КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп’ютерних наук та кібернетики
Кафедра теорії та технології програмування

Кваліфікаційна робота
на здобуття ступеня бакалавра
за спеціальністю 122 Комп'ютерні науки
на тему:

МОБІЛЬНИЙ ЗАСТОСУНОК ДЛЯ ТЕСТУВАННЯ З МАТЕМАТИКИ

Виконала студентка 4-го курсу
Ольга ПРОЦЕНКО _____
(підпис)

Науковий керівник:
доцент, кандидат пед. наук
Наталія РУСІНА _____
(підпис)

Засвідчую, що в цій роботі немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Роботу розглянуто й допущено до захисту на засіданні кафедри теорії та технології програмування
«____» _____ 2023 р., протокол № ____

Завідувач кафедри
Микола НІКІТІЧЕНКО _____

Київ – 2023

РЕФЕРАТ

Обсяг роботи: загальний - 46 сторінок, з них основний текст – 44 сторінки, 22 ілюстрації, 11 таблиць, 33 джерела посилань, 1 додаток.

МОБІЛЬНИЙ ЗАСТОСУНОК, ТЕСТУВАННЯ З МАТЕМАТИКИ, ТЕОРЕТИЧНІ МАТЕРІАЛИ, РІЗНОРІВНЕВІ ЗАВДАННЯ, ГРА-ВІКТОРИНА , РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОПЕРАЦІЙНОЇ СИСТЕМИ ANDROID, FLUTTER, FIREBASE.

Об’єктом дослідження є процес опрацювання контенту для тестування з математики.

Метою кваліфікаційної роботи є розробка мобільного застосунку «Math Boost» для підготовки до тестування з математики, що надасть можливість користувачам вивчати матеріал з математики та розв'язувати завдання. Застосунок повинен бути простим та зручним у використанні, мати інтуїтивний інтерфейс та надавати можливість користувачам відстежувати свій прогрес.

Інструменти розроблення: середовище розробки Android Studio, мова програмування Dart, фреймворк Flutter, хмарний сервіс Firebase, SQLite, Floor.

Результат роботи: виконано огляд ринку застосунків для вивчення математики, розроблено мобільний застосунок «Math Boost» для підготовки до тестування з математики, який дозволяє користувачам: ознайомлюватись з теоретичним матеріалом з різних тем; практикуватися в розв'язуванні завдань; переглядати свій профіль, для відстеження результатів; отримувати бали за кожну правильну відповідь, таким чином збільшувати свій рейтинг.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

DAO – Data Access Object, об'єкт доступу до даних

ORM – Object-Relational Mapping, об'єктно-реляційна проекція

БД – База даних

ЗНО – Зовнішнє незалежне оцінювання

ОС – Операційна система.

ВСТУП

Оцінка сучасного стану об’єкта розробки. Останні роки принесли значні зміни в систему освіти, в результаті чого дистанційне навчання стало невід'ємною частиною багатьох освітніх програм. У контексті вивчення математики, цей перехід до дистанційного навчання викликає певні проблеми, які варто враховувати для забезпечення ефективного навчання учнів.

Одна з головних проблем - втрата особистого контакту з викладачем та іншими учнями, що може вплинути на здатність учня до взаємодії та розуміння матеріалу. Також виникають виклики з організацією та структуруванням навчального процесу, постачанням належних навчальних матеріалів та ефективним оцінюванням знань.

У зв'язку з цим, знаходження нових, цікавих інтерактивних методів навчання стає надзвичайно важливим завданням. Інтерактивні методи дозволяють залучити учнів до активної участі в навчальному процесі, створюючи стимули для їх зацікавленості. Вони сприяють покращенню розуміння математичних концепцій та розвитку критичного мислення.

Актуальність роботи та підстави для її виконання. З поширенням мобільних пристроїв та доступу до них, підлітки стають більш залежними від цих технологій у своєму повсякденному житті.

Враховуючи цей факт, інтеграція мобільних застосунків у навчання, набуває великого значення. Створення застосунку для підготовки до тестування з математики, який поєднує навчальні матеріали з ігровими елементами та забезпечує доступ до них на мобільних пристроях, може стати ефективним способом привернути увагу та зацікавити молодше покоління.

Цей підхід дозволяє створити навчальне середовище, яке є зрозумілим, цікавим та доступним для учнів. Мобільні застосунки забезпечують гнучкість та зручність навчання, дозволяючи учням вивчати математичні концепції в будь-який час та в будь-якому місці. Вони також можуть надати інтерактивний ігровий досвід, що робить навчання більш захоплюючим та мотивуючим.

Мета й завдання роботи. Метою роботи є розробка мобільного застосунку «Math Boost» для підготовки до тестування з математики.

Для досягнення мети були поставлені наступні завдання:

- провести аналіз предметної області з метою оцінки перспектив розробки;
- провести аналіз подібних застосунків;
- обрати інструменти розробки;
- розробити діаграми прецедентів, бази даних та класів;
- розробити мобільний застосунок.

Об’єкт, методи й засоби розробки. Об’єктом дослідження є процес опрацювання контенту для тестування з математики. Предметом роботи є орієнтований на учнів закладів середньої освіти застосунок для тестування з курсу математики.

При розробці використовувались такі засоби як:

- операційна система Android [1];
- мова програмування Dart [2];
- фреймворк Flutter [3] для розробки користувацького інтерфейсу;
- хмарний сервіс Firebase [4] для збереження, керування та обробки даних у мережі.

Можливі сфери застосування. Мобільний застосунок "Math Boost" має широкий спектр можливих сфер застосування. В освітній сфері він може бути використаний для підготовки учнів та студентів до тестування з математики, включаючи вступні іспити та стандартизовані тести. Крім того, він може слугувати інструментом для індивідуальної підготовки, додаткових занять та самостійного навчання. Застосунок може зацікавити людей, які цікавляться математикою і бажають розвивати свої навички у цій галузі. Загалом, "Math Boost" надає зручний спосіб вивчення математики, тренування та відстеження прогресу, сприяючи покращенню математичних здібностей користувачів.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Розвиток мобільних застосунків

Мобільний застосунок – це комп’ютерна програма або програмне забезпечення, створене для використання на різних мобільних пристроях, таких як смартфони, планшети або годинники. Мобільні застосунки часто відрізняються від настільних програм, які призначені для роботи на персональних комп'ютерах, а також веб-застосунків, які запускаються у мобільних веб-браузерах, а не на самому мобільному пристрої [5].

Мобільні застосунки виникли як результат розвитку та поширення мобільних пристроїв, таких як смартфони та планшети. Спочатку, їх основні функціональності були зосереджені на елементарних завданнях, таких як електронна пошта, календар, та бази даних контактів. Проте, зростання популярності мобільних пристроїв та їх потенціалу прискорили розширення функцій мобільних застосунків в інші сфери. Наприклад, виникли мобільні ігри, що стали дуже популярними серед користувачів, а також застосунки для автоматизації виробництва, GPS-навігації та послуг, заснованих на визначенні місцезнаходження, відстеження замовлень та квитків [6].

Мобільні застосунки можуть бути розроблені для різних мобільних платформ, таких як iOS (використовуючи мови програмування, такі як Swift [7], Objective-C [8]) або Android [9] (використовуючи Java або Kotlin [10]). Крім того, існують гібридні застосунки, що комбінують веб-технології, такі як HTML, CSS та JavaScript, для розробки застосунків, які можуть працювати на різних платформах [11].

Мобільні застосунки є невід'ємною частиною сучасного цифрового світу і відіграють важливу роль у покращенні нашого повсякденного життя. Можна виділити різні категорії, що включають розважальні ігри, соціальні медіа, комунікацію, продуктивність, навігацію, фітнес і здоров'я, фінанси, магазини та багато іншого. Вони дозволяють нам грати у високоякісні ігри, спілкуватися з друзями у соціальних мережах, здійснювати покупки онлайн, отримувати

новини та інформацію, організовувати свій час за допомогою календарів та списків справ, контролювати своє здоров'я та фітнес, здійснювати фінансові операції та багато іншого.

Важливою частиною екосистеми мобільних застосунків є магазини застосунків. Такі платформи розповсюдження можуть керуватись власниками мобільної операційної системи пристрою, наприклад App Store (iOS) [12] або Google Play Store [13]; виробниками пристроїв, такими як Galaxy Store [14] і Huawei AppGallery [15]; або третіми сторонами, такими як Amazon Appstore [16] та F-Droid [17].

Зі зростанням кількості мобільних застосунків, доступних у магазинах програмних продуктів, і покращеними можливостями смартфонів, люди завантажують все більше програм на свої пристрої. Використання мобільних застосунків стає все більш поширеним серед користувачів мобільних телефонів. Дослідження у травні 2012 року показало, що протягом попереднього кварталу більше мобільних абонентів використовували застосунки, ніж користувались Інтернетом на своїх пристроях: 51,1% проти 49,8% відповідно [18]. Дослідники виявили, що використання мобільних застосунків сильно корелює з контекстом користувача та залежить від його місцезнаходження та часу доби.

За даними дослідження 2022 року [19], було зроблено понад 255 мільярдів завантажень мобільних застосунків, що становить понад 485 000 за кожну хвилину. Це свідчить про значне зростання на 11% у порівнянні з попереднім роком.

Користувачі також продовжують активно витрачати гроші в мобільних магазинах. За останній рік витратили понад 167 мільярдів доларів, що складає більше ніж 318 000 доларів за хвилину. Хоча цей показник трохи знизився на 2% порівняно з попереднім роком, він все ще вражаючий [19].

Варто зазначити, що Android [9] є найбільш продаваною ОС у світі на смартфонах з 2011 року, а також на планшетах з 2013 року. Станом на травень 2021 року вона налічує понад три мільярди активних користувачів щомісяця, що

є найбільшою базою встановлених операційних систем, [20] і станом на січень 2021 року. Остання версія Android 13, випущена 15 серпня 2022 року [21].

Вищенаведені статистичні дані підтверджують надзвичайну популярність та поширеність мобільних застосунків серед користувачів. Вони стають важливими інструментами для розваг, комунікації, продуктивності та багатьох інших сфер життя. Ці цифри також свідчать про постійне зростання і розвиток цієї предметної області.

1.2 Огляд наявних на ринку систем

Для створення максимально конкурентоспроможного застосунку для підготовки до тестування з математики, доцільно провести аналіз вже наявних програмних продуктів, які надають подібні послуги або розв'язують схожі завдання. В ході дослідження було виокремлено декілька застосунків, що мають схожі основні риси, але водночас мають власні суттєві особливості. Серед них є "ЗНО: Математика" [22], "ЗНО 2023. Математика" [23] та "Математика: формули + тести" [24]. Кожен з цих застосунків займає відповідне місце на ринку та має свою унікальну специфіку, яка зробила їх популярними серед користувачів, які готуються до тестування з математики.

“ЗНО: Математика”

Застосунок "ЗНО: Математика" є одним з провідних рішень на ринку для підготовки до Зовнішнього незалежного оцінювання з математики [22]. Цей застосунок надає широкий спектр інструментів та ресурсів, щоб допомогти учням та абітурієнтам ефективно підготуватися до випробування.

Інтерфейс користувача застосунку простий та інтуїтивно зрозумілий, з функціональним меню та зручним розташуванням пунктів завдань (рисунок 1.1).

Рисунок 1.1 – Інтерфейс користувача застосунку “ЗНО: Математика

Одна з головних переваг "ЗНО: Математика" - це багатошаровий підхід до навчання. Застосунок пропонує чіткі теоретичні пояснення, приклади та вправи, що допомагають засвоїти матеріал. Крім того, він пропонує широкий спектр тестів, які дозволяють перевірити рівень засвоєння матеріалу та практикуватися в розв'язуванні реальних завдань, подібних до тих, що зустрінуться на ЗНО.

"ЗНО 2023. Математика"

"ЗНО 2023. Математика" є одним із застосунків, призначених для підготовки до Зовнішнього незалежного оцінювання з математики. Цей застосунок спеціально розроблений з огляду на вимоги та особливості тестування у 2023 році, що робить його привабливим вибором для абітурієнтів [23].

Однією з переваг "ЗНО 2023. Математика" є актуальність його змісту. Застосунок охоплює весь необхідний матеріал, який відповідає новітнім вимогам ЗНО на 2023 рік. Він надає доступ до теоретичних пояснень, прикладів і вправ, які охоплюють широкий спектр тем та понять математики, що будуть перевірятися під час екзамену.

Крім теоретичного матеріалу, застосунок пропонує також тести, що відповідають формату тестування ЗНО 2023 року. Це дозволяє користувачам практикуватися в розв'язуванні реальних завдань, що допомагає засвоїти стратегії та навички, необхідні для успішного проходження екзамену (рисунок 1.2). Тести можна проходити в режимі симуляції ЗНО, що допомагає абітурієнтам звикнути до умов тестування та відчути тиск, який супроводжує екзамен.

Рисунок 1.2. – Інтерфейс користувача застосунку “ЗНО 2023. Математика”

"Математика: формули + тести"

"Математика: формули + тести" є одним з застосунків для підготовки до ЗНО з математики, який пропонує користувачам зручні інструменти для вивчення математичних формул та тренування в розв'язуванні завдань [24].

Основною перевагою "Математика: формули + тести" є його фокус на математичних формулах (рисунок 1.3). Застосунок надає широкий спектр формул з різних розділів математики, включаючи алгебру, геометрію,

тригонометрію та інші. Користувачі можуть знайти необхідну формулу та використовувати її як довідник під час вивчення або розв'язування завдань.

Крім формул, застосунок також пропонує тести, що дозволяють користувачам перевірити свої знання та навички у розв'язуванні завдань. Тести можуть містити питання з різних розділів математики, що допомагає учням охопити широкий спектр тем. Результати тестів надають зворотний зв'язок щодо правильних та неправильних відповідей, що дозволяє користувачам виявити свої помилки та покращити свої навички.

Рисунок 1.3. – Інтерфейс користувача застосунку “Математика: формули + тести”

Однак, варто зазначити, що "Математика: формули + тести" має свої недоліки. Навчальний матеріал обмежений формулами, не охоплюючи широкий спектр теоретичних пояснень та прикладів. Це може ускладнити повне розуміння математичних концепцій та їх застосування в різних ситуаціях. Крім того,

недостатність додаткового навчального матеріалу може змінити динаміку вивчення та розвитку користувачів.

Після проведення аналізу наявних рішень в області, були виявлені їх особливості та здійснено порівняння сильних і слабких сторін. Отримані результати були узагальнені та представлені у вигляді таблиці порівняння систем (таблиця 1). Цей аналіз сприяє визначенню пріоритетних цілей для реалізації власної системи.

Таблиця 1.1 – Порівняння систем

Функціональні можливості	*ЗНО: Математика*	*ЗНО 2023. Математика*	*Математика: формули + тести*	*Math Boost*
Безкоштовна версія	+	-	+	+
Авторизація	+	+	-	+
Теорія	+	+	+	+
Приклади	+	+	-	+
Тести	+	+	+	+

В результаті проведеного порівняльного аналізу доречно зазначити, що застосунок “Math Boost” містить функціональні можливості відмінні від інших програмних продуктів. По-перше, “Math Boost” надає безкоштовну версію, що робить його доступним для широкого кола користувачів. Крім того, застосунок пропонує зручну авторизацію, що дозволяє користувачам зберігати свій прогрес і використовувати додаткові функціональні можливості Найважливішою функціональною можливістю “Math Boost” є можливість проходження тестів з математики. Користувачі можуть випробувати свої знання, вирішуючи різнорівневі завдання, що охоплюють різні теми. Це дозволяє перевірити свій рівень підготовки до тестування і виявити слабкі місця, які потребують додаткового вивчення.

РОЗДІЛ 2. ОГЛЯД ВИКОРИСТАНИХ ТЕХНОЛОГІЙ

2.1 Операційна система Android

Мобільна операційна система Android - ґрунтується на модифікованій версії ядра Linux та іншому програмному забезпеченні з відкритим кодом. Вона призначена переважно для смартфонів та планшетів з сенсорним екраном. Розробка Android здійснюється консорціумом розробників, відомим як Open Handset Alliance, і її комерційно підтримує Google. Офіційно система була представлена у листопаді 2007 року, а перший комерційний пристрій, що працював на Android, HTC Dream, був випущений у вересні 2008 року [1].

ОС Android використовує основні компоненти з Android Open Source Project, що є безкоштовним програмним забезпеченням з відкритим кодом (FOSS), переважно ліцензованим за ліцензією Apache. Коли Android інстальовано на пристроях, можливість модифікувати програмне забезпечення FOSS зазвичай обмежена, або не надається відповідний вихідний код, або запобігається повторне встановлення. Більшість пристроїв Android постачаються з попередньо встановленим додатковим власним програмним забезпеченням, зокрема Google Mobile Services [25], який включає основні програми, такі як Google Chrome, платформа цифрового розповсюдження Google Play і пов’язана платформа розробки служб Google Play.

Більше 70 відсотків смартфонів, що працюють на операційній системі Android, використовують екосистему Google. Деякі з них мають спеціально налаштований інтерфейс користувача та програмний пакет, такий як TouchWiz і пізніше One UI від Samsung, або HTC Sense [9]. Крім того, існують конкуруючі екосистеми та форки Android, такі як Fire OS від Amazon, ColorOS від OPPO, OriginOS від vivo та MagicUI від Honor, а також користувальницькі ROM, наприклад, LineageOS. Однак назва та логотип «Android» є торговими марками Google, яка встановлює стандарти для обмеження використання бренду Android «несертифікованими» пристроями за межами їхньої екосистеми. Вихідний код використовувався для розробки варіантів Android на ряді іншої електроніки,

наприклад, ігрових приставок, цифрових камер, портативних медіа-плеєрів, ПК, кожен із яких має спеціалізований інтерфейс користувача. Деякі добре відомі деривативи включають Android TV для телевізорів і Wear OS для носимих пристроїв, обидва розроблені Google. Програми для Android зазвичай розповсюджуються у форматі APK через різноманітні магазини програм, такі як Google Play Store [13], Amazon Appstore [16], Samsung Galaxy Store [14], Huawei AppGallery [15], або платформи з відкритим вихідним кодом, такі як F-Droid [17].

2.2 Мова програмування Dart

Dart - це об'єктно-орієнтована мова програмування, розроблена компанією Google, яка дозволяє розробляти швидкі та ефективні кросплатформні застосунки [2]. Основна ідея Dart полягає в тому, щоб надати розробникам продуктивну мову, яка б сприяла швидкому процесу розробки та забезпечувала високу якість програмних рішень.

Одна з особливостей Dart - це його ефективна система типізації. Вона дозволяє визначати типи змінних і функцій, що полегшує виявлення помилок на етапі компіляції. Також Dart підтримує інференцію типів, що дозволяє розробникам необов'язково вказувати типи змінних, оскільки мова може самостійно їх визначати.

Dart має багатий набір функцій та бібліотек, які допомагають розробникам створювати складні програмні рішення. Особливо варто зазначити Flutter - фреймворк для розробки мобільних застосунків, який ґрунтується на мові Dart. Flutter дозволяє розробникам створювати красиві та ефективні застосунки для Android та iOS з використанням єдиного коду.

2.3 Фреймворк Flutter

Відкритий фреймворк розробки користувацького інтерфейсу Flutter [3] є продуктом компанії Google. Він використовує одну кодову базу для створення

кросплатформних мобільних та веб-застосунків. Основною метою Flutter є забезпечення швидкості розробки, гнучкості та високої продуктивності .

За даними з джерела Google Trends, можна спостерігати стрімкий розвиток популярності Flutter протягом останніх років [26]. Ця технологія набула значного інтересу і підвищеного попиту, що відображається в зростанні пошукових запитів на цю тему.

За період понад двох років, Flutter почав наздоганяти React Native і в квітні 2020 року став частіше шуканим запитом у всьому світі (на рисунку 2.1 зображено дані з 2018 по 2023 роки). З початку цього періоду і до теперішнього часу, популярність Flutter продовжує зростати і підтверджується високим інтересом користувачів.

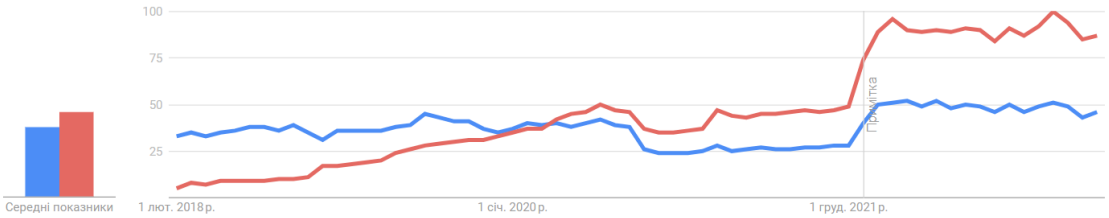

Рисунок 2.1 - Світові тенденції популярності Flutter (червоний) та React Native (блакитний) (2018–2023)

Можна зробити висновок про те, що Flutter вважається важливою та впливовою технологією в галузі мобільної розробки. Його переваги й потужні можливості привертають увагу розробників та підтримують стрімкий розвиток цієї платформи.

Унікальною особливістю архітектури Flutter є його здатність до самостійного малювання кожного пікселя, керування жестами та анімацією. На відміну від React Native, який використовує OEM-віджети, Flutter не підтримує такий підхід. Замість цього, розробники Flutter створили два набори віджетів, спеціально призначених для основних мобільних платформ: Material для Android та Cupertino для iOS. Таким чином, вони повторно відтворили всі компоненти

користувацького інтерфейсу з обох мобільних платформ, повністю зберігаючи їх поведінку. Пряма взаємодія з мобільною платформою (геолокація, звук, Bluetooth) здійснюється через Platform Channels (Рисунок 2.2) [28].

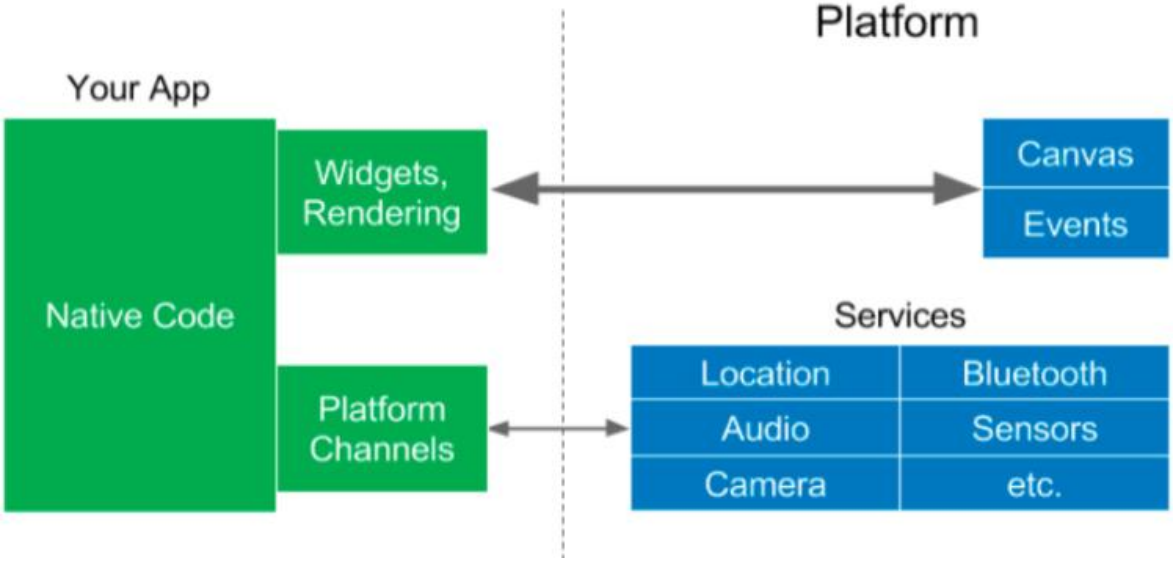

Рисунок 2.2 – Архітектура Flutter

Однією з головних переваг Flutter є "гаряче перезавантаження" (hot reload), що дозволяє розробникам миттєво бачити зміни, внесені в код, без необхідності повного перезапуску застосунку. Це значно прискорює процес розробки та дозволяє швидко експериментувати з різними ідеями та дизайном інтерфейсу [28].

Flutter має багато вбудованих віджетів (widgets), які дозволяють швидко створювати різноманітні елементи інтерфейсу. Віджети можна комбінувати та налаштовувати для створення складних інтерфейсів. Фреймворк також підтримує розширення за допомогою власних віджетів або використання сторонніх пакетів.

Фреймворк Flutter також надає можливості для розробки красивого та привабливого дизайну інтерфейсу. Він надає велику свободу в налаштуванні вигляду елементів інтерфейсу, включаючи кольори, шрифти, анімацію та переходи між екранами.

Вибір даного фреймворку для розробки застосунку “Math Boost” має кілька переваг. Перш за все, Flutter забезпечує одну кодову базу, що дозволяє

створювати застосунки для різних платформ (Android та iOS) з високим рівнем спільного коду. Це економить час та зусилля розробника, оскільки можна швидко розгортати та оновлювати застосунок на обох платформах.

Крім того, Flutter надає багатий набір вбудованих віджетів, які допомагають створювати привабливий та інтерактивний інтерфейс користувача. За допомогою цих віджетів розробник може легко візуалізувати математичні формули, завдання та інші компоненти, що використовуються в застосунку “Math Boost” .

У підсумку, використання Flutter для застосунку “Math Boost” дозволяє забезпечити швидку розробку, кросплатформну сумісність та привабливий інтерфейс.

2.4 Платформа Firebase

Firebase є комплексною платформою, розробленою компанією Google, яка надає розширений набір інструментів та сервісів для створення потужних та масштабованих застосунків (рисунок 2.3) [4], [29]. Основною перевагою Firebase є його простота використання та швидкість розробки.

Використання Firebase в процесі розробки застосунку “Math Boost” включало ключові функціональні можливості, такі як збереження даних та авторизація користувачів.

Збереження матеріалів:

Одна з основних використаних можливостей платформи Firebase - забезпечення збереження та керування матеріалами, необхідними для підготовки тестування з математики. Це включає теорію, приклади, вправи та тести. Завдяки хмарному зберіганню Firebase, ці матеріали є доступними для користувачів з різних пристроїв. Це дозволяє забезпечити швидкий та безперебійний доступ до актуальних навчальних матеріалів.

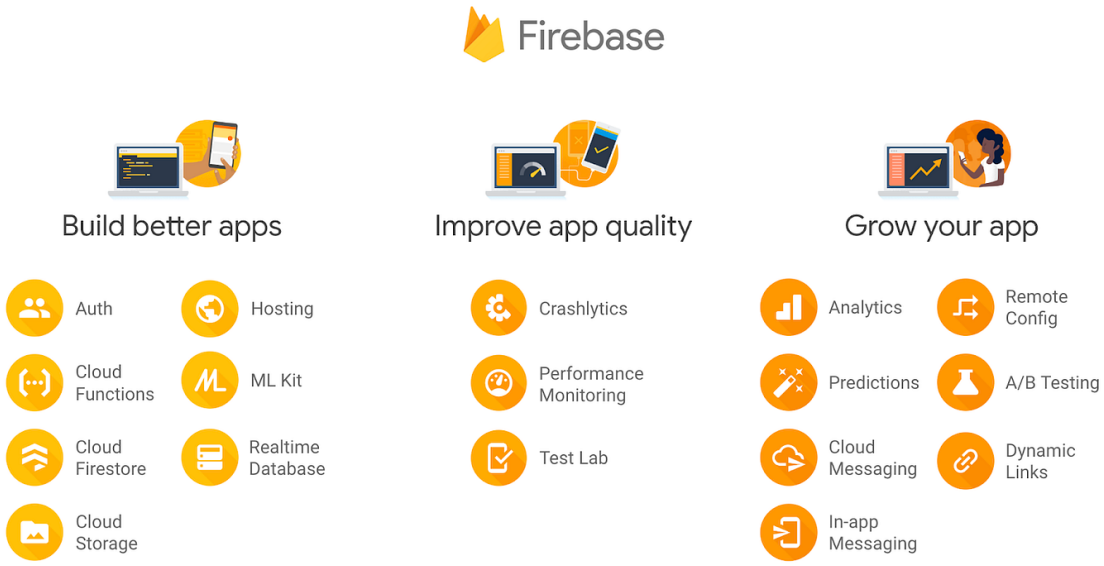

Рисунок 2.3 – Основні переваги та можливості використання Firebase

Авторизація користувачів:

Firebase використовувався для реалізації процесу аутентифікації користувачів у застосунку “Math Boost”. Зокрема, Firebase надав розширені інструменти для реєстрації та входу користувачів, включаючи можливість авторизації через Google-аккаунт. Це забезпечило зручну та безпечну ідентифікацію користувачів, а також дозволяло зберігати та управляти їхніми особистими налаштуваннями та досягненнями.

2.5 Огляд бази даних SQLite та об'єктно-реляційної проекції Floor

SQLite є реляційною базою даних, що використовується для створення таблиць, виконання запитів та збереження даних [30].

Переваги використання SQLite:

- Легкість використання та налаштування: SQLite є простою у використанні технологією. Вона не вимагає налаштування окремого серверу або встановлення додаткових компонентів. Це дозволяє швидко розпочати роботу з базою даних і спростити процес розробки.
- Ефективність та швидкість: SQLite пропонує високу продуктивність при виконанні операцій зчитування та запису даних. Вона використовує

компактну та оптимізовану архітектуру, що забезпечує ефективне використання ресурсів системи.

- Підтримка SQL: SQLite повністю підтримує мову структурованих запитів SQL. Це дозволяє розробникам зручно працювати з базою даних, використовуючи звичні SQL-оператори та функції, такі як SELECT, INSERT, UPDATE, DELETE.

- Портативність**:** SQLite доступний для використання на різних платформах, включаючи Windows, macOS, Linux та мобільні платформи. Це дає змогу розробникам побудувати крос-платформові рішення та забезпечити широку сумісність.

У контексті застосунку “Math Boost”, SQLite використовувався для зберігання навчальних матеріалів, інформації про користувачів та їх досягнення. Його перевагами є легкість використання, швидкість та ефективність.

Floor, як об'єктно-реляційна проекція (ORM), використовувався для спрощення взаємодії з базою даних SQLite. Це дозволило використовувати мову Dart для опису моделей даних та виконання запитів. Floor автоматично генерує SQL-запити на основі моделей даних, що спрощує роботу з даними та забезпечує зручний доступ до них [31].

Крім того, Floor надає можливість використовувати анотації для визначення зв'язків між таблицями, таких як один до одного, один до багатьох або багато до багатьох. Це спрощує взаємодію з даними, дозволяючи легко виконувати запити на отримання пов'язаних об'єктів даних.

Крім базового функціоналу ORM, Floor також надає можливості міграцій, що дозволяє змінювати схему бази даних, не втрачаючи існуючі дані.

Використання SQLite та Floor в застосунку “Math Boost” дозволило забезпечити надійне збереження та керування даними, що пов'язані з навчальним процесом та користувачами.

РОЗДІЛ 3. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

3.1 Вимоги до застосунку

Мова застосунку – українська.

Системні вимоги. Застосунок має коректно працювати і відображати інформацію на пристроях з ОС Android 10.0 і новіше.

Можливості користувача. У застосунку наявні дві ролі – гість та авторизований користувач.

Гість може подивитись наявні теми, переглянути теорію по темах, а також авторизуватись. В авторизованого користувача з’являється можливість проходити різноманітні тести, аналізувати свої результати на основі правильних відповідей, а також переглядати історію проходження тестів та рейтинг в профілі користувача.

Відповідно до цих вимог розроблено діаграму прецедентів (рисунок 3.1).

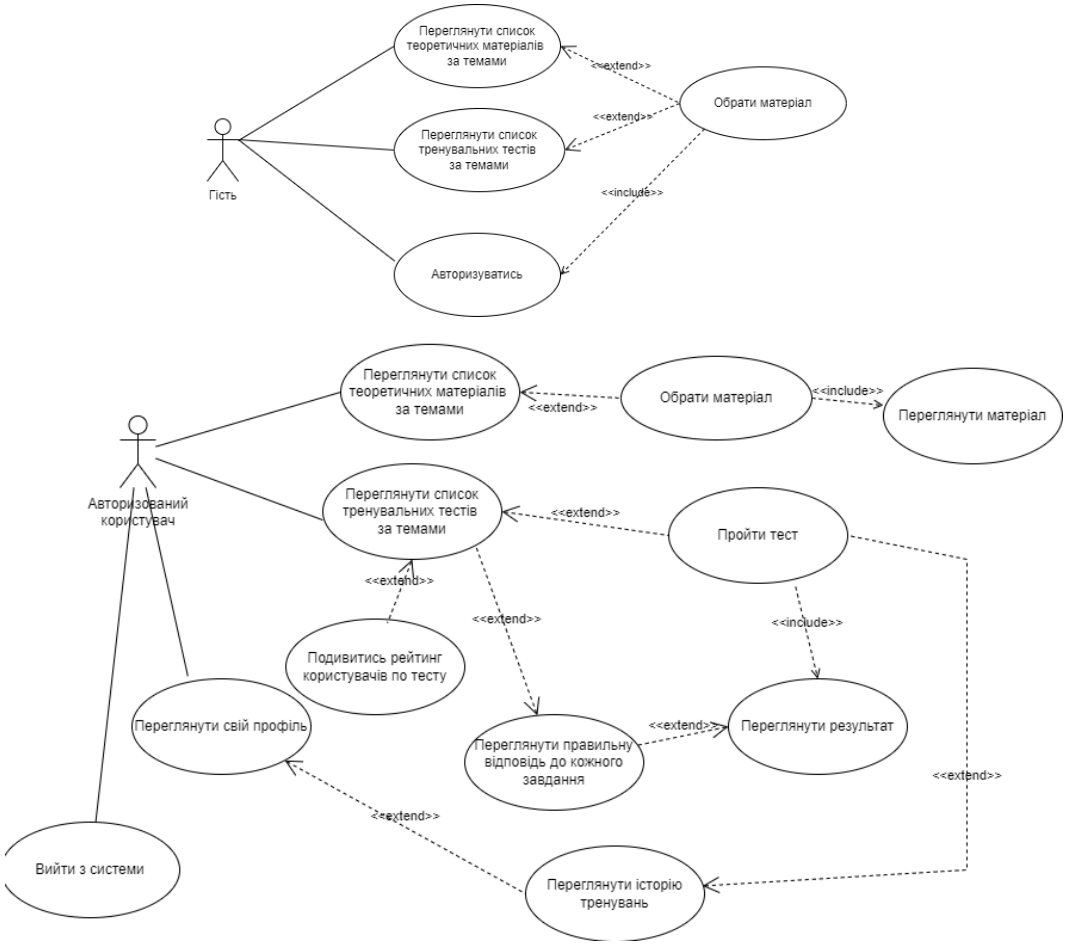

Рисунок 3.1 – Діаграма прецедентів

3.2 Створення бази даних застосунку

У якості системи керування базою даних було обрано SQLite. Розглянемо такі сутності: тема, теоретичний матеріал, тренувальний тест, питання, відповідь, результат, користувач, питання тесту, відповідь користувача, тип запитання.

Для візуалізації структури даних та зв'язків між ними зручно користуватись діаграмою БД (рисунок 3.2).

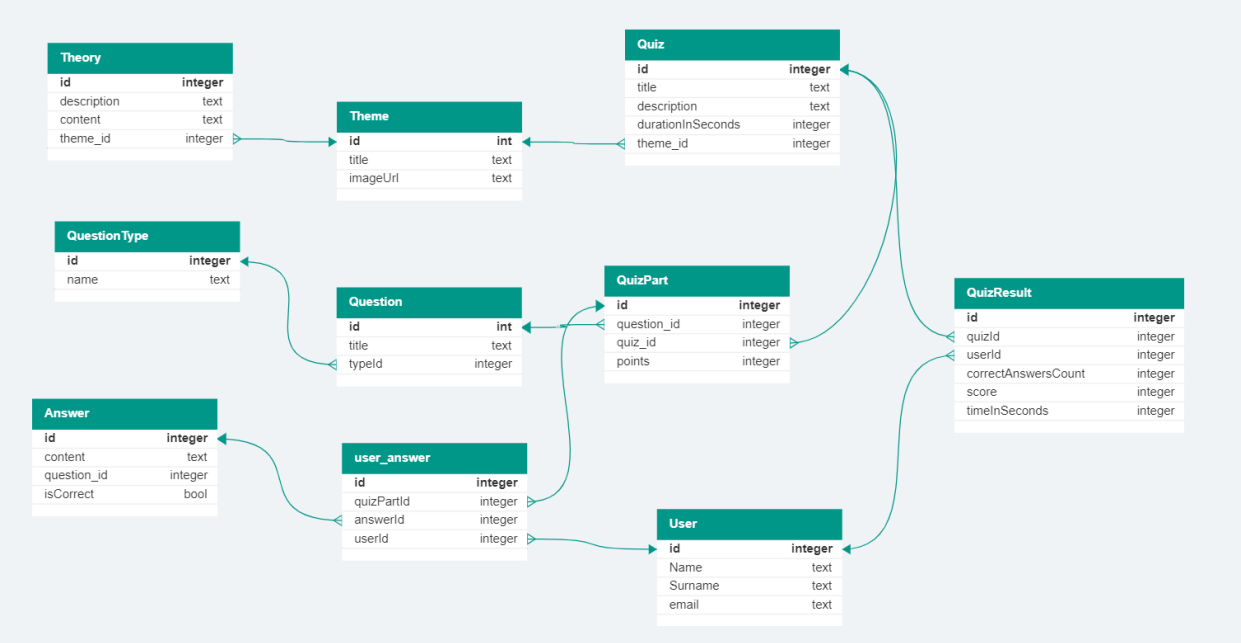

Рисунок 3.2 – Діаграма бази даних

У таблицях 3.1- 3.10 опишемо атрибути усіх сутностей.

Таблиця 3.1 – Опис таблиці "Theme"

Таблиця "Theme" Атрибут	Тип	Опис
Id	INTEGER	Ключовий атрибут
Title	TEXT	Назва теми
imageURL	TEXT	Посилання на картинку

Таблиця 3.2 – Опис таблиці "Theory"

Таблиця "Theory" Атрибут	Тип	Опис
Id	INTEGER	Ключовий атрибут
Description	TEXT	Короткий опис матеріалу

Content	TEXT	Зміст матеріалу
themeId	INTEGER	Зовнішній ключ, зв’язок з таблицею “Theme”

Таблиця 3.3 – Опис таблиці “Quiz”

Таблиця “Quiz” Атрибут	Тип	Опис
Id	INTEGER	Ключовий атрибут
themeId	INTEGER	Зовнішній ключ, зв’язок з таблицею “Theme”
Title	TEXT	Назва тесту
Description	TEXT	Короткий опис тесту
durationInSeconds	INTEGER	Час виконання тесту в секундах

Таблиця 3.4 – Опис таблиці “Question”

Таблиця “Question” Атрибут	Тип	Опис
Id	INTEGER	Ключовий атрибут
Content	TEXT	Зміст питання
typeId	INTEGER	Зовнішній ключ, зв’язок з таблицею “QuestionType”.

Таблиця 3.5 – Опис таблиці “QuestionType”

Таблиця “QuestionType” Атрибут	Тип	Опис
Id	INTEGER	Ключовий атрибут
Name	TEXT	Тип питання (тестове, з короткою відповіддю, з відкритою відповіддю).

Таблиця 3.6 – Опис таблиці “QuizPart”

Таблиця “QuizPart” Атрибут	Тип	Опис
Id	INTEGER	Ключовий атрибут
questionId	INTEGER	Зовнішній ключ, зв’язок з таблицею “Question”.
quizId	INTEGER	Зовнішній ключ, зв’язок з таблицею “Quiz”.
Points	INTEGER	Кількість балів за завдання

Таблиця 3.7 – Опис таблиці “Answer”

Таблиця “Answer” Атрибут	Тип	Опис
Id	INTEGER	Ключовий атрибут
Content	TEXT	Зміст відповіді
questionId	INTEGER	Зовнішній ключ, зв’язок з таблицею “Question”.
isCorrect	BOOL	Булеве значення, позначає чи вірна відповідь.

Таблиця 3.8 – Опис таблиці “User”

Таблиця “User” Атрибут	Тип	Опис
Id	INTEGER	Ключовий атрибут
Name	TEXT	Ім'я користувача
Surname	TEXT	Прізвище користувача
Email	TEXT	Електронна пошта користувача

Таблиця 3.9 – Опис таблиці “UserAnswer”

Таблиця “UserAnswer” Атрибут	Тип	Опис
Id	INTEGER	Ключовий атрибут
quizPartId	INTEGER	Зовнішній ключ, зв’язок з таблицею “QuizPart”.
answerId	INTEGER	Зовнішній ключ, зв’язок з таблицею “Answer”.
userId	INTEGER	Зовнішній ключ, зв’язок з таблицею “User”.

Таблиця 3.10 – Опис таблиці “QuizResult”

Таблиця “QuizResult” Атрибут	Тип	Опис
Id	INTEGER	Ключовий атрибут
quizId	INTEGER	Зовнішній ключ, зв’язок з таблицею “Quiz”.
userId	INTEGER	Зовнішній ключ, зв’язок з таблицею “User”.
correctAnswersCount	INTEGER	Кількість правильних відповідей
Score	REAL	Загальний бал

timeInSeconds	INTEGER	Витрачений час на проходження тесту в секундах

Узагальнюючи, описана база даних, відображає структуру, необхідну для ефективного управління навчальними ресурсами та пов'язаними з ними даними. Ця структура даних дозволяє зберігати та організовувати інформацію про теми, теоретичний матеріал, тренувальні тести, питання, відповіді, результати та користувачів. Використання такої бази даних допомагає забезпечити зручний доступ до навчальних ресурсів, швидке виконання запитів та ефективне управління навчальним процесом.

3.3 Створення класів

Розглянемо процес створення БД в застосунку "Math Boost" за допомогою бібліотеки Floor, яка є ORM для мови Dart.

Для початку, створимо сутність з назвою TheoryEntity, яка представлятиме запис теорії в БД (рисунок 3.3).

```
@Entity(tableName: theoryTableName)
class TheoryEntity {
  TheoryEntity({
    required this.id,
    required this.description,
    required this.content,
    required this.themeId,
  });

  @primaryKey
  @ColumnInfo(name: column.id)
  final String id;

  @ColumnInfo(name: column.description)
  final String description;

  @ColumnInfo(name: column.content)
  final String content;

  @ColumnInfo(name: column.themeId)
  final String themeId;
}
```

Рисунок 3.3 – Імплементація класу TheoryEntity

Наступним кроком є створення DAO (Data Access Object) для класу TheoryEntity (рисунок 3.4). DAO відповідає за виконання операцій з БД, таких як додавання, оновлення, видалення та запити на вибірку даних.

```
@dao
abstract class TheoryDao{
  static const tableName=theoryTableName;

  @Query('SELECT * FROM $tableName')
  Future<List<TheoryEntity>> getAllTheories();

  @Query('SELECT * FROM Person WHERE ${column.themeId} = :themeId')
  Future<List<TheoryEntity>> findTheoryByThemeId(int themeId);

  @insert
  Future<void> insertTheory(TheoryEntity theory);

  @insert
  Future<List<int>> insertTheories(List<TheoryEntity> theories);

  @Update(onConflict: OnConflictStrategy.replace)
  Future<void> updateTheory(TheoryEntity theory);

  @update
  Future<int> updateTheories(List<TheoryEntity> theory);

  @delete
  Future<void> deleteTheory(TheoryEntity theory);

  @delete
  Future<int> deleteTheories(List<TheoryEntity> theory);

}
```

Рисунок 3.4 – Імплементація класу TheoryDAO

У DAO використовується анотація @dao для позначення класу як Data Access Object. Далі, використовуючи анотації @Query, @insert, @update та @delete, визначаються методи для виконання запитів до БД.

Аналогічно імплементуються сутності AnswerEntity, QuestionEntity, QuizEntity, QuizPartEntity, QuizResultEntity, ThemeEntity, UserEntity, UserAnswerEntity та відповідні DAO для них.

Далі створимо абстрактний клас Database, який розширює клас FloorDatabase. В анотації @Database ми вказуємо версію БД (у даному випадку - 1) та перелік сутностей, які будуть зберігатись в БД (рисунок 3.5).

Визначимо абстрактні методи, що повертають DAO для кожної сутності.

Після визначення абстрактного класу Database та сутностей, необхідно запустити процес збірки, який згенерує відповідний код для БД. Цей процес генерації автоматично створить код для створення БД, включаючи таблиці, запити та взаємодію з даними.

```dart
@Database(version: 1, entities: [
  TheoryEntity,
  AnswerEntity,
  QuestionEntity,
  QuizEntity,
  QuizPartEntity,
  QuizResultEntity,
  ThemeEntity,
  UserEntity,
  UserAnswerEntity,
])
abstract class AppDatabase extends FloorDatabase {
  TheoryDao get theoryDao;

  AnswerDao get answerDao;

  QuestionDao get questionDao;

  QuizDao get quizDao;

  QuizPartDao get quizPartDao;

  ThemeDao get themeDao;

  UserDao get userDao;

  UserAnswerDao get userAnswerDao;

  QuizResultDao get quizResultDao;
}
```

Рисунок 3.5 – Імплементація класу AppDatabase

Створимо клас UploadDataRepository, який дозволяє отримувати дані з Firebase та локальної БД, а також здійснювати синхронізацію даних між цими джерелами (рисунок 3.6).

```dart
class UploadDataRepository {
  final FirebaseRepository _firebaseRepository;
  final AppDatabase _database;

  UploadDataRepository(this._firebaseRepository, this._database);

  Future<void> syncData() async {
    try {
      List<TheoryEntity> theories = await _firebaseRepository.getTheories();
      List<QuestionEntity> questions = await _firebaseRepository.getQuestions();
      List<AnswerEntity> answers = await _firebaseRepository.getAnswers();
      List<QuizEntity> quizzes = await _firebaseRepository.getQuizzes();
      List<QuizPartEntity> quizParts = await _firebaseRepository.getQuizParts();
      List<QuizResultEntity> quizResults = await _firebaseRepository.getQuizResultsByUserId();
      List<ThemeEntity> themes = await _firebaseRepository.getThemes();
      List<UserAnswerEntity> userAnswers = await _firebaseRepository.getUserAnswersByUserId();

      await _database.theoryDao.insertTheories(theories);
      await _database.questionDao.insertQuestions(questions);
      await _database.answerDao.insertAnswers(answers);
      await _database.quizDao.insertQuizes(quizzes);
      await _database.quizPartDao.insertQuizParts(quizParts);
      await _database.quizResultDao.insertQuizResults(quizResults);
      await _database.themeDao.insertThemes(themes);
      await _database.userAnswerDao.insertUserAnswers(userAnswers);
    } catch (e) {
      AppLogger.e(e);
    }
```

Рисунок 3.6 – Імплементація класу UploadDataRepository

Метод syncData() виконує процес синхронізації даних: спочатку отримує дані з Firebase (theories, questions, answers), а потім зберігає їх у відповідні таблиці в локальній БД.

Інші методи цього класу, наприклад getTheoriesByThemeId та getQuestionsByIds, перевіряють параметр getFromNetwork, який вказує, чи потрібно отримати дані з мережі чи з локальної БД. В залежності від значення цього параметра, методи викликають відповідні методи з FirebaseRepository або з локальної БД для отримання даних (рисунок 3.7).

Якщо виникають помилки під час виконання запитів, вони логуються за допомогою класу AppLogger і повертаються пусті списки.

```
Future<List<TheoryEntity>> getTheoriesByThemeId(int themeId, bool getFromNetwork) async {
  try {
    if (getFromNetwork) {
      return await _firebaseRepository.getTheoriesByTheme(themeId);
    }
    return await _database.theoryDao.findTheoryByThemeId(themeId);
  } catch (e) {
    AppLogger.e(e);
    return [];
  }
}

Future<List<QuestionEntity>> getQuestionsByIds(List<int> questionsIds, bool getFromNetwork) async {
  try {
    if(getFromNetwork){
      return await _firebaseRepository.getQuestionsByIds(questionsIds);
    }
    return await _database.questionDao.findQuestionsByIds(questionsIds);
  } catch (e) {
    AppLogger.e(e);
  return [];
}
```

Рисунок 3.7 – Імплементація класу UploadDataRepository

GetX - це популярна бібліотека управління станом та навігацією для фреймворка Flutter. Вона надає зручні інструменти для ефективної роботи зі станом застосунку, а також прості та потужні функції для навігації між екранами [32].

Одним з ключових елементів GetX є GetxController, який використовується для управління станом та забезпечення реактивності застосунку. За допомогою спостережуваних змінних (Rx-змінних) та методу update() контролер може автоматично оновлювати відповідні частини інтерфейсу при зміні стану.

GetX також надає зручні методи навігації, такі як Get.to() та Get.back(), які дозволяють легко переміщатися між екранами застосунку. Крім того, вона пропонує різні типи діалогових вікон, які можна викликати за допомогою методів Get.

Розглянемо імплементацію контролера QuizController (Код контролера надано в додатку А).

Контролер QuizController використовується для керування станом тестування. Його основна функціональність полягає у завантаженні даних тесту, керуванні поточним питанням та відстеженні часу, залишеного до завершення.

Після створення екземпляра контролера, він ініціалізується залежністю від UploadDataRepository, який використовується для отримання даних зі сховища.

При готовності контролера (метод onReady), відбувається завантаження необхідних даних, включаючи питання та варіанти відповідей. Якщо завантаження пройшло успішно, дані присвоюються відповідним змінним контролера. Якщо завантаження не вдалося, відображається помилка.

Контролер має методи для перемикання між питаннями, перевірки, чи поточне питання є першим або останнім, а також вибору відповіді. Також, контролер відстежує час, залишений для завершення квізу, та оновлює його значення відповідно до таймера.

Контролер також містить деякі додаткові функції, такі як обробка виходу з тестування, перехід до наступного питання, завершення, повторне проходження тестування та перехід до головного екрану.

Застосування контролера QuizController у застосунку “Math Boost” дозволяє ефективно керувати станом тестування та забезпечити користувачеві зручний інтерфейс для проходження тесту з математики.

РОЗДІЛ 4. ІНТЕРФЕЙС КОРИСТУВАЧА

UI/UX (User Interface/User Experience) дизайн включає в себе набір принципів та практик, спрямованих на задоволення потреб користувачів при взаємодії з мобільним застосунком. UI описує, як інтерфейс виглядає, включаючи дизайн, кольори, шрифти та компоненти, тоді як UX зосереджується на взаємодії користувача з інтерфейсом, його зручності та ефективності [33].

При розробці мобільного застосунку важливо врахувати вимоги до якісного UX/UI дизайну. Це включає:

Ясність: Інтерфейс повинен бути чітким і зрозумілим для користувача. Текст, іконки та структура повинні допомагати користувачеві досягти своєї мети без зайвих запитань.

Лаконічність: Інтерфейс має бути простим і економним у використанні. Зайві підказки, спливаючі вікна та анімація можуть заважати користувачу. Важливо зосередитися на основних елементах інтерфейсу, щоб уникнути перевантаження.

Пізнаваність: Елементи дизайну мають бути легко розпізнаваними навіть для нового користувача. Використання стандартних патернів та елементів інтерфейсу допомагає забезпечити зрозумілість та зручність використання.

Чутливість: Інтерфейс повинен бути чутливим до дій користувача та забезпечувати миттєву зворотну відповідь. Користувач повинен отримувати візуальні та аудіо сигнали, які підтверджують успішне виконання дії, наприклад, підтвердження надсилання повідомлення або завершення процесу оплати. Крім того, інтерфейс повинен надавати інформацію про поточний стан застосунку, наприклад, процес завантаження або обробки даних.

Сталість: Важливо дотримуватись єдиної стилістики та поведінки елементів інтерфейсу на всіх сторінках та розділах застосунку. Це забезпечить консистентність та прогнозованість для користувача.

Естетика: Створення візуально привабливого інтерфейсу є важливим аспектом. Гармонійне поєднання кольорів, використання простору та зворотний зв'язок забезпечать приємний візуальний досвід для користувача.

Ефективність: Інтерфейс повинен бути ефективним і заощаджувати час користувача. Мінімізація кількості кроків та чітка організація функцій допомагають користувачеві швидко досягти своєї мети [33].

4.1 Сторінка авторизації

Основна функція сторінки – надати користувачам можливість зручно авторизуватись у застосунку.

У центрі сторінки розміщено зображення логотипу застосунку “MathBoost”, що допомагає користувачеві ідентифікувати застосунок. Під ним розташований текстовий блок, який пояснює користувачеві суть застосунку та його переваги.

Також на сторінці присутня кнопка "Увійти через Google", яка дозволяє користувачеві авторизуватись за допомогою облікового запису Google (рисунок 4.1). Кнопка має привабливий дизайн та контрастний колір.

Рисунок 4.1 – Сторінка авторизації

Усі елементи сторінки відповідають загальному стилю застосунку “MathBoost”, використовують гармонійну кольорову палітру, що забезпечує читабельність тексту та видимість елементів інтерфейсу.

4.2 Сторінка вибору тесту за темою

За замовчуванням користувачу на головній сторінці застосунку пропонується попрактикуватись за однією з обраних тем (рисунок 4.2).

Ця сторінка містить привітання, що створює особистий зв'язок з користувачем та додає персоналізованості застосунку.

На сторінці присутній список тестів за вибраною темою. Кожен тест представлений у вигляді картки, що містить інформацію про тему, кількість завдань, час відведений на проходження тесту. Для того щоб обрати тест, користувачу потрібно натиснути на відповідну картку (рисунок 4.2).

Для забезпечення актуальності списку тестів на сторінці присутня можливість оновлення. Користувач може оновити список, потягнувши його вниз.

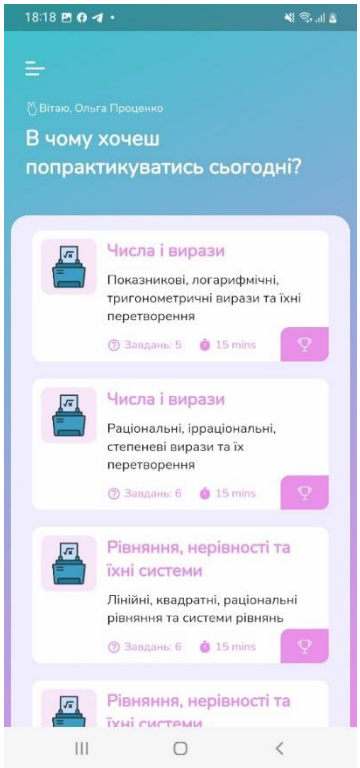

Рисунок 4.2 – Головна сторінка

Також головна сторінка містить спеціальне бічне висувне меню, яке можна відкрити, натиснувши на відповідну іконку у верхньому лівому куті застосунку (рисунок 4.3).

У цьому меню можна побачити такі елементи інтерфейсу як:

- Кнопка автентифікації. Якщо користувач не увійшов у систему, він може натиснути на цю кнопку, щоб увійти. Якщо користувач уже увійшов у систему, то замість кнопки відображається його фото профілю. При натисканні на фото профілю відбувається перехід на сторінку профілю користувача.
- Кнопка переключення між режимами "Тести" та "Теорія". При натисканні на цю кнопку відбувається зміна режиму вибору тестів та теоретичних матеріалів. Назва кнопки залежить від поточного режиму, в якому знаходиться застосунок. Після натискання кнопки відбувається переключення режиму та закриття бічного меню.
- Кнопка виходу. При натисканні на цю кнопку користувач виходить зі свого облікового запису.

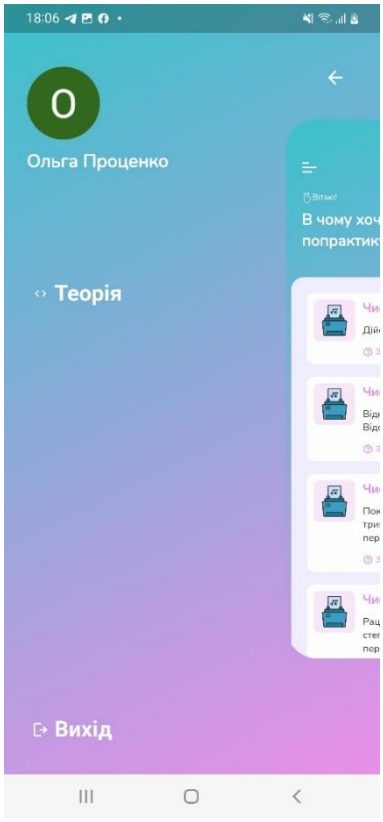

Рисунок 4.3 – Бічне меню

4.3 Сторінка тестування

Верхня частина екрану містить панель, яка складається з заголовка та таймера. Таймер відображає залишок часу для виконання тесту.

Основна частина екрану містить питання та варіанти відповідей. Питання відображається у вигляді тексту з математичними формулами, якщо такі є. Для питань з вибором однієї правильної відповіді, варіанти відповідей представлені у вигляді списку з можливістю вибору. Для питань з короткою відповіддю присутнє текстове поле, куди користувач може ввести свою відповідь.

Внизу екрану розташовані кнопки для навігації між питаннями. Якщо поточне питання є першим, відображається лише кнопка "Далі", яка дозволяє перейти до наступного питання. Якщо поточне питання не є першим, з'являються дві кнопки: "Назад" і "Далі". Кнопка "Назад" дозволяє перейти до попереднього питання, а кнопка "Далі" - до наступного питання. Після останнього питання кнопка "Далі" змінює свій текст на "Завершити" і веде до сторінки з оглядом даних відповідей, для остаточного завершення тестування (рисунок 4.4).

Рисунок 4.4 – Сторінки тестування

4.4 Сторінка з оглядом тестування

Верхня частина екрану містить панель з заголовком, який відображає назву пройденого тесту.

Основна частина екрану містить список питань, які були включені в тест. Кожне питання представлене у вигляді номера питання (індексу) та статусу відповіді. Статус може бути одним з таких: "відповідь дано" (рожевий колір), "відповідь не дано" (рожевий, майже прозорий). Користувач може натиснути на будь-яке питання, щоб перейти до нього.

Кнопка "Завершити": Внизу екрану розташована кнопка "Завершити", яка дозволяє остаточно завершити тест та переглянути результати (рисунок 4.5).

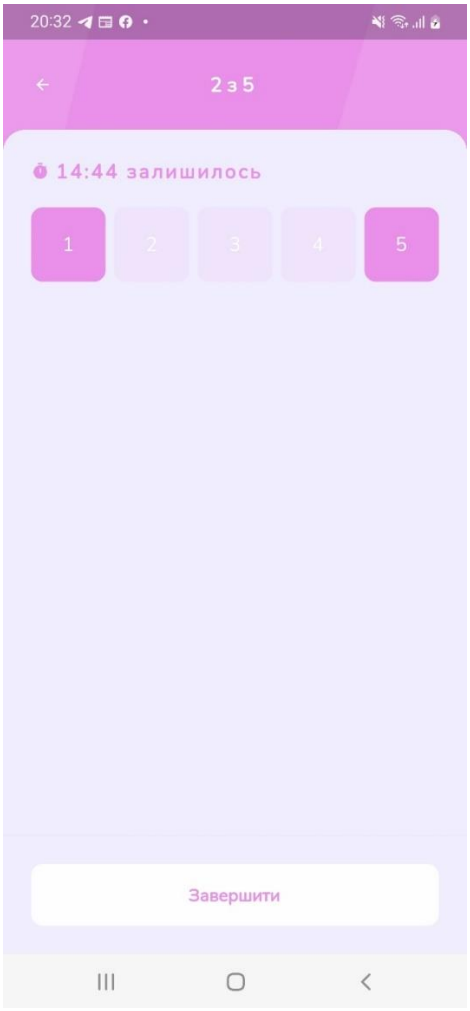

Рисунок 4.5 – Сторінка огляд тестування

4.5 Сторінка з результатом тестування

Верхня частина екрану містить панель з заголовком, який відображає кількість правильних відповідей.

Основна частина екрану містить ілюстрацію, що символізує успішне завершення тесту. Під ілюстрацією відображаються привітання до користувача та кількість отриманих балів за тест.

Нижче відображається сітка з номерами питань, які були включені в тест. Кожне питання позначене своїм номером та кольором. Якщо відповідь на питання була дана правильно, воно виділяється зеленим кольором, якщо користувач пропустив питання або дав неправильну відповідь – рожевим. Користувач може натиснути на будь-яке питання, щоб переглянути правильну відповідь.

Внизу екрану розташовані дві кнопки. Перша кнопка "Спробувати ще раз" дозволяє користувачу повторити тест знову. Друга кнопка "На головну" перенаправляє користувача на головний екран застосунку (рисунок 4.6)

Рисунок 4.6 – Сторінка з результатами тестування

4.6 Сторінка з правильною відповіддю на запитання

Основна частина сторінки візуально схожа на сторінку проходження тесту. Але тут немає таймеру, оскільки це етап перевірки відповідей, а не активного проходження тесту. І варіанти відповідей тут відображаються у вигляді картки з певним статусом. Статуси включають "правильна відповідь" (зелений колір), "неправильна відповідь" (рожевий колір) та "інша відповідь" (кольором не виділяється). Це дозволяє користувачеві швидко знайти та опрацювати свої помилки (рисунок 4.7).

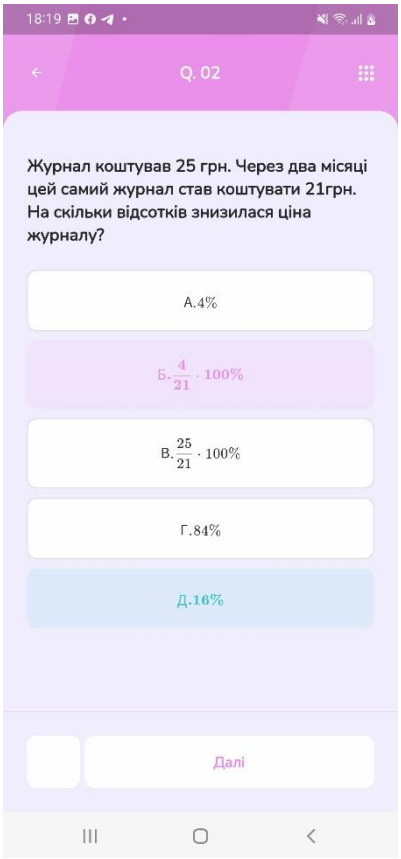

Рисунок 4.7 – Сторінка з правильною відповіддю

4.7 Сторінка з теоретичним матеріалом

Сторінка відображає теоретичний матеріал з обраної теми (рисунок 4.8). Основний інтерфейс користувача складається з наступних елементів:

- Верхня панель, яка містить назву обраної теми. На панелі також присутня кнопка навігації "Назад", яка повертає користувача на попередній екран.

- Головна область екрану, де відображається теоретичний матеріал. За допомогою прокрутки можна переглянути весь зміст. Теоретичний матеріал відображається у форматі тексту та картинок, які відповідають обраній темі.

Інтерфейс сторінки простий і мінімалістичний, щоб забезпечити фокус користувача на читанні та сприяти легкому сприйняттю теоретичного матеріалу.

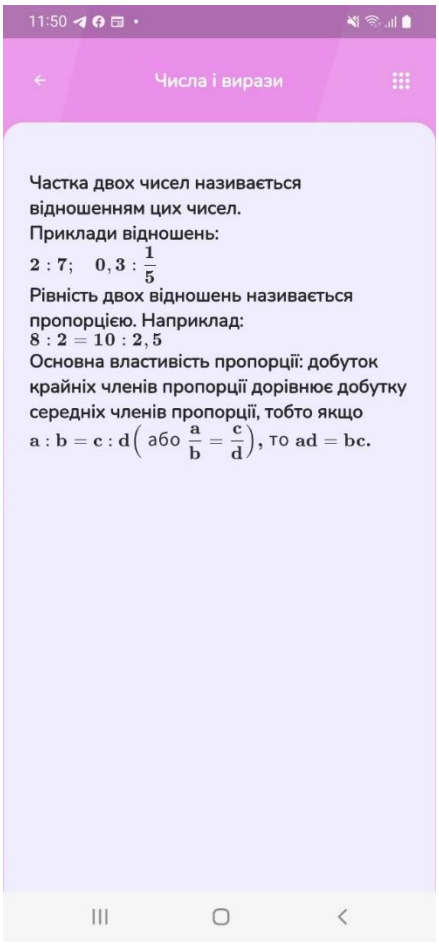

Рисунок 4.8 – Сторінка з теоретичним матеріалом

4.8 Сторінка профілю користувача

Сторінка містить дані профілю користувача, зокрема його фото та ім'я.

Також тут відображається список нещодавніх тестів, які користувач проходив. Кожен елемент списку відображається у вигляді картки зі значеннями тесту, такими як назва тесту, результат, час проходження тощо (рисунок 4.9).

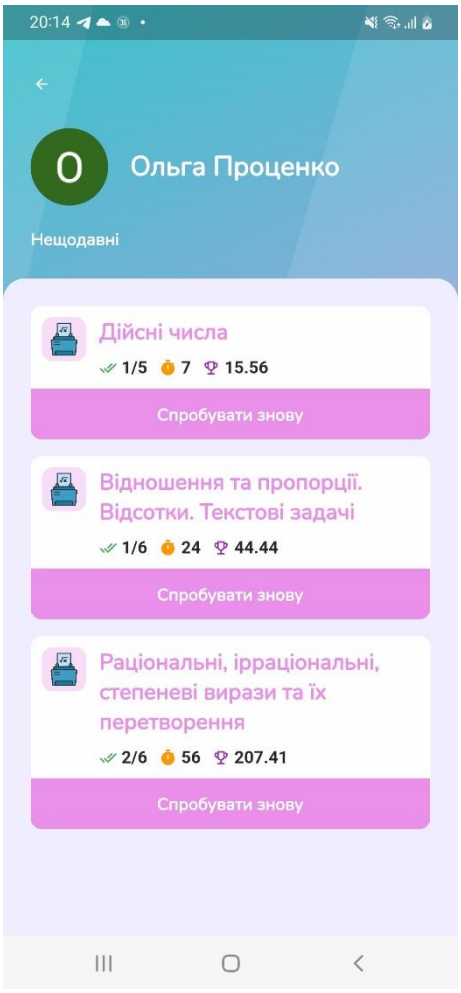

Рисунок 4.9 – Сторінка профіль користувача

Додаток "Math Boost" надає зручний та ефективний інтерфейс для покращення математичних навичок користувачів. Він пропонує чіткі та зрозумілі елементи, що допомагають користувачам досягти своїх цілей без зайвих запитань. Простота та лаконічність дизайну сприяють швидкому орієнтуванню та ефективному використанню додатку. Стандартні паттерни та легко впізнавані елементи інтерфейсу сприяють зручності використання, навіть для нових користувачів.

ВИСНОВКИ

У результаті виконання даної роботи було розроблено функціональний застосунок для підготовки користувачів до тестування з математики.

Відповідно поставленої мети, виконано наступні завдання:

- проведено аналіз предметної області з метою оцінки перспектив розробки;
- проведено аналіз подібних застосунків;
- обрано інструменти розробки;
- розроблено діаграми прецедентів, бази даних та класів;
- розроблено мобільний застосунок.

Застосунок реалізований з використанням мови програмування Dart та фреймворку Flutter, що забезпечує високу продуктивність, кросплатформну сумісність та зручний інтерфейс користувача. “Math Boost” надає користувачам широкий спектр можливостей для вивчення математики, розв'язування завдань та підготовки до тестування. Застосунок забезпечує доступ до теоретичного матеріалу, практичних завдань та тестів з різних розділів математики.

Завдяки своїм функціональним можливостям, “Math Boost” може бути використаний в різних сферах, таких як освіта, самостійне навчання, підготовка до тестування та індивідуальне самовдосконалення. Застосунок дозволяє користувачам ефективно вивчати математику, систематизувати свої знання та підготуватися до тестування з математики на високому рівні.

У майбутньому планується розширення функціональності та покращення можливостей “Math Boost”. Серед можливих напрямків розвитку можуть бути реалізація додаткових розділів математики, вдосконалення системи відстеження прогресу користувача, розширення бази завдань та тестів, а також впровадження інтерактивних елементів та спільного навчання для покращення взаємодії між користувачами.

Загалом, мобільний застосунок “Math Boost” може стати корисним інструментом для покращення математичних навичок та підвищення рівня підготовки користувачів. Окрім основних функціональних можливостей, таких

як теорія, приклади, вправи та тести, “Math Boost” також має інтерфейс, що нагадує гру-вікторину. Це може допомогти зняти психологічний тиск від навчання та зробити процес вивчання матеріалу більш захопливим і заохотити користувачів до активної участі. Ігрова форма навчання в “Math Boost” сприяє кращому засвоєнню матеріалу та підвищенню мотивації до навчання.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Філіпс Б. Android Programming: The Big Nerd Ranch Guide / Б. Філіпс, Б. Харді., 2013.
2. Buckett C. Dart in Action / Chris Buckett., 2013.
3. Zaccagnino C. Programming Flutter Native, Cross-Platform Apps the Easy Way / Carmine Zaccagnino., 2020.
4. Firebase [Електронний ресурс] – Режим доступу до ресурсу: https://firebase.google.com/ .
5. MOBILE APPLICATION: DEFINITION, TECHNOLOGY TYPES AND EXAMPLES [Електронний ресурс]. – 2021. – Режим доступу до ресурсу: https://magenest.com/en/mobile-application/.
6. A Brief History of Mobile Apps [Електронний ресурс]. – 2021. – Режим доступу до ресурсу: https://www.captechu.edu/blog/brief-history-of-mobile-apps .
7. Schaffer E. What is Swift? Features, advantages, and syntax basics [Електронний ресурс] / Erin Schaffer – Режим доступу до ресурсу: https://www.educative.io/blog/swift-programming.
8. Jalli A. What Is Objective C? [Електронний ресурс] / Artturi Jalli – Режим доступу до ресурсу: https://builtin.com/software-engineering-perspectives/objective-c.
9. Дейтел П. Android для розробників / П. Дейтел, Х. Дейтел., 2016
10. Vaguez L. Kotlin vs. Java for Android development [Електронний ресурс] / Levi Vaguez – Режим доступу до ресурсу: https://blog.logrocket.com/kotlin-vs-java-android-development/ .
11. Shiotsu Y. What Is a Hybrid App? (Detailed Guide for 2023) [Електронний ресурс] / Yoshitaka Shiotsu – Режим доступу до ресурсу: https://www.upwork.com/resources/hybrid-app .
12. Kenton W. Apple App Store [Електронний ресурс] / Will Kenton – Режим доступу до ресурсу: https://www.investopedia.com/terms/a/apple-app-store.asp .

13. Hindy J. Google Play Store: A definitive guide for beginners [Електронний ресурс] / Joe Hindy – Режим доступу до ресурсу: https://www.androidauthority.com/google-play-store-1093442/ .
Samsung Galaxy Store [Електронний ресурс] – Режим доступу до ресурсу: https://help.branch.io/using-branch/page/samsung-galaxy-store .
14. Samsung Galaxy Store Overview [Електронний ресурс] – Режим доступу до ресурсу: https://help.branch.io/using-branch/page/samsung-galaxy-store.
15. HUAWEI AppGallery [Електронний ресурс] – Режим доступу до ресурсу: https://consumer.huawei.com/ua/mobileservices/appgallery/ .
16. What is Amazon App Store? [Електронний ресурс] – Режим доступу до ресурсу: https://nandbox.com/what-is-amazon-app-store/ .
17. Fedewa J. What Is F-Droid and How Is It Different From the Play Store? [Електронний ресурс] / Joe Fedewa – Режим доступу до ресурсу: https://www.howtogeek.com/790674/what-is-f-droid-and-how-is-it-different-from-the-play-store/ .
18. 2013 Mobile Future in Focus [Електронний ресурс]. – 2013. – Режим доступу до ресурсу: https://www.comscore.com/Insights/Presentations-and-Whitepapers/2013/2013-Mobile-Future-in-Focus .
19. Ринок мобільного трафіку: підсумки року та тренди 2023 [Електронний ресурс]. – 2023. – Режим доступу до ресурсу: https://palai.media/news/mobile-traffic-report-2022/ .
20. Rajput M. Tracing the History and Evolution of Mobile Apps [Електронний ресурс] / Mehul Rajput. – 2021. – Режим доступу до ресурсу: https://tech.co/news/mobile-app-history-evolution-2015-11
21. Android 13 Highlights [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: android.com/android-13/
22. ЗНО: Математика [Електронний ресурс] – Режим доступу до ресурсу: https://play.google.com/store/apps/details?id=com.copynets.znomathapp.
23. ЗНО 2023. Математика [Електронний ресурс] – Режим доступу до ресурсу: https://play.google.com/store/apps/details?id=com.hardworkingdev.mathapp.

24. Математика: формули + тести [Електронний ресурс] – Режим доступу до ресурсу: https://play.google.com/store/apps/details?id=com.netchargesoftware.mathformulas
25. Wankhede C. What are Google Mobile Services (GMS)? [Електронний ресурс] / Calvin Wankhede. – 2023. – Режим доступу до ресурсу: https://www.androidauthority.com/google-mobile-services-gms-3025963/.
26. Google Trends. React Native VS Flutter [Електронний ресурс]. – 2023. – Режим доступу до ресурсу: https://trends.google.com/trends/explore?date=2018-01-02%202023-05-15&q=%2Fg%2F11h03gfxy9,%2Fg%2F11f03_rzbg .
27. React Native vs Flutter. Огляд архітектур [Електронний ресурс] / Закіра. – 2021. – Режим доступу до ресурсу: https://dou.ua/forums/topic/34042/.
28. Губін А. Що таке Flutter та які його особливості [Електронний ресурс] / Андрій Губін – Режим доступу до ресурсу: https://highload.today/uk/shho-take-flutter-ta-yaki-jogo-osoblivosti/.
29. Що таке Firebase? [Електронний ресурс] – Режим доступу до ресурсу: https://avada-media.ua/ua/services/firebase/.
30. What is SQLite? [Електронний ресурс] – Режим доступу до ресурсу: https://www.codecademy.com/article/what-is-sqlite.
31. Floor Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://pinchbv.github.io/floor/ .
32. Khan S. GetX State Management In Flutter [Електронний ресурс] / Shaiq Khan – Режим доступу до ресурсу: https://medium.flutterdevs.com/getx-state-management-in-flutter-a9710277b0bc
33. What is UI design? A complete introductory guide [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://www.uxdesigninstitute.com/blog/what-is-ui-design/.

ДОДАТОК А

Імплементація контроллера QuizController

```
class QuizController extends GetxController {
  final loadingStatus = LoadingStatus.loading.obs;
  final allQuestions = <Question>[];
  late QuizPaperModel quizPaperModel;
  late ThemeModel themeModel;
  Timer? _timer;
  int remainSeconds = 1;
  final time = '00:00'.obs;

  final UploadDataRepository _repository;

  QuizController(this._repository);

  @override
  void onReady() {
    final quizPaper = Get.arguments as QuizPaperModel;
    loadData(quizPaper);
    super.onReady();
  }

  @override
  void onClose() {
    _timer?.cancel();
    super.onClose();
  }

  Future<bool> onExitOfQuiz() async {
    return Dialogs.quizEndDialog();
  }
```

```
void _startTimer(int seconds) {
  const duration = Duration(seconds: 1);
  remainSeconds = seconds;
  _timer = Timer.periodic(
    duration,
        (Timer timer) {
      if (remainSeconds == 0) {
        timer.cancel();
      } else {
        int minutes = remainSeconds ~/ 60;
        int seconds = remainSeconds % 60;
        time.value = "${minutes.toString().padLeft(2, "0")}:${seconds.toString().padLeft(2, "0")}";
        remainSeconds--;
      }
    },
  );
}
```

```
void loadData(QuizPaperModel quizPaper) async {
  quizPaperModel = quizPaper;
  loadingStatus.value = LoadingStatus.loading;
  try {
    final questions = await _repository.getQuestionsForQuiz(quizId,true);
    for (var question in questions) {
      final answers = await _repository.getAnswersForQuestion(question.id, true);
      question.answers = answers;
    }
  } catch (e) {
    RegExp exp = RegExp(r'permission-denied', caseSensitive: false);
    if (e.toString().contains(exp)) {
      AuthController _authController = Get.find();
      Get.back();
      _authController.showLoginAlertDialog();
    }
    AppLogger.e(e);
    loadingStatus.value = LoadingStatus.error;
  }

  if (quizPaper.questions != null && quizPaper.questions!.isNotEmpty) {
    allQuestions.assignAll(quizPaper.questions!);
    currentQuestion.value = quizPaper.questions![0];
    _startTimer(quizPaper.timeSeconds);
    loadingStatus.value = LoadingStatus.completed;
  } else {
    loadingStatus.value = LoadingStatus.noResult;
  }
}
```

```
}

Rxn<Question> currentQuestion = Rxn<Question>();
final questionIndex = 0.obs; //_curruntQuestionIndex

bool get isFirstQuestion => questionIndex.value > 0;

bool get isLastQuestion => questionIndex.value >= allQuestions.length - 1;

void nextQuestion() {
  if (questionIndex.value >= allQuestions.length - 1) return;
  questionIndex.value++;
  currentQuestion.value = allQuestions[questionIndex.value];
}

void prevQuestion() {
  if (questionIndex.value <= 0) return;
  questionIndex.value--;
  currentQuestion.value = allQuestions[questionIndex.value];
}
```