роботи програмного продукту без використання автоматичних інструментів. Усі дії виконуються вручну — тестувальник самостійно відкриває вікна, взаємодіє з інтерфейсом, вводить дані та перевіряє, чи система реагує так, як було заплановано. Такий підхід дозволяє детально перевірити логіку поведінки програми, особливо на ранніх етапах розробки або у випадках, коли тестуються візуальні елементи інтерфейсу. Однак ручне тестування є трудомістким і часозатратним, особливо якщо необхідно регулярно перевіряти одні й ті самі дії.

Автоматизоване тестування (Automated Testing), навпаки, використовує спеціальні скрипти або програми, які автоматично виконують задані сценарії тестування. Це дозволяє швидко й ефективно перевіряти великі обсяги коду, виконувати повторювані тести і виявляти помилки, які можуть залишитися непоміченими при ручному тестуванні. Особливо ефективним автоматизоване тестування є у великих проєктах, де постійно вносяться зміни, і необхідна швидка перевірка вже реалізованого функціоналу.

Кожен із цих основних підходів включає в себе низку підвидів тестування, зокрема:

- функціональне тестування — перевірка того, чи функції працюють відповідно до вимог;

- регресійне тестування — тестування для переконання, що після змін у коді

попередній функціонал не порушився;

- юніт-тестування — тестування окремих модулів або функцій;

- інтеграційне тестування — перевірка взаємодії між різними модулями програми;

- системне тестування — комплексна перевірка всієї системи в цілому;

- тестування з навантаженням — оцінка поведінки системи при великій кількості запитів або користувачів. Тестування є невід'ємною частиною життєвого циклу програмного продукту, а правильне поєднання ручного та автоматизованого підходів дозволяє забезпечити якість, зручність та надійність кінцевого продукту.

Через тісний зв'язок логіки мобільного додатка з його інтерфейсом, unit-тестування є неефективним способом тестування системи, тому що не може забезпечити достатній відсоток покриття вихідного коду. Для автоматизації тестування Android-додатки використовувався сервіс Firebase Test Lab, який дозволяє, зокрема, розробляти UI-тести. Для реалізації UI-тестів в фреймворку використовується машинний обхід всіх сторінок (за умови достатності даних для переходу на іншу сторінку). Нижче наведено результати тестування швидкості ініціалізації сторінки і завантаження інших сторінок (рис. 3.18). В якості віртуального апарату для тестування використовується Google Pixel 2.

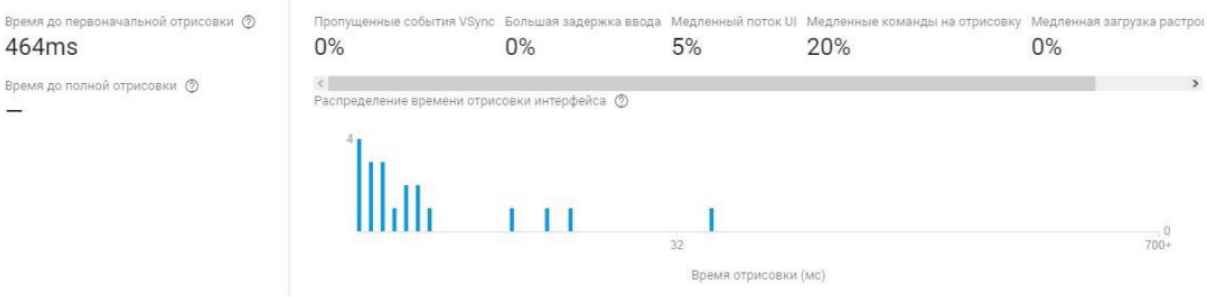


Рисунок 3.18 – Статистика тестування відтворення екрану

Швидкість запуску додатка від натискання кнопки запуску, до відтворення екрану: 464мс. При використанні в реальності це значення значно більше (від 1 до 3 секунд в залежності від числа додатків, обсягу кешу, загальної продуктивності пристрою).

Також ведеться метрика продуктивності додатка (рис. 3.19). Можна побачити, що ініціалізація при запуску вимагає 30% від продуктивності восьмиядерного процесора Snapdragon 835 тактової частоти 2450 МГц, далі використовується не більше 10-15% і трохи більше ніж 100 МБ оперативної пам'яті.


Рисунок 3.19 – Метрика продуктивності додатка

Результатом машинного обходу сторінок є задовільний результат, пов'язаний зі швидкістю відображення сторінок додатку. При першому запуску, додаток завантажується до 3 секунд на різних пристроях. Подальші сторінки відкриваються набагато швидше: від отриманих в тестуванні 464 мілісекунд до половини секунди (при первинному завантажені даних користувача).

Тестування продуктивності дало позитивні результати: споживання потужності процесора верхнього цінового сегмента при запуску вимагає не більше 30%, що говорить про відсутність зависань і виключає проблем з багатозадачністю в ОС Android (користувач може використовувати інші додатки, в тому числі фонові) навіть при використанні менш продуктивних пристроїв продуктивності цілком достатньо для стабільної роботи.

Функціональне тестування - це тестування програмного забезпечення з метою перевірки можливості бути реалізованим функціональних вимог [4]. Функціональні вимоги визначають, що саме робить програмне забезпечення, які завдання воно

вирішує. Використовуючи методологію функціонального тестування, перевіримо роботу розробленого програмного модуля.

Тест № 1. Коректне відображення екрану головної сторінки додатка.

Вхідні дані: -

Мета: Перевірити коректність відображення екрану, панелі навігації та вибору тесту.

Результат: тест пройдено.

Тест № 2. Коректність відображення зробленої помилки.

Вхідні дані: Додаток відкрито на екрані проходження тесту.

Мета: Перевірити коректність проходження тесту, роботи відображення неправильної та коректної відповіді в тесті (рис. 3.20).

Результат: тест пройдено.

Тест № 3. Коректність відображення вікна результату тесту.

Вхідні дані: Додаток відкрито на екрані головного меню.

Мета: Перевірити коректність відображення вікна результату тесту та можливості відправлення цього результату (рис. 3.21).

Хід проведення:

1. Обрати тест;

2. Розпочати проходження тесту та завершити його;

3. Отримати результат тестування;

4. Поділитись результатами тестування з іншими користувачами ;

Результат: тест пройдено.

Інтеграційне тестування - повна перевірка програмного продукту після його складання з метою виявлення помилок, що виникають в процесі інтеграції програмних модулів або компонентів [4].

Зібраний проект був перевірений на віртуальних пристроях з різними характеристиками екрану (перевірялася некоректність відображення елементів інтерфейсу): 1280x768px, 1280x800px, 1980x1080px, 2160x1080px; версіями: Android 5.1.1, Android 6, Android 7, Android 8.

Помилок не було виявлено, всі елементи на всіх дозволах відобразилися коректно. В ході перевірки на різних версіях операційної системи також не було виявлено помилок, мобільний додаток працює коректно на всіх підтримуваних версіях ОС.

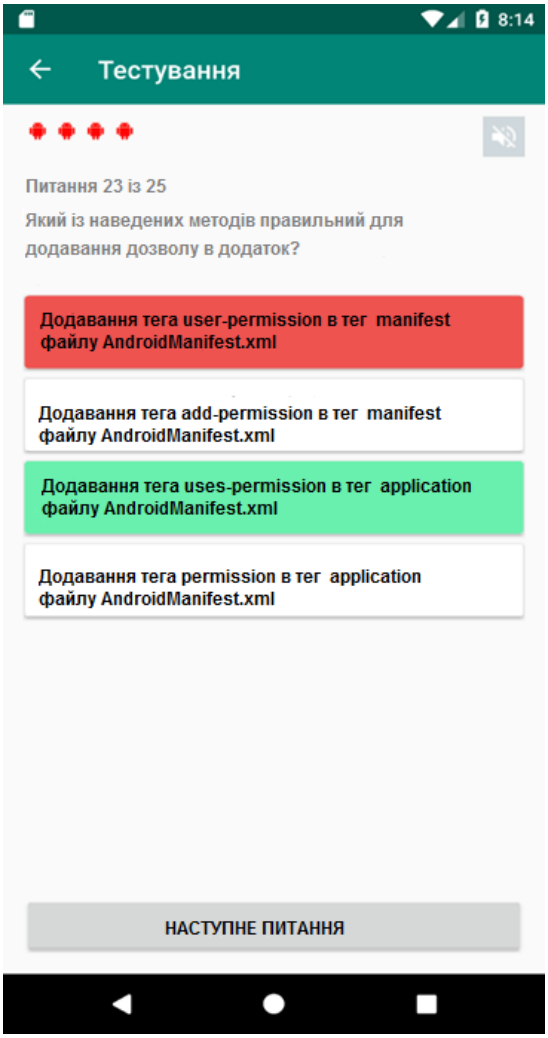


Рисунок 3.20 – Результат проходження тесту роботи відображення обраної та коректної відповіді в тесті

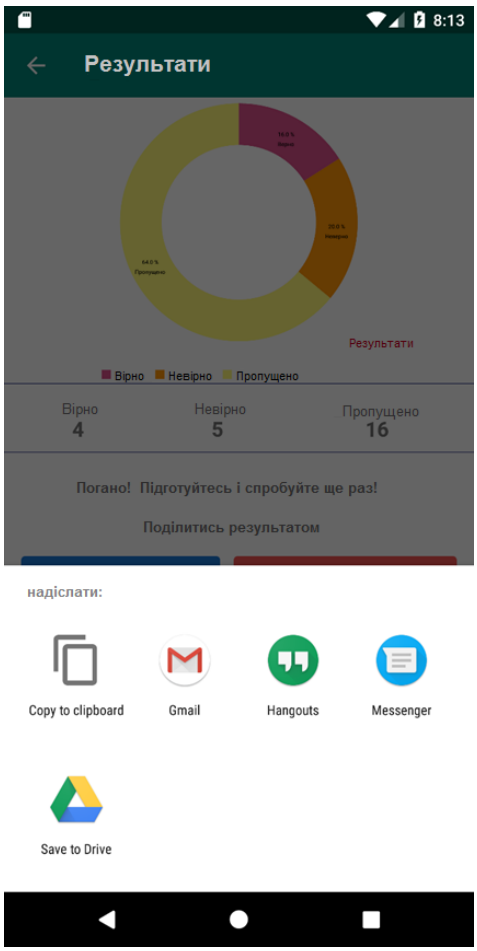


Рисунок 3.21 – Результат проходження тесту коректної роботи вікна результату тесту та можливості відправлення цього результату

Проведемо порівняння функціональних можливостей програм для оцінювання якості контрольних завдань (табл. 3.2)

Таблиця 3.2 – порівняння функціональних можливостей програм для оцінювання якості контрольних завдань

| | Оновлення бази тестів | Навчальний режим проходження тестів | Глибокий статистичний аналіз результатів | Згенерувати власний тест | Надіслати тест друзям |
|---|---|---|---|---|---|
| Створена програма | + | + | + | + | + |
| QuizUp | + | - | + | - | - |
| Quiz Your Friends | + | - | - | - | + |
| DK Quiz | + | - | - | - | - |

Згідно з табл. 3.2, новий програмний продукт володіє розширеним функціоналом, який підвищує його ефективність та зручність використання у навчальному процесі. Однією з ключових особливостей є навчальний режим проходження тестів, який дозволяє користувачу проходити тести в інтерактивному форматі з можливістю отримання підказок, пояснень до відповідей, а також негайного зворотного зв’язку після кожного питання. Це суттєво покращує засвоєння матеріалу, дозволяючи користувачеві не просто відповідати на запитання, а й розуміти логіку правильної відповіді.

Ще однією важливою функцією є можливість створення власних тестів. Користувачі можуть самостійно формувати тести, обираючи тематику, кількість запитань, типи відповідей (тест із вибором однієї правильної відповіді, множинний вибір, відкриті запитання тощо), а також встановлювати час на проходження.

Завдяки таким функціональним можливостям програмний продукт стає гнучким, індивідуалізованим інструментом, який може бути адаптований під різноманітні освітні потреби користувача.

### 3.8 Висновок до розділу 3

Описано та проаналізовано існуючі сучасні мови програмування та середовищ розробки в яких можна розробляти мобільні додатки. Також було ознайомлено з основними принципами та специфікаціями на яких будуються вищевказані технології. Кожна з представлених технологій використовує найсучасніші методи розробки програмного забезпечення.

Вирішено було обрати Firebase, як основу для використання в реалізації програми. Даний сервіс є кращим, оскільки зумовлений вибором мови і можливостями безкоштовного тарифу. Для зручності написання коду, його налагодження та виконання було обрано середовище розробки Android Studio.

Проведене тестування програми на наявність помилок або дефектів, показало, що додаток повністю придатний для користування.

# ВИСНОВКИ

У ході виконання бакалаврської кваліфікаційної роботи розроблено програмний модуль оцінювання якості контрольних завдань.

В роботі визначені функціональні і нефункціональні вимоги до додатка оцінювання якості контрольних завдань. Виконано порівняння існуючих додатків оцінювання якості контрольних завдань. В ході аналізу об’єкту проектування було визначено вимоги до програмного модуля оцінювання якості контрольних завдань.

В ході дослідження було розроблено алгоритм роботи програмного засобу оцінювання якості завдань. Обгрунтовано використання клієнт-серверної технології для реалізації програмного модуля оцінювання якості контрольних завдань

Розроблено UML-діаграму класів програмного модуля оцінювання якості контрольних завдань. Описано та проаналізовано існуючі сучасні мови програмування та середовищ розробки в яких можна розробляти мобільні додатки.

Вирішено було обрати Firebase, як основу для використання в реалізації програми. Даний сервіс є кращим, оскільки зумовлений вибором мови і можливостями безкоштовного тарифу. Для зручності написання коду, його налагодження та виконання було обрано середовище розробки Android Studio.

Проведене тестування програми на наявність помилок або дефектів, показало, що додаток повністю придатний для користування.

Новий програмний продукт володіє розширеним функціоналом, який підвищує його ефективність та зручність використання у навчальному процесі. Однією з ключових особливостей є навчальний режим проходження тестів, який дозволяє користувачу проходити тести в інтерактивному форматі з можливістю отримання підказок, пояснень до відповідей, а також негайного зворотного зв’язку після кожного питання. Ще однією важливою функцією є можливість створення власних тестів.

Отже всі поставлені задачі виконано, мету роботи досягнуто.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Бондаренко В. Мобільні застосунки як засіб комунікації в суспільстві знань: освітній аспект / В. Бондаренко // Наук. пр. Нац. б-ки України ім. В. І. Вернадського : зб. наук. пр. / НАН України, Нац. б-ка України ім. В. І. Вернадського, Асоц. б-к України. – Київ, 2017. – Вип. 48. – С. 576–590.

2. Наукова освіта як основа формування інноваційної компетентності в умовах цифрової трансформації суспільства [Електроний ресурс] – Режим доступу: https://www.researchgate.net/publication/342576932_NAUKOVA_OSVITA_AK_OSNOVA_FORMUVANNA_INNOVACIJNOI_KOMPETENTNOSTI_V_UMOVAH_CIFROVOI_TRANSFORMACII_SUSPILSTVA

3. Мобільний додаток QuizUp. [Електронний ресурс]. – Режим доступу: https://itunes.apple.com/us/app/quizup/id718421443

4. Мобільний додаток Quiz Your Friends. [Електронний ресурс]. – Режим доступу: https://itunes.apple.com/us/app/quiz-your-friends-see-who/id919383759

5. Мобільний додаток DK Quiz. [Електронний ресурс]. – Режим доступу: https://itunes.apple.com/us/app/dk-quiz/id556884950

6. Understanding the 3 Types of Mobile Apps: Native, Mobile, and Hybrid [Електронний ресурс]. – Режим доступу: https://www.charterglobal.com/understanding-the-3-types-of-mobile-apps-development-services/

7. Native applications and platforms [Електронний ресурс]. – Режим доступу: https://searchsoftwarequality.techtarget.com/definition/native-application-native-app

8. App Store [Електронний ресурс]. – Режим доступу: https://ru.wikipedia.org/wiki/App_Store

9. From Android Market to Google Play: a brief history of the Play Store [Електронний ресурс]. – Режим доступу: androidauthority.com/android-market-google-play-history-754989/

10. WebView [Електронний ресурс]. – Режим доступу: https://developer.android.com/reference/android/webkit/WebView

11. Beginning App Development with Flutter / Rap Payne, 2019. – 334 с

12. Dart programming language [Електронний ресурс] – Режим доступу: https://dart.dev/

13. Slavomir Stankov, Branko Žitko and Ani Grubišić. Ontology as a Foundation for Knowledge Evaluation in Intelligent E-learning Systems. AIED’05 Workshop SW-EL’05: Applications of Semantic Web Technologies for E-Learning. Papers of 12th International Conference on Artificial Intelligence in Education (AIED 2005). Amsterdam, 2005. [Електронний ресурс] – Режим доступу: http://hcs.science.uva.nl/AIED2005/W3proc.pdf

14. Berners-Lee T. "Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential", The MI Press, 2005

15. Мови програмування. [Електронний ресурс] – Режим доступу: http://zei.narod.ru/Comparison_C__Java_Cpp_3.pdf

16. .Eclipse. [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/Eclipse

17. Android Studio. [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/Android_Studio

18. Класифікація мобільних додатків [Електронний ресурс]. – Режим доступу: http://voroninstudio.eu/uk/service/razrabotka-mobilnih-prilozheniy

19. Flutter [Електронний ресурс] – Режим доступу: https://flutter.dev/

20. Методичні вказівки до виконання бакалаврських кваліфікаційних робіт для студентів спеціальності 122 «Комп’ютерні науки» [Електронний ресурс] / уклад.: А. А. Яровий, О. В. Сілагін, І. В. Богач. – Вінниця : ВНТУ, 2023. – (PDF, 54 с.).

## ДОДАТОК А (обов'язковий)
## ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Назва роботи: Програмний модуль оцінювання якості контрольних завдань

Тип роботи: бакалаврська кваліфікаційна робота
(бакалаврська кваліфікаційна робота / магістерська кваліфікаційна робота)

Підрозділ кафедра комп'ютерних наук
(кафедра, факультет, навчальна група)

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism 21,25 %

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне)

✓ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту

☐ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.

☐ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

Яровий А.А., зав. каф. КН _____
(прізвище, ініціали, посада) (підпис)

Колесницький О.К., проф. каф КН _____
(прізвище, ініціали, посада) (підпис)

Особа, відповідальна за перевірку _____ Озеранський В.С.
(підпис) (прізвище, ініціали)

З висновком експертної комісії ознайомлений(-на)

Керівник _____ Петришин С.І.
(підпис) (прізвище, ініціали, посада)

Здобувач _____ Гнатюк М.І.
(підпис) (прізвище, ініціали)

# ДОДАТОК Б (довідниковий)
# Інструкція користувача

Після запуску додатку відкривається вікно авторизації (рис. Б.1).

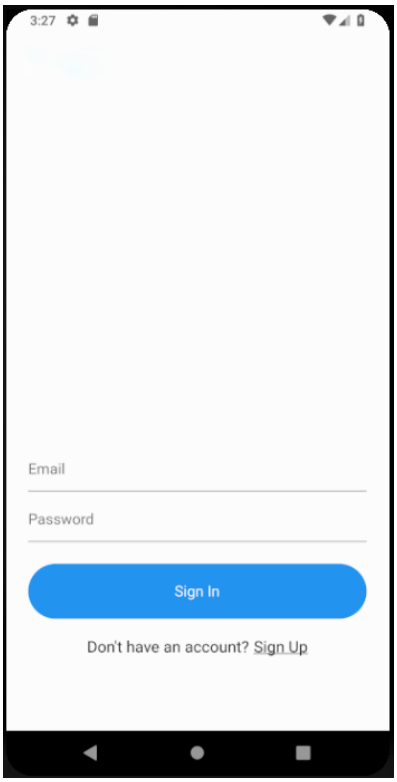


Рисунок Б.1 – Вікно авторизації

Для роботи з додатком необхідно мати обліковий запис та бути авторизованим, якщо у вас немає облікового запису, вам необхідно буде зареєструватись.

Після того як ви зареєструвалися або авторизувалися, ви потрапите на головний екран додатка, потім вам потрібно буд обрати тест (рис. Б.2).

Після вибору тесту, у користувача буде запитано чи він впевнений, що хоче почати проходження тесту, якщо це так, то йому потрібно буде натиснути клавішу «Так», якщо ні, то йому потрібно буде натиснути «Ні».

При натисканні клавіши «Ні», користувач повернеться в попереднє меню, а саме в головне меню вибору тестів.

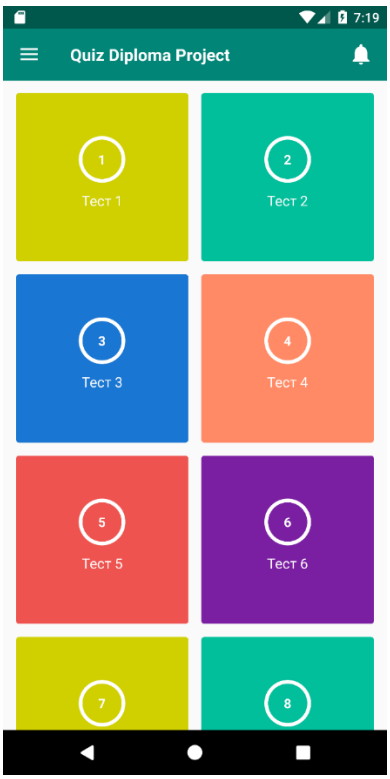


Рисунок Б.2 – Головне меню вибору тестів

При натисканні клавіши «Так» розпочнеться тестування користувача, йому будуть задані питання та наданні варіанти відповідей. Кожний тест містить 25 запитань та 4 варіанти відповідей.

Для проходження тесту користувач повинний обрати одну відповідь та натиснути клавішу «Наступне питання» (рис. Б.3), кількість неправильних відповідей обмежена 5 помилками.

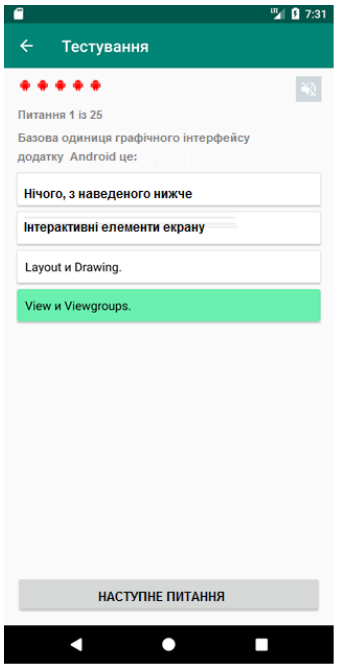


Рисунок Б.3 – Проходження тесту користувачем.

При закінчені ліміту неправильних відповідей, користувач отримає повідомлення про те що ліміт неправильних відповідей вичерпано та тестування завершено, після чого йому буть показані результати тестування, Також функціоналом додатку передбачено, що користувач має можливість пропускати питання, при умові, якщо в нього є невикористанні помилки, пропущені питання зменшують кількість можливих неправильних відповідей яких користувач може припуститись. Такі відповіді в результатах тестування будуть відображатись як пропущені відповіді, а також відображено які саме питання були пропущені (рис. Б.4).

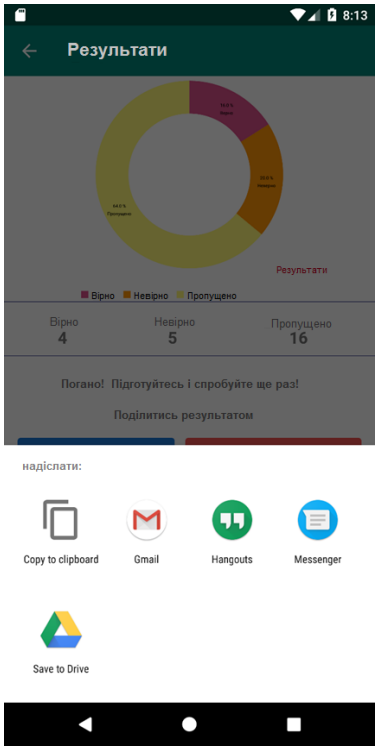


Рисунок Б.4 – Результат проходження тесту користувачем

Для завершення роботи з мобільним додатком потрібно повернутись в головне меню, відкрити бокове меню та натиснути клавішу «Вихід», альтернативним способом виходу з мобільного додатку є подвійне натискання клавіши «Назад». Після вибору одного з способів з’явиться вікно підтвердження виходу, в ньому потрібно буде підтвердити вихід натиснувши клавішу «Так».

## ДОДАТОК В (обов'язковий)
## Лістинг програми

### Лістинг компоненту SignIn

```
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:quizapp2/services/auth.dart';
import 'package:quizapp2/widget/widget.dart';

class SignIn extends StatefulWidget {
  final Function toogleView;

  SignIn({this.toogleView});

  @override
  _SignInState createState() => _SignInState();
}

class _SignInState extends State<SignIn> {
  final AuthService _authService = AuthService();
  TextEditingController emailEditingController = new TextEditingController();
  TextEditingController passwordEditingController = new TextEditingController();

  @override
  Widget build(BuildContext context) {
    SystemChrome.setSystemUIOverlayStyle(
        SystemUiOverlayStyle(statusBarColor: Colors.white));
    return Scaffold(
      backgroundColor: Colors.white,
      appBar: AppBar(
        title: AppLogo(),
        brightness: Brightness.light,
        elevation: 0.0,
        backgroundColor: Colors.transparent,
        //brightness: Brightness.li,
      ),
      body: Container(
        padding: EdgeInsets.symmetric(horizontal: 24),
        child: Column(
          children: [
            Spacer(),
            Container(
              child: Column(
```

```
children: [
  TextField(
    decoration: InputDecoration(hintText: "Email"),
  ),
  SizedBox(
    height: 6,
  ),
  TextField(
    decoration: InputDecoration(hintText: "Password"),
  ),
  SizedBox(
    height: 24,
  ),
  Container(
    alignment: Alignment.center,
    width: MediaQuery.of(context).size.width,
    padding: EdgeInsets.symmetric(horizontal: 24, vertical: 20),
    decoration: BoxDecoration(
        color: Colors.blue,
        borderRadius: BorderRadius.circular(30)),
    child: Text(
      "Sign In",
      style: TextStyle(fontSize: 16, color: Colors.white),
    ),
  ),
  SizedBox(
    height: 20,
  ),
  Row(
    mainAxisAlignment: MainAxisAlignment.center,
    children: [
      Text('Don\'t have an account? ',
          style:
              TextStyle(color: Colors.black87, fontSize: 17)),
      GestureDetector(
        onTap: () {
          widget.toogleView();
        },
        child: Container(
          child: Text('Sign Up',
              style: TextStyle(
                  color: Colors.black87,
                  decoration: TextDecoration.underline,
                  fontSize: 17)),
        ),
```

```
                  ),
                ],
              ),
            ],
          ),
        ),
        SizedBox(
          height: 80,
        )
      ],
    ),
  ),
 );
 }
}
```

**Лістинг компоненту SignUp**

```
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:quizapp2/helper/constants.dart';
import 'package:quizapp2/services/auth.dart';
import 'package:quizapp2/services/database.dart';
import 'package:quizapp2/views/home.dart';
import 'package:quizapp2/widget/widget.dart';

class SignUp extends StatefulWidget {
 final Function toogleView;

 SignUp({this.toogleView});

 @override
 _SignUpState createState() => _SignUpState();
}

class _SignUpState extends State<SignUp> {
 AuthService authService = new AuthService();
 DatabaseService databaseService = new DatabaseService();
 final _formKey = GlobalKey<FormState>();

 // text feild
 bool _loading = false;
 String email = '', password = '', name = '';

 getInfoAndSignUp() async {
```

```
  if (_formKey.currentState.validate()) {
    setState(() {
      _loading = true;
    });

    await authService
        .signUpWithEmailAndPassword(email, password)
        .then((value) {
      Map<String, String> userInfo = {
        "userName": name,
        "email": email,
      };

      databaseService.addData(userInfo);
      Constants.saveUserLoggedInSharedPreference(true);

      Navigator.pushReplacement(
          context, MaterialPageRoute(builder: (context) => Home()));
    });
  }
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.white,
    appBar: AppBar(
      title: AppLogo(),
      brightness: Brightness.light,
      elevation: 0.0,
      backgroundColor: Colors.transparent,
      //brightness: Brightness.li,
    ),
    body: Container(
      padding: EdgeInsets.symmetric(horizontal: 24),
      child: _loading
          ? Container(
              child: Center(child: CircularProgressIndicator()),
            )
          : Column(
              children: [
                Spacer(),
                Form(
                  key: _formKey,
                  child: Container(
```

```
child: Column(
  children: [
    TextFormField(
      validator: (val) =>
          val.isEmpty ? "Enter an Name" : null,
      decoration: InputDecoration(hintText: "Name"),
      onChanged: (val) {
        name = val;
      },
    ),
    SizedBox(
      height: 6,
    ),
    TextFormField(
      validator: (val) => validateEmail(email)
          ? null
          : "Enter correct email",
      decoration: InputDecoration(hintText: "Email"),
      onChanged: (val) {
        email = val;
      },
    ),
    SizedBox(
      height: 6,
    ),
    TextFormField(
      obscureText: true,
      validator: (val) => val.length < 6
          ? "Password must be 6+ characters"
          : null,
      decoration: InputDecoration(hintText: "Password"),
      onChanged: (val) {
        password = val;
      },
    ),
    SizedBox(
      height: 24,
    ),
    GestureDetector(
      onTap: () {
        getInfoAndSignUp();
      },
      child: Container(
        alignment: Alignment.center,
        width: MediaQuery.of(context).size.width,
```

```
                padding: EdgeInsets.symmetric(
                    horizontal: 24, vertical: 20),
                decoration: BoxDecoration(
                    color: Colors.blue,
                    borderRadius: BorderRadius.circular(30)),
                child: Text(
                  "Sign Up",
                  style: TextStyle(
                      fontSize: 16, color: Colors.white),
                ),
              ),
            ),
            SizedBox(
              height: 20,
            ),
            Row(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                Text('Already have and account? ',
                    style: TextStyle(
                        color: Colors.black87, fontSize: 17)),
                GestureDetector(
                  onTap: () {
                    widget.toogleView();
                  },
                  child: Container(
                    child: Text('Sign In',
                        style: TextStyle(
                            color: Colors.black87,
                            decoration: TextDecoration.underline,
                            fontSize: 17)),
                  ),
                ),
              ],
            ),
          ],
        ),
      ),
      SizedBox(
        height: 80,
      )
    ],
  ),
),
```

```
    );
  }
}

bool validateEmail(String value) {
  Pattern pattern =
      r'^(([^<>()[\]\\.,;:\s@\"]+(\.[^<>()[\]\\.,;:\s@\"]+)*)|(\".+\"))@((\[[0-9]{1,3}\.[0-
9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$';
  RegExp regex = new RegExp(pattern);
  return (!regex.hasMatch(value)) ? false : true;
}
```

**Лістинг компоненту Home**

```
import 'package:flutter/material.dart';
import 'package:quizapp2/services/database.dart';
import 'package:quizapp2/views/create_quiz.dart';
import 'package:quizapp2/views/quiz_play.dart';
import 'package:quizapp2/widget/widget.dart';

class Home extends StatefulWidget {
  @override
  _HomeState createState() => _HomeState();
}

class _HomeState extends State<Home> {
  Stream quizStream;
  DatabaseService databaseService = new DatabaseService();

  Widget quizList() {
    return Container(
      child: Column(
        children: [
          StreamBuilder(
            stream: quizStream,
            builder: (context, snapshot) {
              return snapshot.data == null
                  ? Container()
                  : ListView.builder(
                      shrinkWrap: true,
                      physics: ClampingScrollPhysics(),
                      itemCount: snapshot.data.documents.length,
                      itemBuilder: (context, index) {
                        return QuizTile(
                          noOfQuestions: snapshot.data.documents.length,
```

```
                    imageUrl:
                        snapshot.data.documents[index].data['quizImgUrl'],
                    title:
                        snapshot.data.documents[index].data['quizTitle'],
                    description:
                        snapshot.data.documents[index].data['quizDesc'],
                    id: snapshot.data.documents[index].data["id"],
                  );
                });
          },
        )
      ],
    ),
  );
}

@override
void initState() {
  databaseService.getQuizData().then((value) {
    quizStream = value;
    setState(() {});
  });
  super.initState();
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.white,
    appBar: AppBar(
      title: AppLogo(),
      brightness: Brightness.light,
      elevation: 0.0,
      backgroundColor: Colors.transparent,
      //brightness: Brightness.li,
    ),
    body: quizList(),
    floatingActionButton: FloatingActionButton(
      child: Icon(Icons.add),
      onPressed: () {
        Navigator.push(
            context, MaterialPageRoute(builder: (context) => CreateQuiz()));
      },
    ),
  );
```

```
  }
}

class QuizTile extends StatelessWidget {
  final String imageUrl, title, id, description;
  final int noOfQuestions;

  QuizTile(
      {@required this.title,
      @required this.imageUrl,
      @required this.description,
      @required this.id,
      @required this.noOfQuestions});

  @override
  Widget build(BuildContext context) {
    return GestureDetector(
      onTap: (){
        Navigator.push(context, MaterialPageRoute(
          builder: (context) => QuizPlay(id)
        ));
      },
      child: Container(
        padding: EdgeInsets.symmetric(horizontal: 24),
        height: 150,
        child: ClipRRect(
          borderRadius: BorderRadius.circular(8),
          child: Stack(
            children: [
              Image.network(
                imageUrl,
                fit: BoxFit.cover,
                width: MediaQuery.of(context).size.width,
              ),
              Container(
                color: Colors.black26,
                child: Center(
                  child: Column(
                    mainAxisAlignment: MainAxisAlignment.center,
                    children: [
                      Text(
                        title,
                        style: TextStyle(
                            fontSize: 18,
                            color: Colors.white,
```

```
                    fontWeight: FontWeight.w500),
                ),
                SizedBox(height: 4,),
                Text(
                  description,
                  style: TextStyle(
                    fontSize: 13,
                    color: Colors.white,
                    fontWeight: FontWeight.w500),
                )
              ],
            ),
          ),
        )
      ],
    ),
  ),
),
);
  }
}
```

## ДОДАТОК Г (обов'язковий)

## ГРАФІЧНА ЧАСТИНА

# «ПРОГРАМНИЙ МОДУЛЬ ОЦІНЮВАННЯ ЯКОСТІ КОНТРОЛЬНИХ ЗАВДАНЬ»

Виконав: студент 4 курсу, групи 4КН-21б
спеціальності 122 – Комп'ютерні науки
(шифр і назва напряму підготовки, спеціальності)

Гнатюк М.І.
(прізвище та ініціали)

Керівник: к.т.н., ст. викл. кафедри КН
Петришин С.І.
(прізвище та ініціали)

03.06.2025р.

Вінниця ВНТУ - 2025 рік


Рисунок Г.1 – Програми аналоги для оцінювання якості контрольних завдань

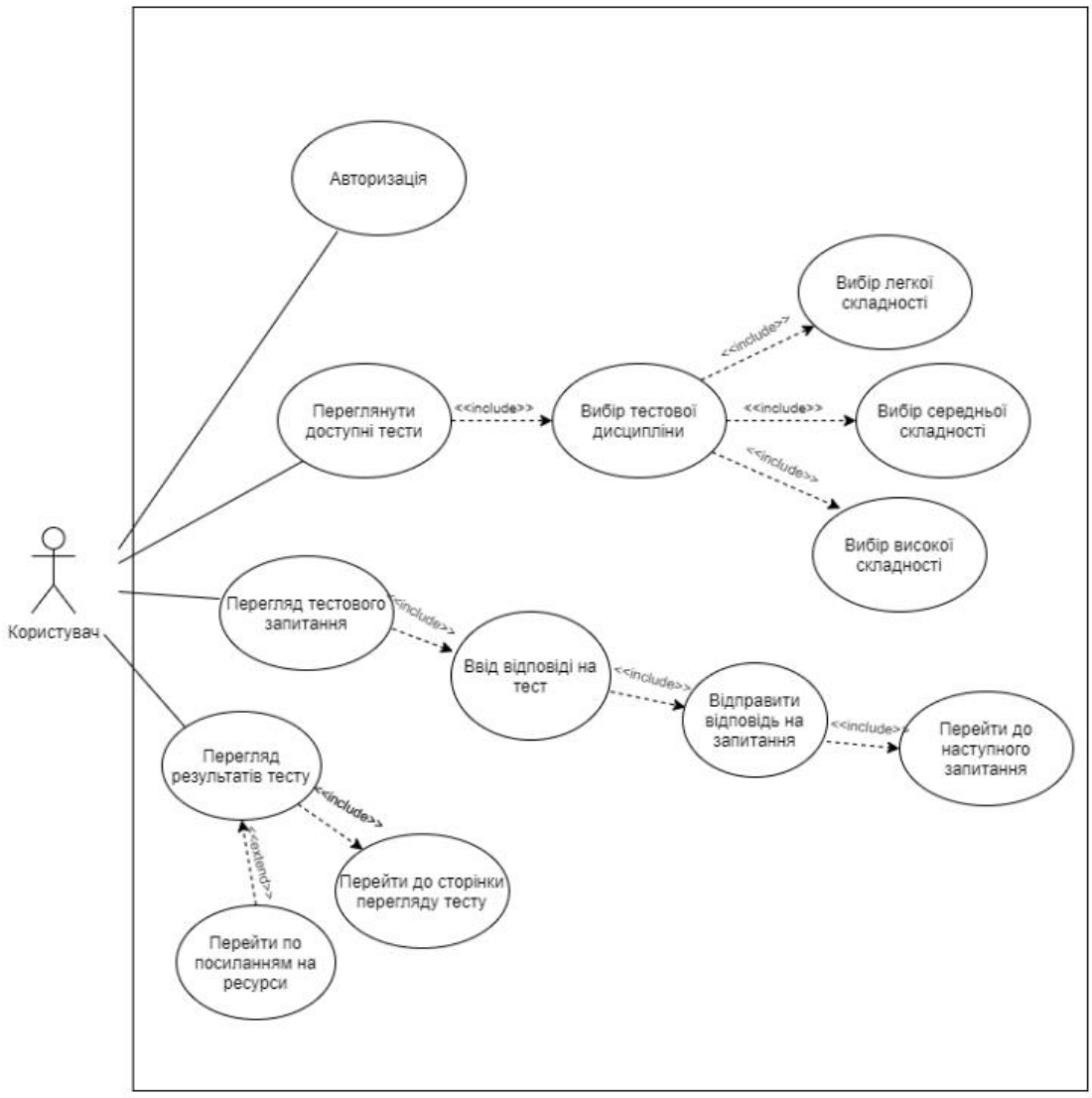


Рисунок Г.2 – Діаграма прецедентів програмного модуля оцінювання якості контрольних завдань

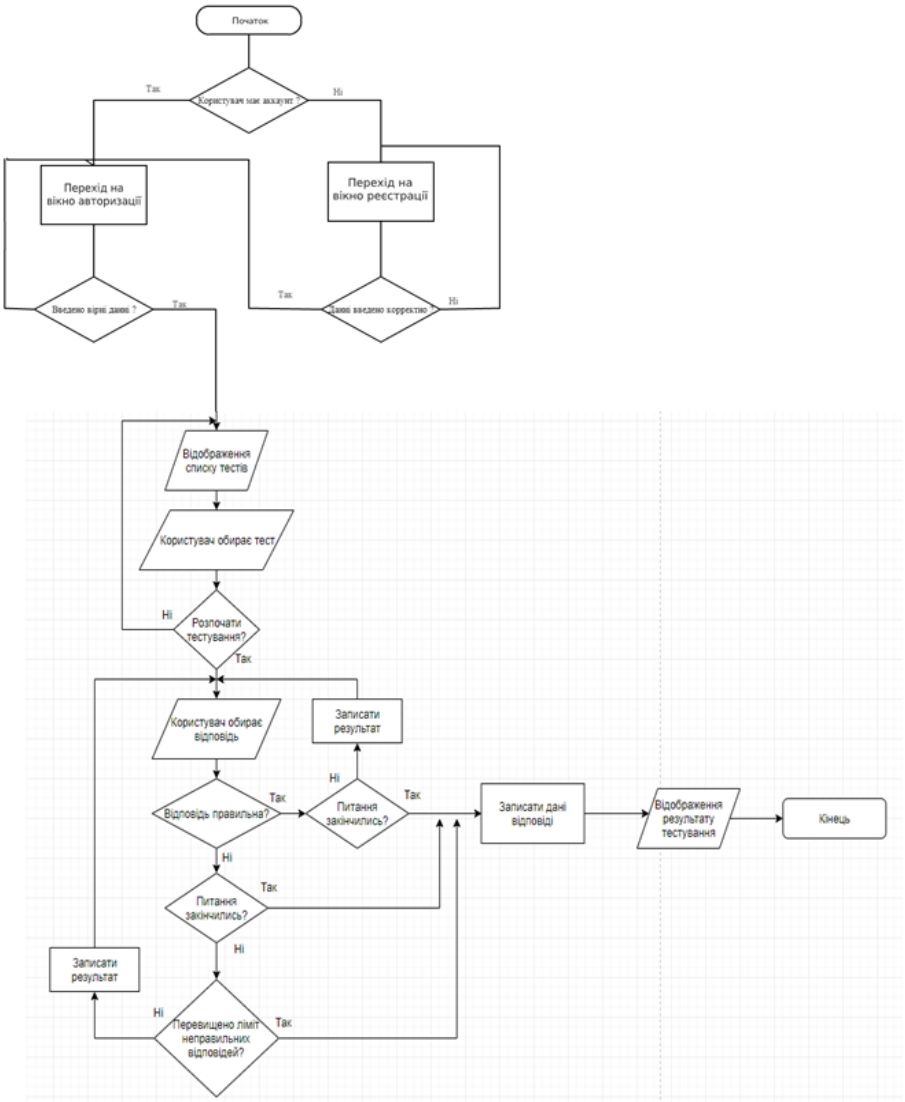


Рисунок Г.3 – Загальний алгоритм роботи програмного модуля оцінювання якості контрольних завдань

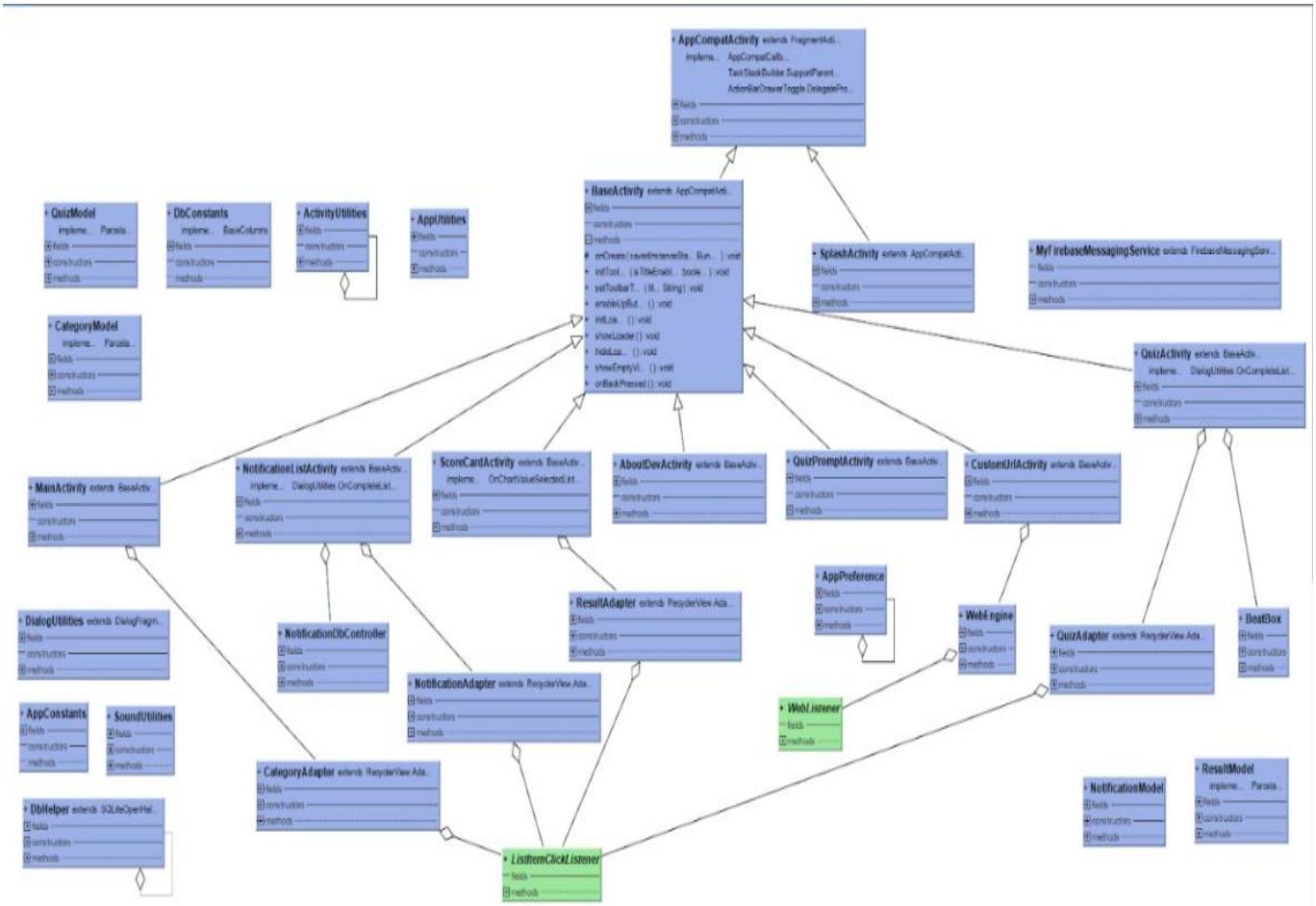


Рисунок Г.4 – Діаграма класів додатку оцінювання якості контрольних завдань

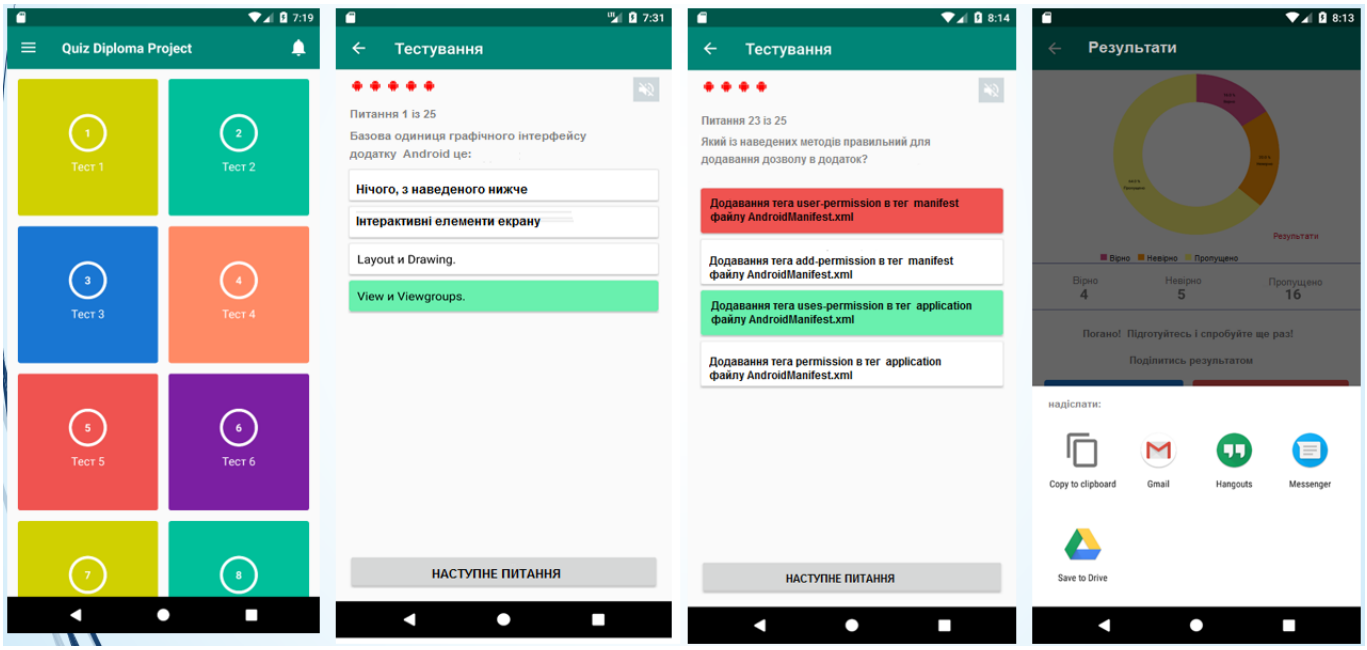


Рисунок Г.5 – Інтерфейс розробленого програмного модуля