Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра комп’ютерних наук

Пояснювальна записка

до магістерської кваліфікаційної роботи
на тему:

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ОЦІНЮВАННЯ ЯКОСТІ ЗНАНЬ

Виконав: студент 2 курсу, гр. 1КН-19м
спеціальності

122 «Комп'ютерні науки

Британський В.А.
(прізвище та ініціали)

Керівник к.т.н., ст. викл. Озеранський В.С.
(прізвище та ініціали)
Рецензент д.т.н., проф. Романюк О.Н.
(прізвище та ініціали)

Вінниця - 2020 року

ЗАТВЕРДЖУЮ
Ректор ПВНЗ ВІКОП
Мізрах А.А. .
(наук. ст., вч. зв., ініц. та прізв.) (підпис)
«____»_____2020 р.

ЗАТВЕРДЖУЮ
Завідувач кафедри КН
д. т. н., проф. Яровий А. А.
_____
(підпис)
“____” _____2020 року

ЗАВДАННЯ

на магістерську кваліфікаційну роботу на здобуття кваліфікації магістра зі спеціальності: 122 - «Комп’ютерні науки»

08-22.МКР.002.19.000.ПЗ

Магістранта групи 1КН-19м Британського Владислава Андрійовича

Тема магістерської кваліфікаційної роботи: «Інформаційна технологія оцінювання якості знань»

Вхідні дані: операційна система Android з рівнем API не нижче 19, операційна система Windows 10 чи Linux, середовище розробки, об’єктно-орієнтована мова програмування.

Короткий зміст частин магістерської кваліфікаційної роботи:

1. Графічна: алгоритм роботи інформаційної технологія оцінювання якості знань, структурна схема інформаційної технологія оцінювання якості знань, діаграма класів, результати тестування додатку №1, результати тестування додатку №2, результати тестування додатку №3.

2. Текстова (пояснювальна записка): вступ, аналіз предметної області, розробка інформаційної технологія оцінювання якості знань, програмна реалізація інформаційної технологія оцінювання якості знань, тестування та аналіз результатів роботи технології., економічна частина, висновки, перелік використаних джерел, додатки.

КАЛЕНДАРНИЙ ПЛАН

№ етапу	Назва етапу	Термін виконання		Очікувані результати
		початок	кінець	
1	Обґрунтування доцільності розробки інформаційної технології оцінювання якості знань			Аналітичний огляд літературних джерел, задачі досліджень, розділ 1 ПЗ
2	Аналіз параметрів платформ та ПЗ яке використовується для реалізації інформаційної технології			розділ 2
3	Програмна реалізація розробленої інформаційної технології, тестування та оцінка параметрів			розділ 3
4	Підготовка економічної частини			розділ 4
5	Апробація та/або впровадження результатів дослідження			тези доповідей/акт впровадження
6	Оформлення пояснювальної записки, графічного матеріалу та презентації			Пояснювальна записка, графічний матеріал, презентація

Консультанти з окремих розділів магістерської кваліфікаційної роботи

1. Науковий керівник _____ к.т.н., ст. викладач кафедри КН
(підпис) наук. ступінь, вчене звання (посада)
"___" _____20__ р. _____ В.С. Озеранський
ініціали та прізвище

2. Економічна частина _____ к.е.н, доцент кафедри ЕПВМ
(підпис) наук. ступінь, вчене звання (посада)
"___" _____20__ р. М.В. Бальзан
ініціали та прізвище

Дата попереднього захисту роботи "___" _____20__ р.

Рецензент _____ д.т.н., проф. зав. кафедри ПЗ
(підпис) наук. ступінь, вчене звання (посада)
О.Н. Романюк
ініціали та прізвище

Завдання видав
науковий керівник _____ к.т.н., ст. викладач кафедри КН
(підпис) наук. ступінь, вчене звання (посада)
В.С. Озеранський
ініціали та прізвище

"___" _____20__ р.
Завдання отримав магістрант _____ В.А. Британський
(підпис) ініціали та прізвище
"___" _____20__ р

АНОТАЦІЯ

Магістерська кваліфікаційна робота присвячена розробці інформаційної технології оцінювання якості знань.

В ході роботи обґрунтовано доцільність розробки, проведено аналіз переваг та недоліків програмних засобів у своїх категоріях. Обрано оптимальні програмні засоби що будуть використані при проектуванні та програмній реалізації інформаційної технології оцінювання якості знань.

Здійснено програмну реалізацію інформаційної технології оцінювання якості знань в інтегрованому середовищі розробки Android Studio. Здійснено тестування розробленої інформаційної технології оцінювання якості знань та проаналізовано його роботу.

ABSTRACT

Master's thesis is devoted to the development of information technology for assessing the quality of knowledge.

In the course of the work the expediency of development is substantiated, the analysis of advantages and disadvantages of software in the categories is carried out. The optimal software tools that will be used in the design and software implementation of information technology for assessing the quality of knowledge are selected.

The software implementation of information technology for assessing the quality of knowledge in the integrated development environment of Android Studio has been implemented. The developed information technology for assessing the quality of knowledge was tested and its work was analyzed.

ЗМІСТ

ВСТУП

Актуальність теми дослідження. Світ технологій розвивається безперервно і динамічно. Із активним розвитком комп`ютерних технологій не менш активно розвивається мобільні пристрої, а в свою чергу мобільні додатки.

Розвиток і використання інформаційних технологій викликав необхідність перегляду старих уявлень щодо методики викладання, практики проведення навчальних курсів, тощо.

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 знань.

Зв'язок роботи з науковими програмами, планами, темами. Магістерська робота виконана відповідно до напрямку наукових досліджень кафедри комп’ютерних наук Вінницького національного технічного університету 22 К1 «Розробка спеціалізованих засобів штучного інтелекту на основі інтелектуального аналізу даних та машинного навчання» та плану наукової та навчально-методичної роботи кафедри.

Мета та завдання дослідження. Метою дослідження магістерської кваліфікаційної роботи є зменшення часу для створення тестових завдань та ймовірності недобросовісного проходження тесту.

Для досягнення поставленої мети необхідно вирішити такі задачі:

1. проаналізувати переваги та недоліки аналогів програм оцінювання якості знань;
2. удосконалити інформаційну технологію оцінювання якості знань, яка відрізняється від аналогів тим що тести не вводяться вручну і вони не є однотипні, що дає перевагу в зменшені можливого недобросовісного проходження тесту через фактичне запам’ятовування завдань;
3. розробити структуру мобільного додатку оцінювання якості знань;
4. розробити алгоритм функціонування мобільного додатку оцінювання якості знань;
5. розробити інтерфейс мобільного додатку оцінювання якості знань;
6. здійснити програмну реалізацію інформаційної технології оцінювання якості знань;
7. здійснити тестування розробленого програмного продукту та аналіз результатів роботи.

Об’єкт дослідження - процес оцінювання якості знань.

Предметом дослідження є програмні засоби для оцінювання якості знань.

Наукова новизна одержаних результатів полягає в наступному:

- Удосконалено інформаційну технологію оцінювання якості знань, що відрізняється від відомих аналогів побудовою тестових завдань на основі понятійно-тезисних елементів, що зменшує затрати часу на створення тестових завдань та збільшує якість оцінювання набутих знань.

Практичне значення одержаних результатів полягає у наступному:

- Розроблено алгоритм побудови тестових завдань на основі понятійно-тезисних елементів, який дозволяє зменшити затрати часу на розробку самих тестів та зменшує можливість їх недобросовісного проходження.

Особистий внесок магістранта. Результати даної магістерської кваліфікаційної роботи отримані самостійно. В публікації у співавторстві здобувачу належить аналіз можливостей реалізації мобільного додатка для реалізації технології оцінювання якості знань.

Апробація результатів роботи. Результати роботи були апробовані на конференції XLIX науково-технічній конференції підрозділів ВНТУ [2].

1 ОБҐРУНТУВАННЯ ДОЦІЛЬНОСТІ РОЗРОБКИ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ОЦІНЮВАННЯ ЯКОСТІ ЗНАНЬ

1.1 Аналіз предметної області інформаційної технології 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 інформаційної технології оцінювання якості знань, було обрано створити мобільний додаток. Адже, на сьогодні мобільні технології створюють підґрунтя для використання нових способів комунікації: соціальної, освітньої, наукової.

Інформаційну технологію оцінювання якості знань за допомого мобільних додатків можна використовувати для:

– дистанційності 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 аналогів інформаційної технології оцінювання якості знань базі мобільних додатків

Існує певна кількість мобільних додатків які надають змогу оцінити якість знань, але не завжди вони повною мірою виконують функції, потрібні користувачеві. Для розробки додатку з оптимальним набором функцій проаналізовано переваги та недоліки найвідоміших аналогів.

Основним конкурентом є додаток QuizUp [5] – найбільш простий і цікавий додаток. Він має інтуїтивно простий дизайн і зрозумілий користувачеві набір функцій (рисунок 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 хороший додаток, яким можна користуватися разом з друзями. Тест можна зробити і відправити друзям (рисунок 1.2). Особливість цього додатку полягає в тому, що вікторини також можна зберігати.

DK Quiz [7] – додаток має більше 100 тестів, що охоплюють безліч тем, таких як історія, наука або їжа (рисунок 1.3). Тести доступні в різних темах, і чим швидше відповідь, тим більше балів можна отримати. Добре записані результати з відсотками за виграшами і програшів. Однак нові оновлення не надають нові функції або функціональні можливості.

Рисунок 1.2 - Додаток Quiz Your Friends

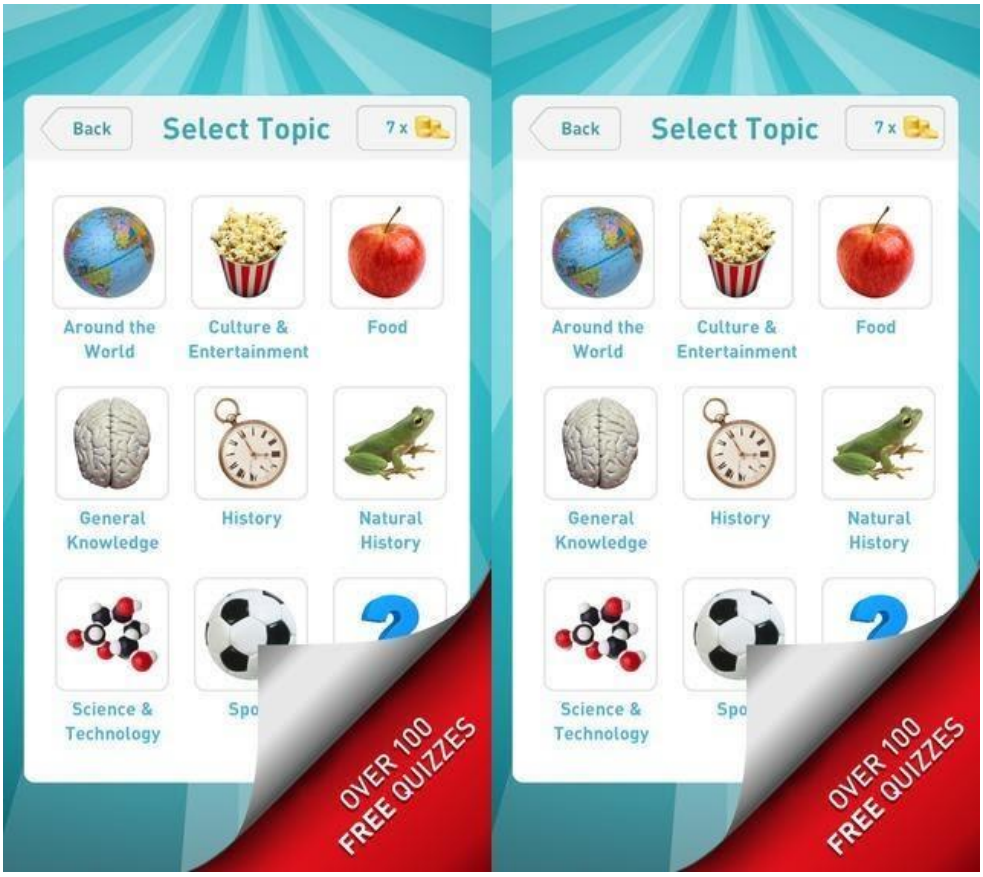

Рисунок 1.3 - Додаток DK Quiz

Переваги і недоліки аналогів розробки наведено в таблиці 1.1.

Таблиця 1.1 – Порівняння програм аналогів оцінювання якості знань

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

1.3 Постановка вимог до інформаційної технології оцінювання якості знань

Інформаційна технологія 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 інформаційої технології:

– 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 інформаційної технології:

– 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

Під час роботи над розділом обґрунтування доцільності розробки інформаційної технології оцінювання якості знань, проаналізовано предметну область, були визначені функціональні і нефункціональні вимоги до додатка.

Виконано порівняння існуючих додатків оцінювання знань. В ході аналізу об’єкту проектування було визначено вимоги до інформаційної технології.

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

2 РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ОЦІНЮВАННЯ ЯКОСТІ ЗНАНЬ

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:

1. 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.

2. 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).

3. 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 Аналіз середовищ розробки

Створення програмного забезпечення здійснюється за допомогую 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 [23]. 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 [21].

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.

2.4 Аналіз наборів засобів розробки інформаційної технології оцінювання якості знань

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 (таб. 2.1), на рисунку 2.1 відображений перелік програмних рішень для розробки додатків з відкритим вихідним кодом.

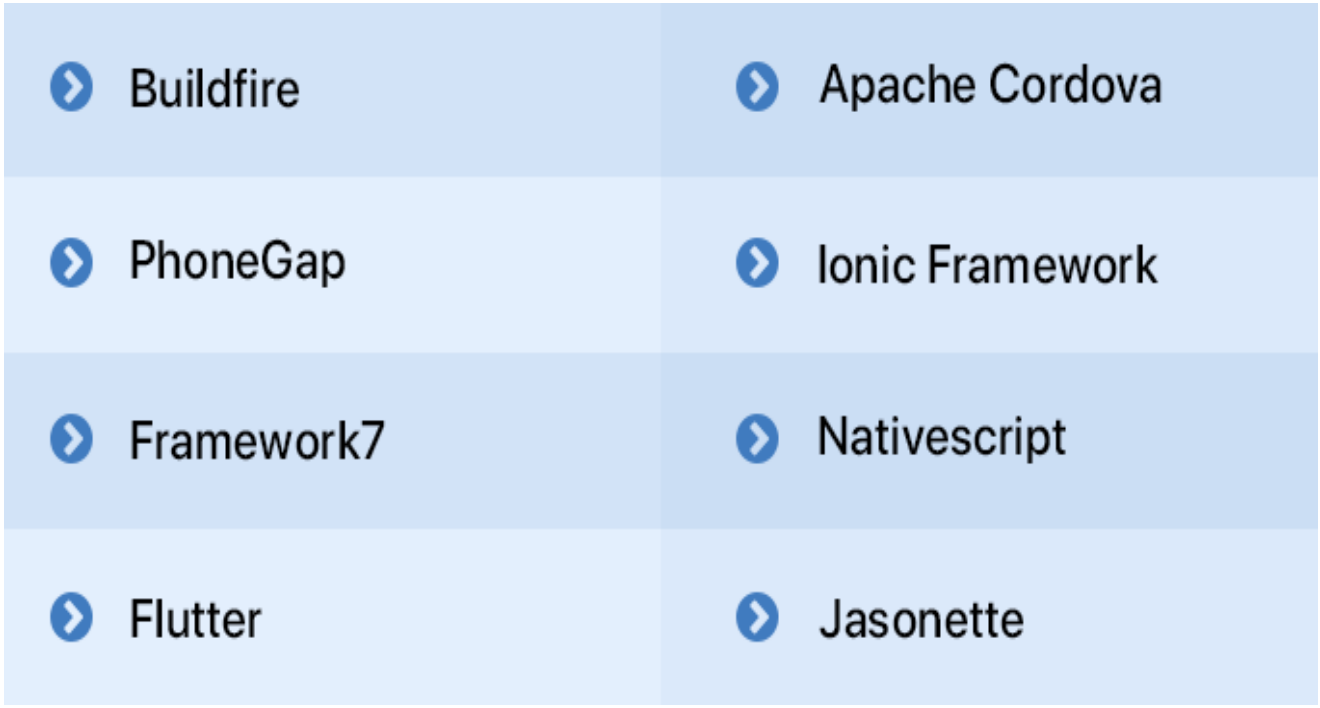

Рисунок 2.1 - Програмні рішення для розробки додатків з відкритим вихідним кодом

На таблиці 2.1 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 інформаційної технології побудови оцінювання якості знань.

Таблиця 2.1 - 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 • Інструменти для спільної роботи • Тестування сумісності • Моделювання даних • Звітність та аналітика • Управління версіями • Розробка веб-додатків
Apache Cordova	Windows, Mac, Linux	Open API	• Контроль доступу / дозволів • Інструменти для спільної роботи • Управління розгортанням програмного забезпечення • Звітність та аналітика • Тестування сумісності • Управління версіями
PhoneGap	Windows, Mac, Linux	Open API, Cloud Hosted	• Контроль доступу / дозволів • Сумісне тестування • Управління розгортанням • Звітність і аналітика • Налагодження програми •Розробка програмного забезпечення • Розробка веб-додатків • Управління версіями
Ionic Framework	Windows, Mac, Linux	Open API, Cloud Hosted	• Контроль доступу / дозволів • Інструменти співпраці • Моделювання даних • Звітування та аналітика • Розробка без коду • Сумісне тестування • Розробка програмного забезпечення

Продовження Таблиці 2.1

Програмні рішення для розробки додатків	Підтримувані пристрої	Розгортання програмного забезпечення	Особливі можливості
Framework7	Windows, Mac, Linux	Open API	• Контроль доступу / дозволів • Рефакторинг коду • Розробка без коду • Інструменти співпраці • Налагодження програми • Розробка програмного забезпечення • Розробка веб-додатків
NativeScript	Windows, Mac, Linux	Open API, Cloud Hosted	• Контроль доступу / дозволів • Рефакторинг коду • Розробка без коду • Тестування сумісності • Налагодження програми • Управління розгортанням • Звітування та аналітика • Розробка програмного забезпечення • Управління версіями • Розробка веб-додатків
Jasonette	Windows, Mac, Linux	Open API	• Контроль доступу / дозволів • Рефакторинг коду • Розробка без коду • Інструменти співпраці • Налагодження програми • Управління версіями

Продовження Таблиці 2.1

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

1. Buildfire - це повністю гнучкий до налаштування, безкоштовний, простий та інтуїтивно зрозумілий засіб для створення додатків для Android, iOS та прогресивних веб-програм [30]. Програмне забезпечення забезпечує елегантний інтерфейс (рисунок 2.2) 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.

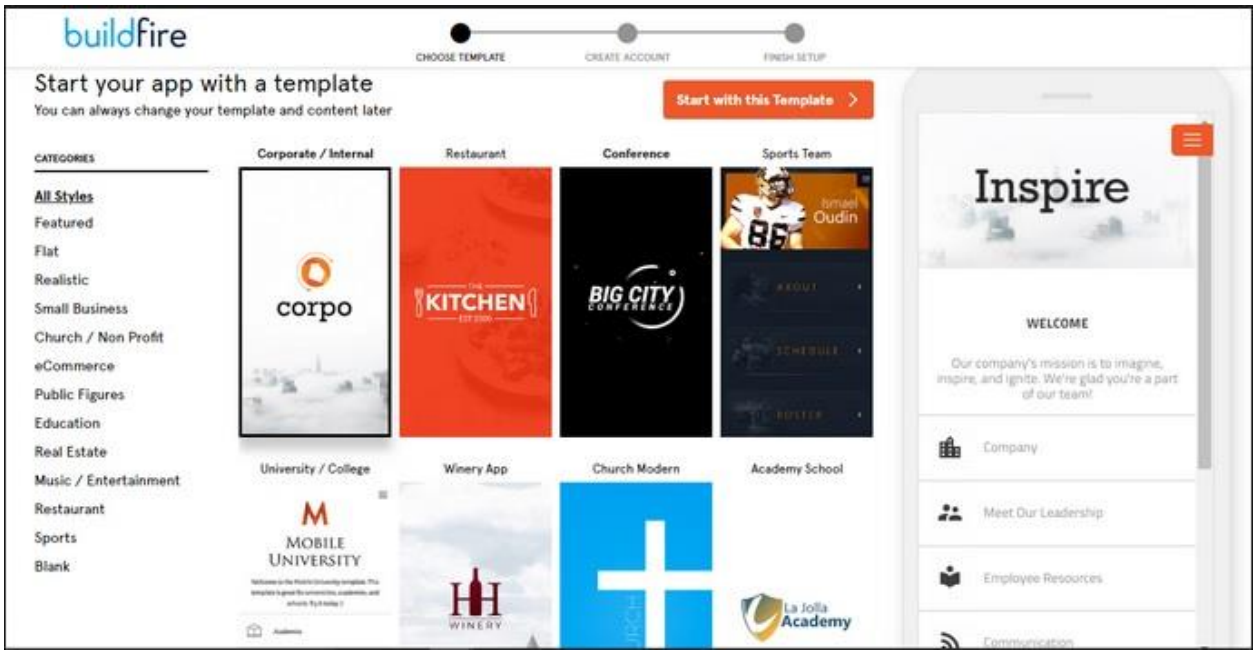

Рисунок 2.2 - Інтерфейс Buildfire

2. 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 [31]. 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 2.3 - 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.

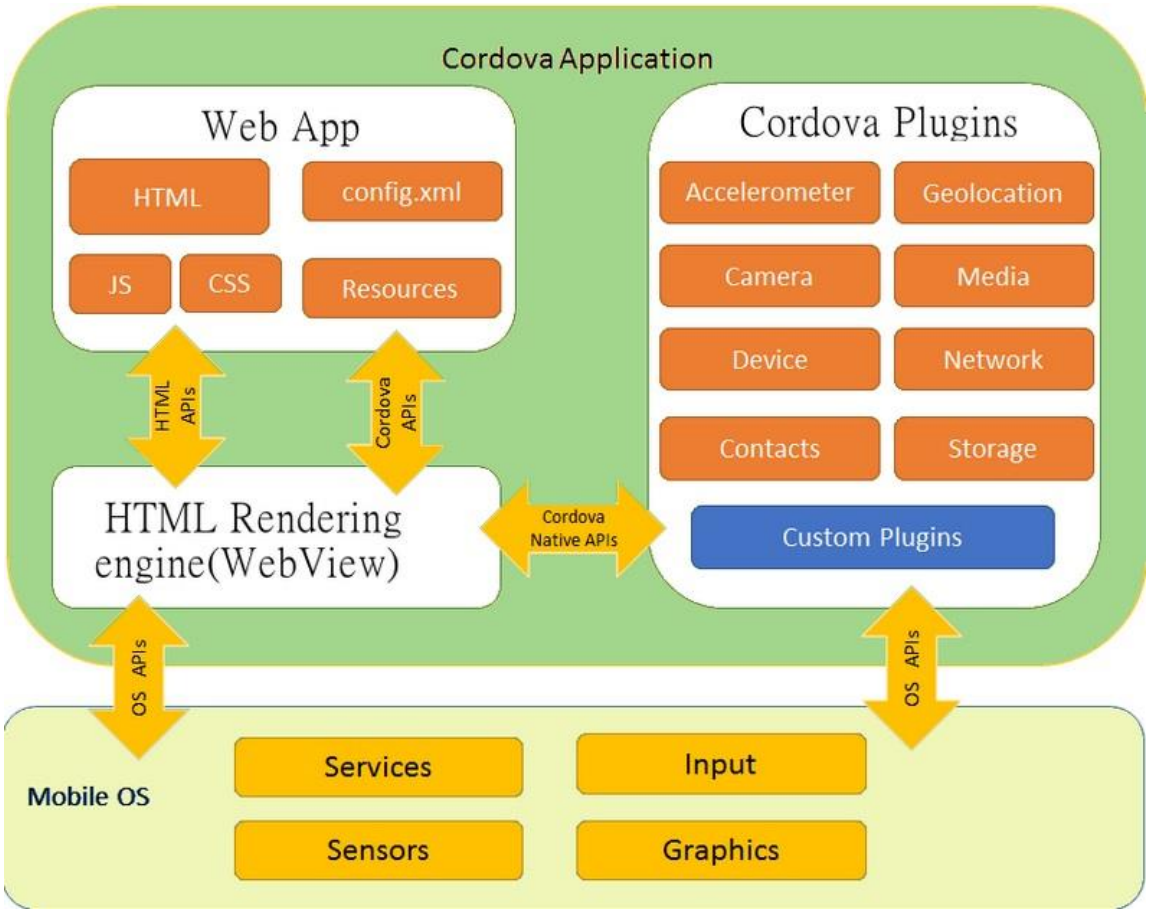

Рисунок 2.3 - принцип роботи додатку Apache Cordovав

3. PhoneGap - безкоштовний open-source фреймворк для створення мобільних додатків [32], 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.

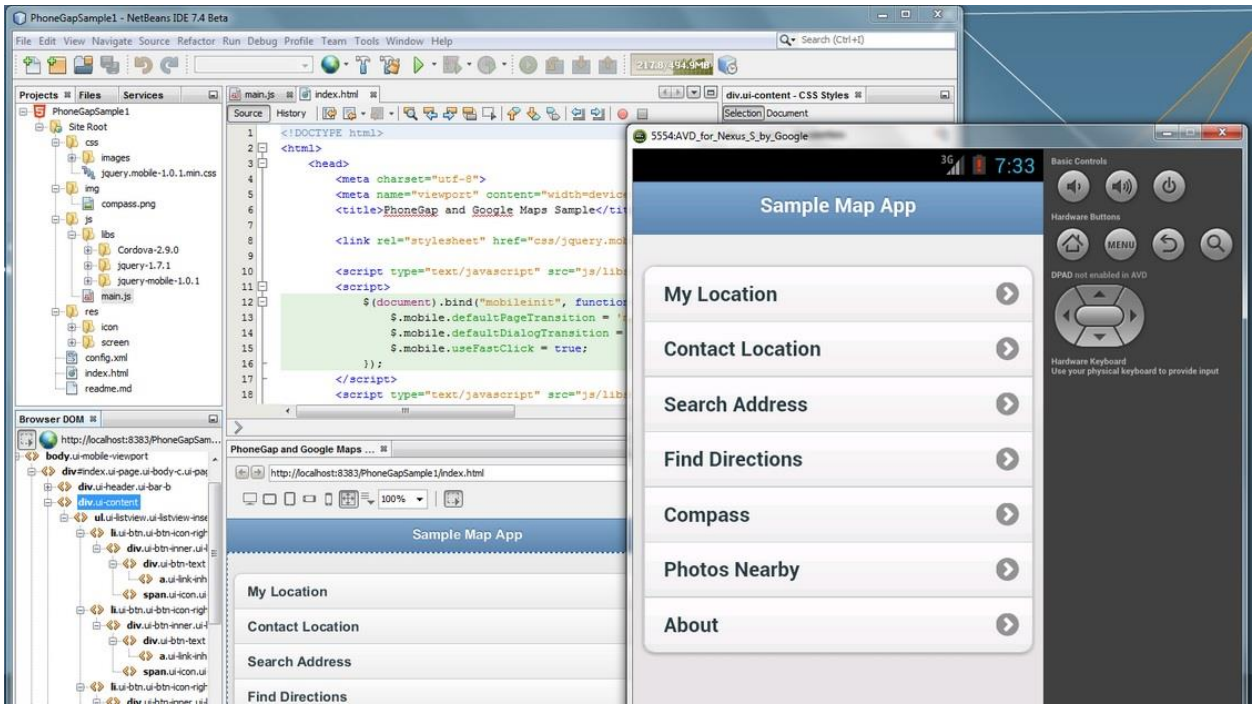

Рисунок 2.4 - Інтерфейс PhoneGap

4. 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 2.5. Цей фреймворк також дозволяє використовувати іонічні компоненти, які взагалі не мають інтерфейсу користувача [33]. 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 [39]. 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.

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.

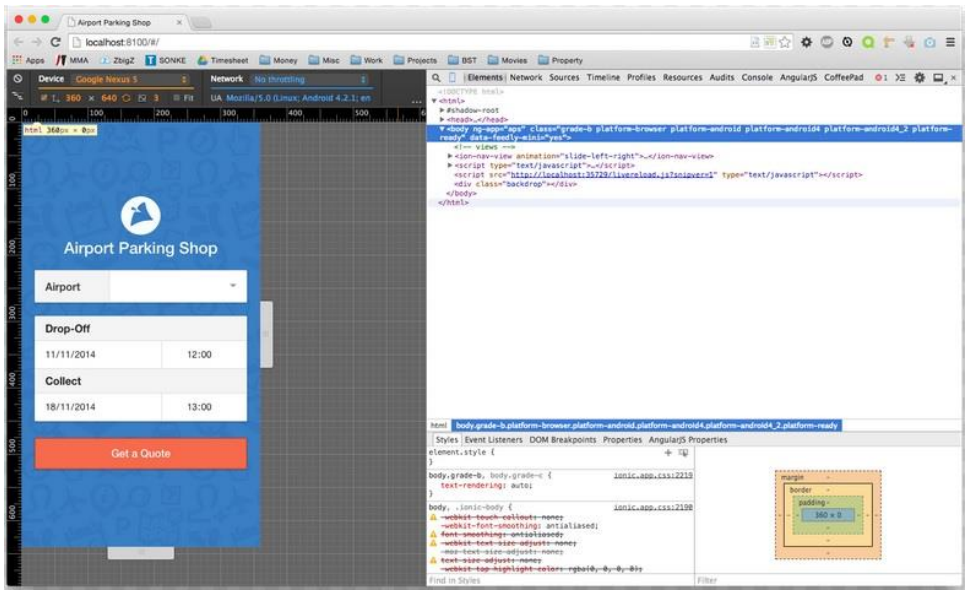

Рисунок 2.5 – Інтерфейс вихідного коду інтерфейсу користувача та його зображення яке розроблене за допомогою Ionic Framework

5. Framework7 - це безкоштовна платформа з відкритим вихідним кодом для мобільного HTML, інтерфейс якого зображено на рисунку 2.6. Він використовується для розробки гібридних мобільних додатків або веб-додатків для пристроїв iOS і Android.

Основні особливості платформи Framework7 [34]:

- 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.

6. NativeScript - це фреймворк з відкритим вихідним кодом, розроблений компанією Telerik, для розробки додатків на платформах Android та iOS, на рисунку 2.7 зображено інтерфейс мобільного додатку створеного за допомогою NativeScript [35]. 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.

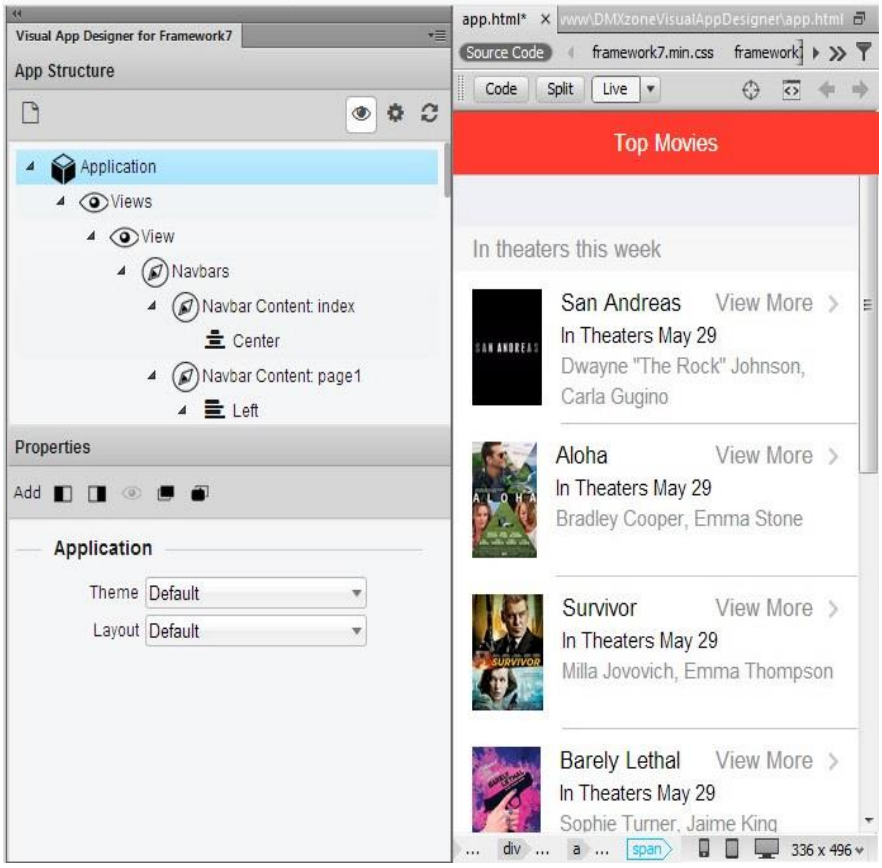

Рисунок 2.6 – Інтерфейс мобільного додатку створеного за допомогою Framework7

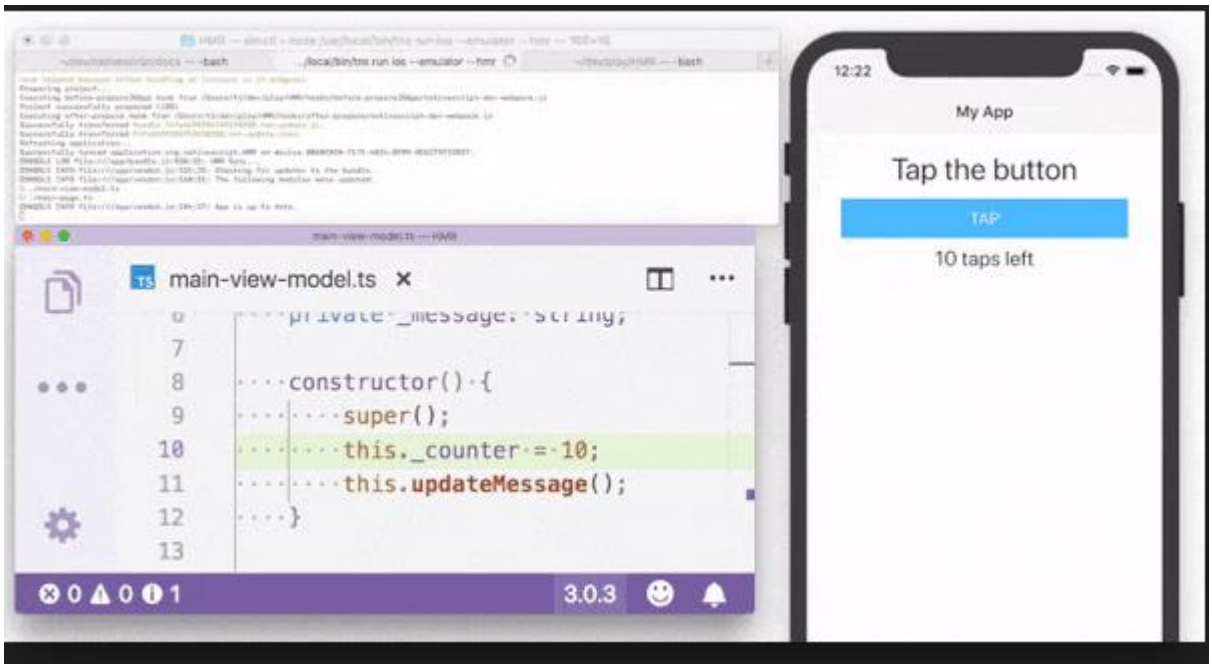

Рисунок 2.7 – Інтерфейс мобільного додатку створеного за допомогою NativeScript

7. Jasonette - це платформа програмування високого рівня, яка використовує розмітку на основі JSON для розробки додатків (рис. 2.8) для iOS та Android [36]. Основа програми – браузер який використовує JSON [34], він бере файли з сервера та перетворює їх у власні елементи.

Рисунок 2.8 – 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.

8. Flutter - SDK з відкритим вихідним кодом для створення мобільних додатків від компанії Google [37]. Він використовується для розробки додатків (рис. 2.9) 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.

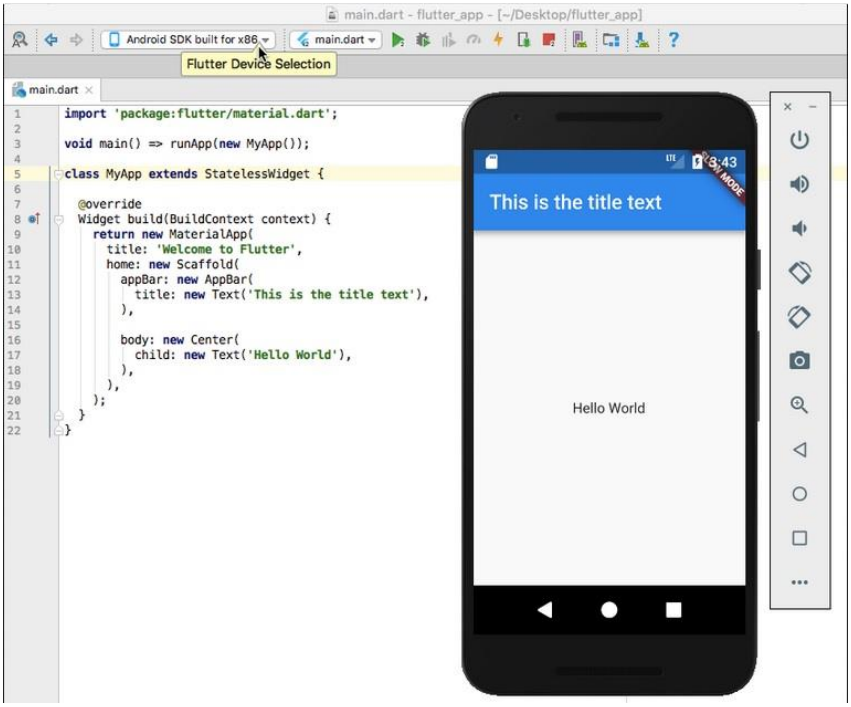

Рисунок 2.9 – Інтерфейс мобільного додатку створеного за допомогою Flutter

2.5 Вибір мови програмування

На сучасному етапі розвитку комп’ютерних технологій існує велика кількість мов програмування для розробки мобільних додатків. Для розробки додатків під платформу Android найчастіше використовують мову Java, Kotlin, C# та C++. Кожна з цих мов програмування має свої особливості і створена для вирішення задач певного типу [22].

C і C ++ неможна виконувати на JVM [36], але можна виконати безпосередньо на самому пристрої, що дає більший контроль над такими речами, як, наприклад, пам'ять. Для вимогливих додатків це допоможе отримати з пристрою максимум продуктивності. До речі, також можна використовувати бібліотеки, написані на C або C ++.

C # - це більш простий, більш зручний об'єктно-орієнтований аналог мов C і C ++, розроблений компанією Microsoft [37]. Він поєднує в собі міць C ++, зручність Visual Basic, а також багато особливостей синтаксису Java. Як і в Java, в C # реалізована функція збірки сміття (garbage collection), яка періодично

звільняє пам'ять, видаляючи об'єкти, які не затребувані додатками, тому не доводиться турбуватися про такі речі, як витік і звільнення пам'яті.

Java – об'єктно-орієнтована мова програмування, випущена 1995 року компанією «Sun Microsystems» як основний компонент платформи Java. З 2009 року мовою займається компанія «Oracle», яка того року придбала «Sun Microsystems». В офіційній реалізації Java-програми компілюються у байткод, який при виконанні інтерпретується віртуальною машиною для конкретної платформи.

Мова значно запозичила синтаксис із C і C++. Зокрема, взято за основу об'єктну модель C++, проте її модифіковано. Усунуто можливість появи де- 23 яких конфліктних ситуацій, що могли виникнути через помилки програміста та полегшено сам процес розробки об'єктно-орієнтованих програм. Ряд дій, які в C/C++ повинні здійснювати програмісти, доручено віртуальній машині. Передусім Java розроблялась як платформо-незалежна мова, тому вона має менше низькорівневих можливостей для роботи з апаратним забезпеченням, що в порівнянні, наприклад, з C++ зменшує швидкість роботи програм. За необхідності таких дій Java дозволяє викликати підпрограми, написані іншими мовами програмування.

Kotlin — статично типізована мова програмування, що працює поверх JVM і розробляється компанією JetBrains [38]. Також компілюється в JavaScript. Мову названо на честь острова Котлін у Фінській затоці, на якому розміщена частина Кронштадту. Автори ставили перед собою ціль створити лаконічнішу та типобезпечнішу мову, ніж Java, і простішу, ніж Scala. Наслідками спрощення, порівняно з Scala стали також швидша компіляція та краща підтримка IDE.

Dart - використовує об’єктно-орієнтований підхід і C-подібний синтаксис для більшої простоти і доступності., призначений насамперед для розробки мобільних додатків [17].

Особливості мови Dart [40]:

– забезпечення швидкого запуску та високої продуктивності для всіх сучасних веб-браузерів та різних типів середовищ, від портативних пристроїв до потужних серверів;

– Можливість визначення класів та інтерфейсів, що дозволяють використовувати інкапсуляцію та повторно використовувати існуючі методи та дані;

– Додаткові інструкції щодо типу, визначення типів полегшує налагодження та ідентифікацію помилок, робить код більш зрозумілим для сприйняття, спрощує його вдосконалення та аналіз сторонніми розробниками. [10]

– Серед підтримуваних типів: різні типи хешей, масивів і списків, черг, числових і рядкових типів, типи визначення дати та часу, регулярні вирази (RegExp). Можна створити власні типи; [10] для організації паралельного виконання пропонується використовувати класи з атрибутом isolate, код яких працює повністю в ізольованому просторі в окремій області пам’яті, взаємодіючі з основним процесом за допомогою надсилання повідомлень;

– Набір готових інструментів для підтримки розвитку мовою Dart, включаючи реалізацію динамічних інструментів розробки та налагодження з виправленням коду на ходу ("редагування та продовження"). [10]

– Можливість створення однорідних систем, що охоплюють як клієнтські, так і серверні частини.

2.6 Аналіз засобів реалізації роботи 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 [40].

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 2.10);
– Підтримувати особливості інтеграції з додатками під операційні системи Android і iOS;
– Управління налаштуваннями , створення колекцій.
– Зберігання даних в хмарі від Google (рис 2.11), 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).

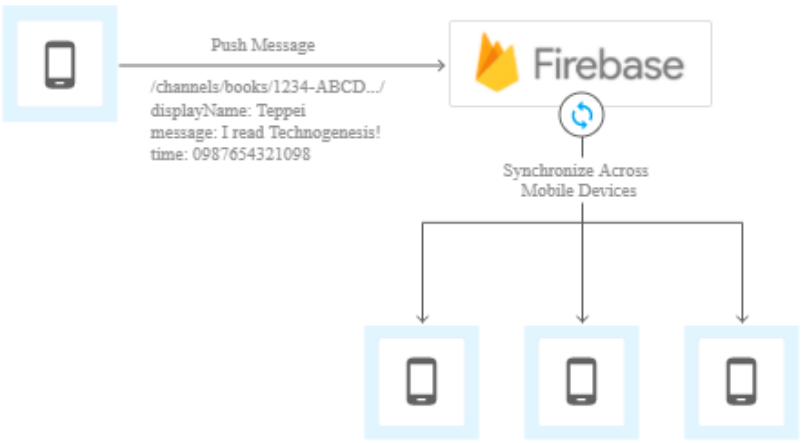

Рисунок 2.10 - Зв'язок між додатком і хмарним сховищем

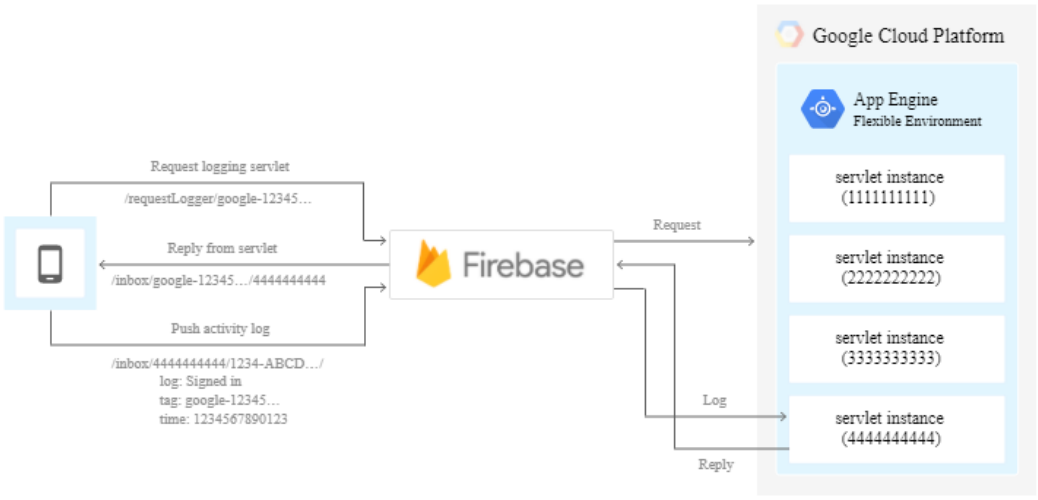

Рисунок 2.11 - Зв'язок між додатком і хмарним сховищем

2.7 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 однієїз 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.

2.8 Удосконалення підходу генерації тестів

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. Множина понять в понятійно-тезисній моделі позначається наступним чином.(2.1)

$$C = \{c1, \dots, cn1\} \quad (2.1)$$

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. Множина тез у програмі позначається так (2.2):

$$T = \{t1, \dots, tn2\} \tag{2.2}$$

В сукупності наведені вище елементи складають з себе ПТ-елементи. Будь-яка теза відповідає лише одному поняттю. Його позначено відношенням (2.2):

$$CT: T \rightarrow C \tag{2.3}$$

Поняття може відповідати довільній кількості тез (2.3.4):

$$TC: C \rightarrow 2T \tag{2.4}$$

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. Вони позначаються таким чином (2.5):

$$V = \{v1, \dots, vn3\} \tag{2.5}$$

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, що задається наступним відображенням (2.6).

$$TV: V \rightarrow 2T \quad (2.6)$$

Кожна tj, у свою чергу, стосується одного навчального фрагменту vi (2.7):

$$VT: T \rightarrow V \quad (2.7)$$

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. Шаблони, які використовуються для тестових запитань задаються наступним відображенням (2.8):

$$TaskTempl: Task \rightarrow TTempl \quad (2.8)$$

Спосіб побудови тестових завдань на основі понятійно-тезисних елементів описується шаблонами. Функція додавання нових шаблонів надає можливість удосконалення системи не лише на етапу проектування, але й після її розгортання і введення в експлуатацію для налаштування алгоритму створення завдань для різних дисциплін.

2.9 Висновок

В даному розділі магістерської кваліфікаційної роботи було виконано аналіз особливостей класифікації мобільних додатків. Описано переваги та недоліки популярних програмних засобів у своїх категоріях.

Описано та проаналізовано існуючі сучасні мови програмування та середовищ розробки в яких можна розробляти додатки.

Було удосконалено спосіб побудови тестових завдань, і доведено доцільність модифікації.

Також було ознайомлено з основними принципами та специфікаціями на яких будуються вищевказані технології. Кожна з представлених технологій використовує найсучасніші методи розробки програмного забезпечення.

Сучасне товариство Android розробників використовує мову Kotlin в якості основного. Він є оптимізованою і адаптованої версією Java для розробки мобільних додатків. В процесі аналізу та ознайомлення, була обрана менш популярна об'єктно-орієнтована багатоплатформена мова Dart і платформа для розробки Flutter, що розробляються Google.

Для платформи Flutter розроблені бібліотеки для взаємодії з СУБД як були визначені функціональні і нефункціональні вимоги до додатка, SQLite. Ці бібліотеки реалізують ORM та raw-запити, які дозволяють інтерпретувати запити мови Dart, в запити СУБД. Реалізація бази даних за допомогою подібних бібліотек практичні, але припускають локальне сховище, створення власних функцій для взаємодії бази даних.

Все це зумовило обрати Firebase, як основи 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.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ОЦІНЮВАННЯ ЯКОСТІ ЗНАНЬ

3.1 Реалізація функціональних можливостей інформаційної технології оцінювання якості знань

Перед тим як почати розробку інформаційної технології оцінювання якості знань, потрібно визначити які функціональні можливості вона має реалізовувати, які вхідні і вихідні дані мають бути, тощо [26]. Для наглядної демонстрації функціональних можливостей інформаційної технології була створена діаграма прецедентів інформаційної технології. (Рисунок 3.1)

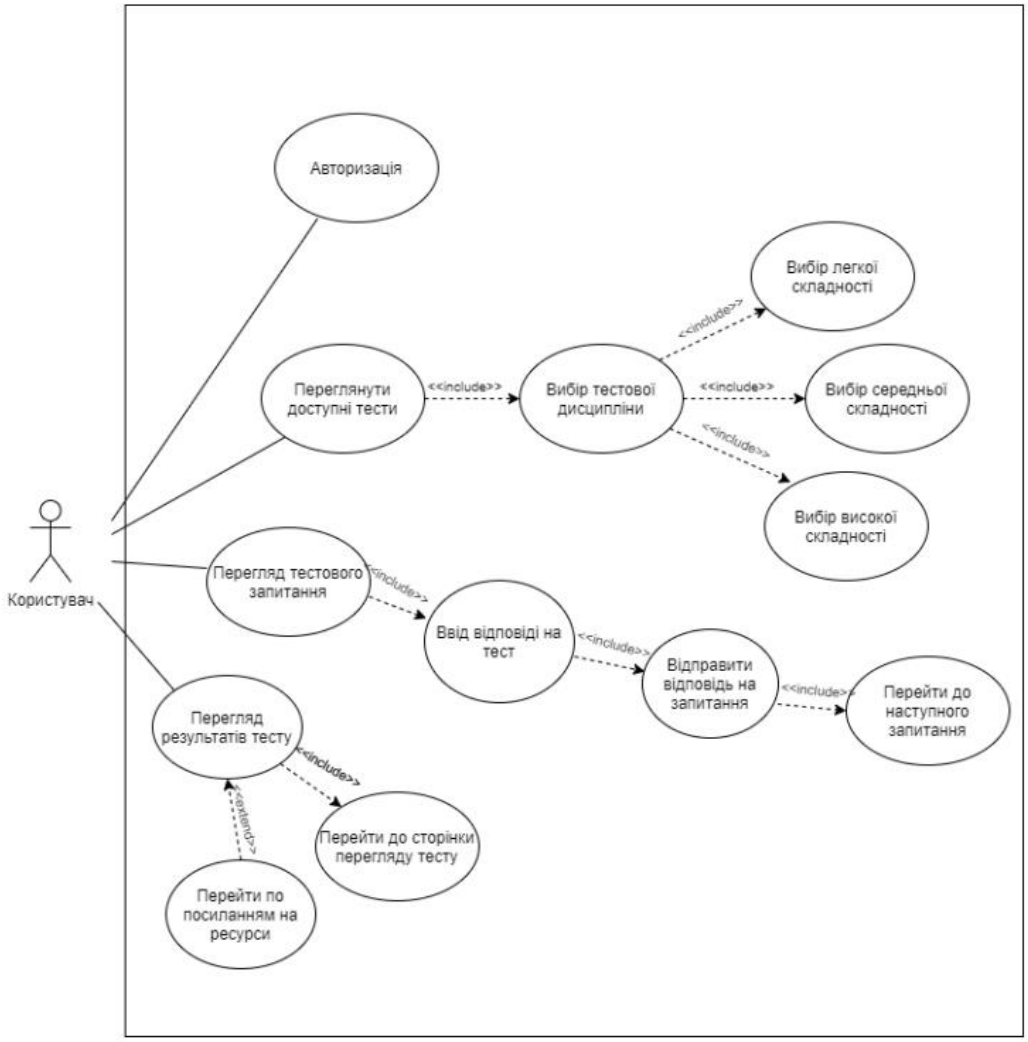

Рисунок 3.1 — Діаграма прецедентів інформаційної технології оцінювання якості знань

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.

3.2 Розробка алгоритму оцінювання якості знань

Загальний алгоритм роботи інформаційної системи технології оцінювання якості знань наведено на рис. 3.2.

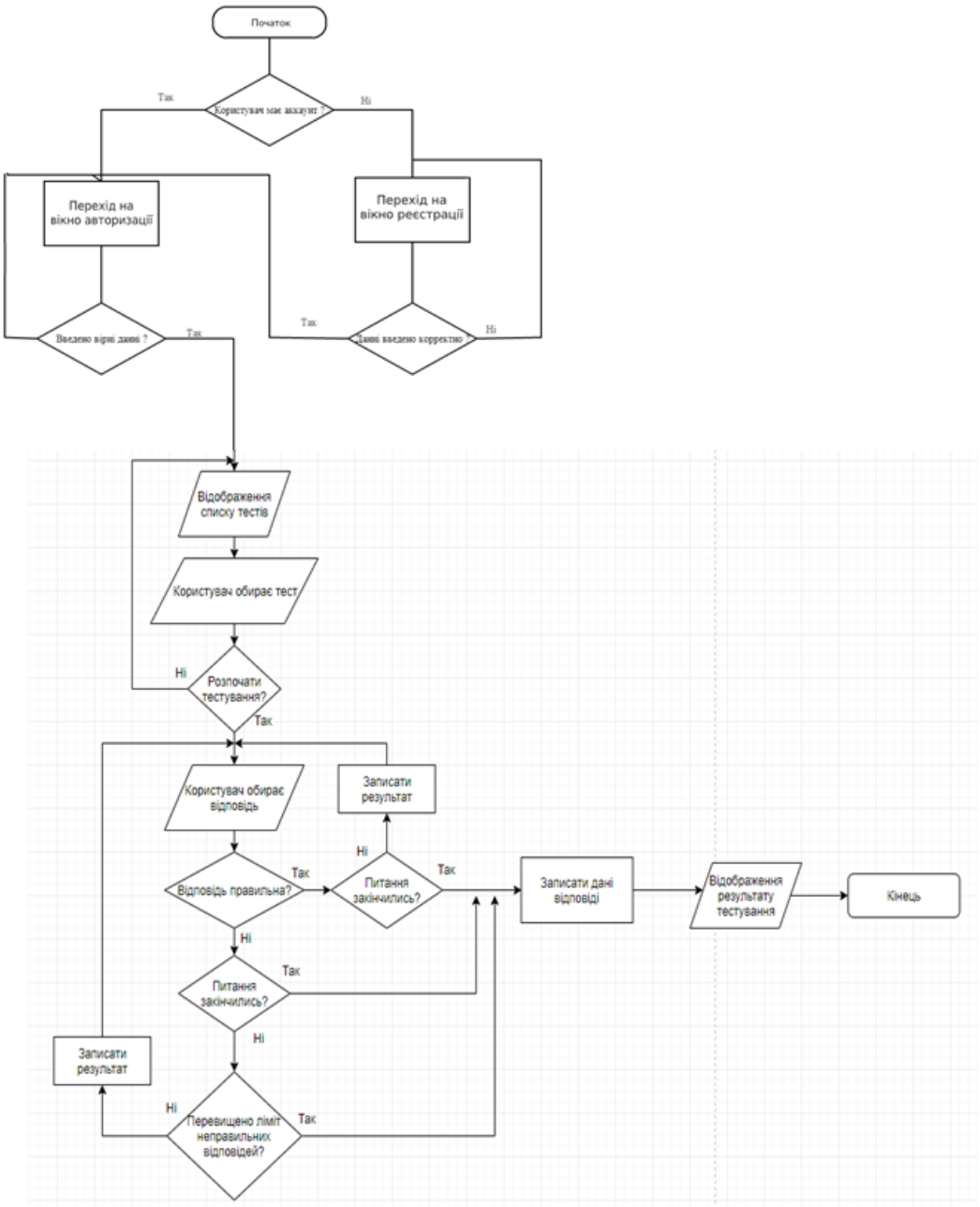

Рисунок 3.2 – Загальний алгоритм роботи мобільного додатку оцінювання якості знань

3.3 Опис загальної структури проекту оцінювання якості знань

Проект складається з окремих структурних елементів і знаходиться в директорії lib (Рисунок 3.3)

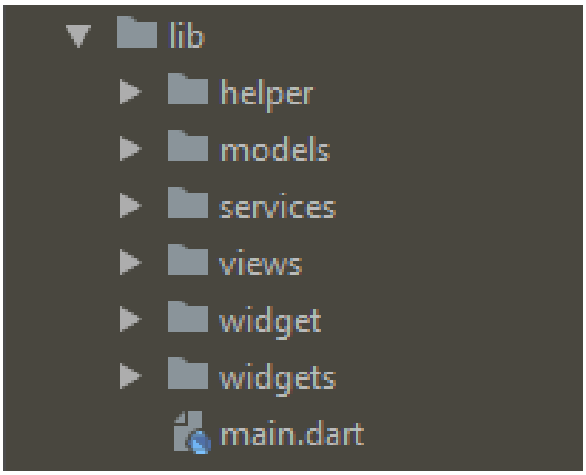

Рисунок 3.3 — Структура файлів проекту

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.

3.4 Реалізація клієнт-серверної частини проекту оцінювання якості знань

Під час розробки мобільного застосунку була спроектована і реалізована така схема системи мобільного застосунку, рисунок 3.4.

Клієнт-серверний застосунок, при використані котрого користувач отримує дані з одного із сервісів Firebase, назва якого Cloud Firestore [29 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.

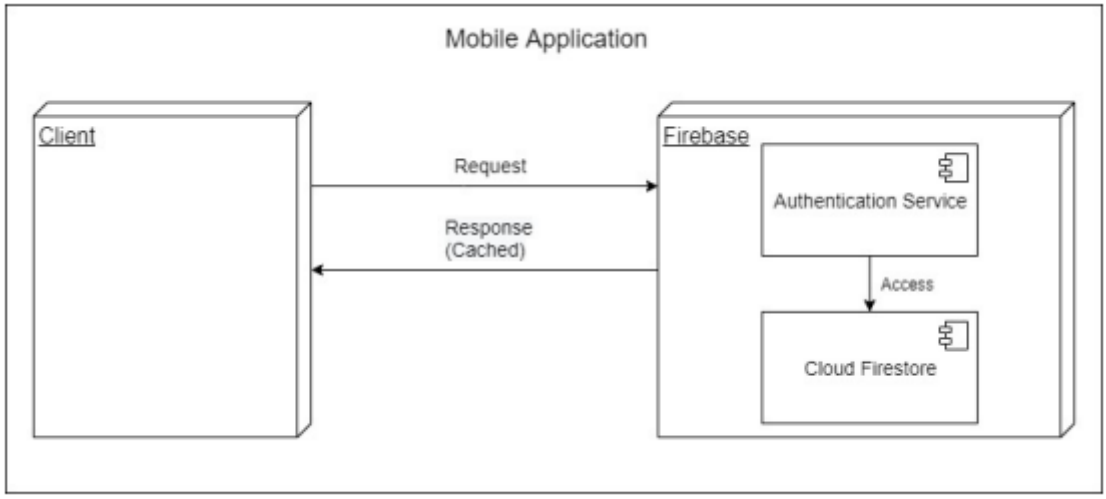

Рисунок 3.4 – Принцип роботи клієнт-серверної частини додатка

3.5 Розробка UML-діаграми класів мобільного додатку оцінювання якості знань

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 [27].

Під час проектування мобільного додатку було розроблено діаграму класів (рисунок. 3.5).

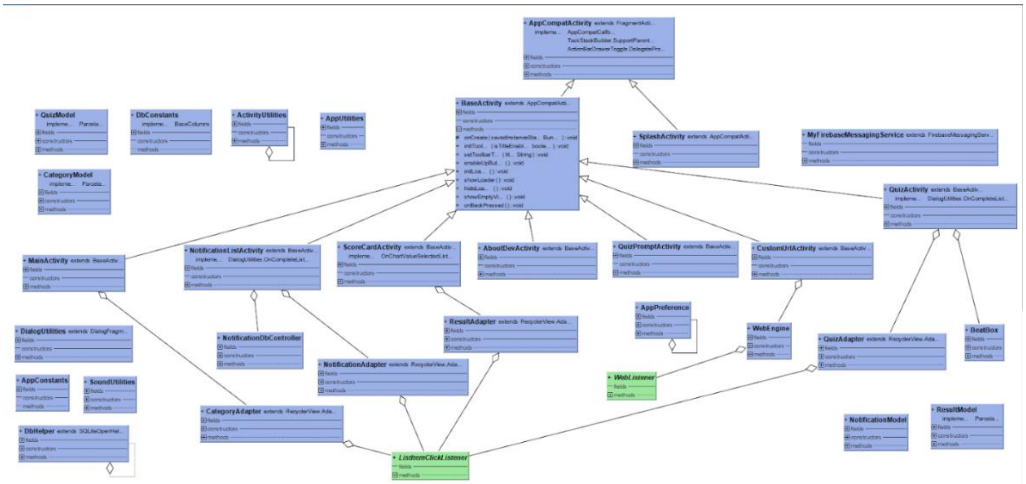

Рисунок 3.5 – Діаграма класів додатку оцінювання якості знань

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

3.6 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. 3.5) 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 3.5. - Реалізація функції створення запису в базі даних

Асинхрона функція getData (Рис. 3.6) 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 3.6 - Реалізація функції читання даних

Функція поновлення даних (рис. 3.7) 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 3.7 - Реалізація функції створення запису в базі даних

Функція видалення даних (Рис. 3.8) 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 3.8 - Реалізація функції видалення даних

3.7 Розробка інтерфейсу мобільного додатку оцінювання якості знань

Під час розроблення мобільного додатку було дотримано всіх вище наведених правил та спроектовано оптимальний на вигляд і для застосування інтуїтивно зрозумілий інтерфейс. На рисунку 3.9 наведено головне меню вибору тестів. На рисунку 3.10 зображено сторінку проходження тесту, де користувач може обирати певні відповіді.

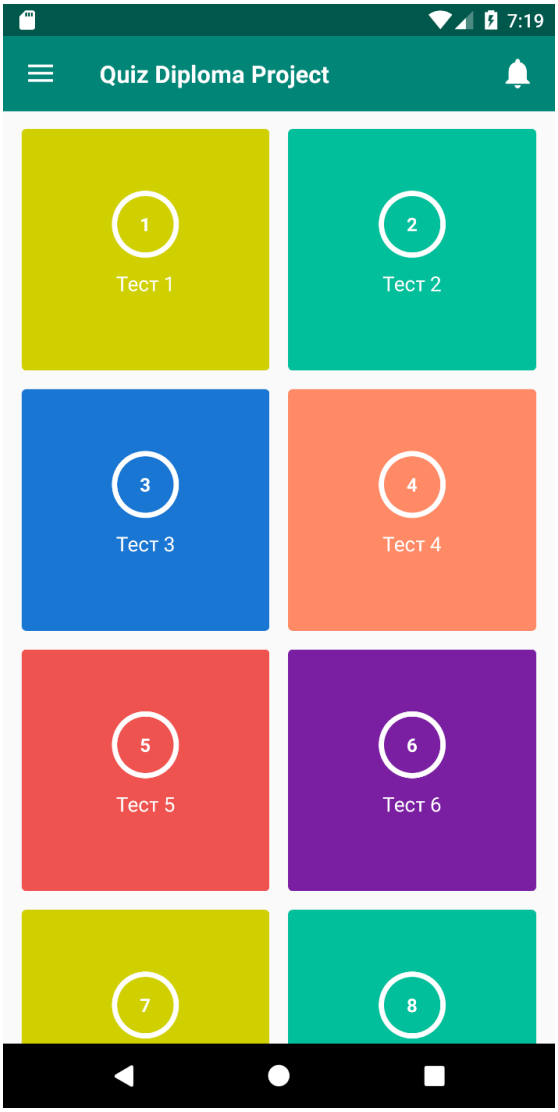

Рисунок 3.9 – Головне меню вибору тестів

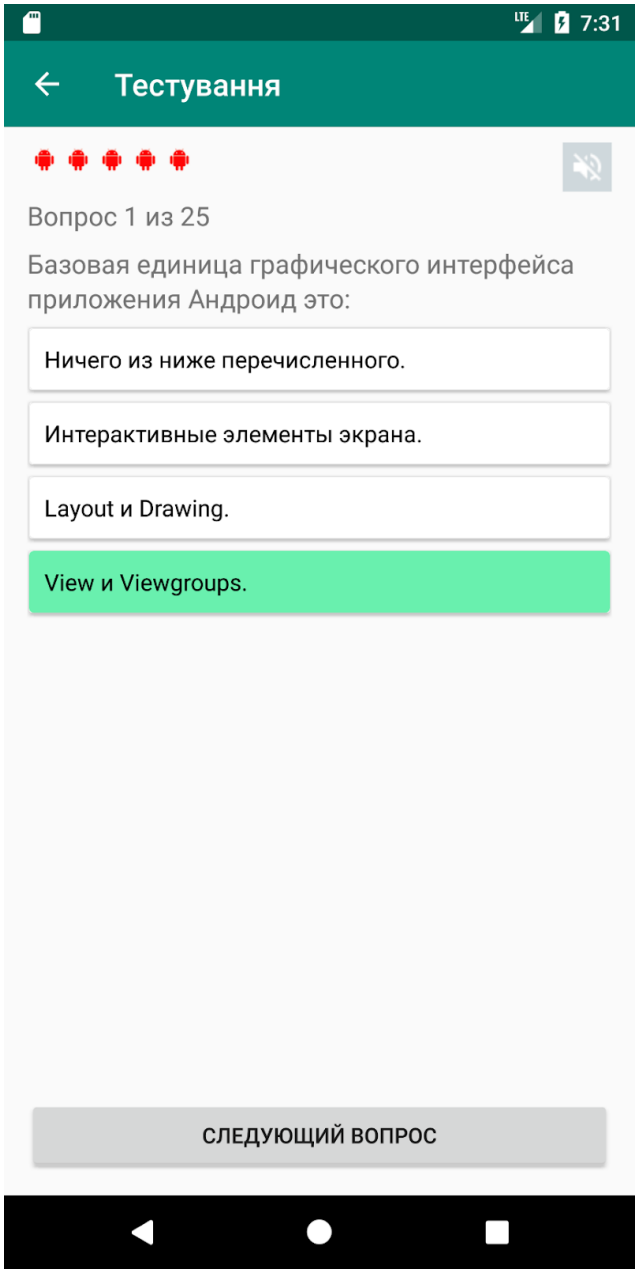

Рисунок 3.10 – Сторінка проходження тесту

3.8 Тестування мобільного додатку оцінювання якості знань та аналіз результатів

Невід'ємною частиною будь-якої реалізації програмного продукту є процес тестування або так званої перевірки роботи програми згідно вимог, які були зазначенні при початку реалізації продукту. Під час перевірки програмного продукту зазвичай виявляються дефекти системи, при яких система може не працювати, блокуватися або працювати не вірно. В разі виявлення дефекту або

помилки, розробник має виправити її. Тестування за своєю специфікою поділяється на два види: ручне тестування та автоматизоване. Під час ручного тестування всі операції виконуються вручну, тобто самі перевіряємо і оцінюємо, як працює та чи інша функція [7]. Автоматизоване тестування дає змогу перевірити код реалізації функцій, за допомогою якого маємо можливість протестувати фунціонал, і перевірити, чи повертається потрібний результат. Кожен з цих видів містить в собі вже свої види тестування.

3.8.1 UI-тестування, тестування продуктивності

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. 3.11). В якості віртуального апарату для тестування використовується Google Pixel 2.

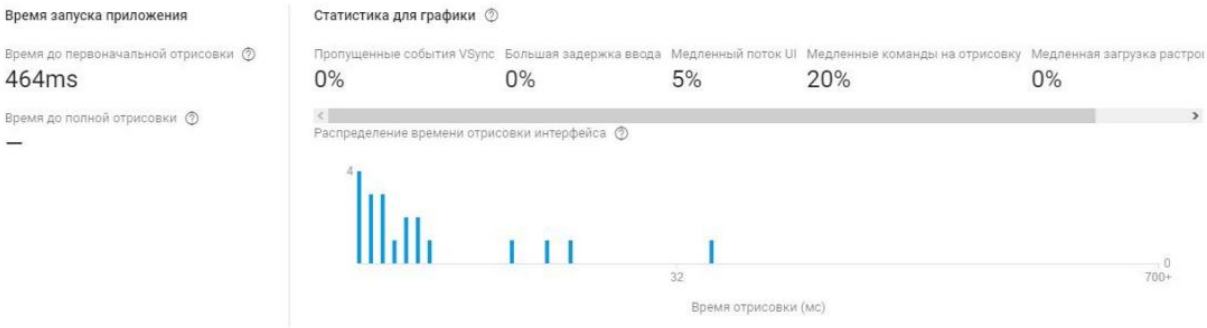

Рисунок 3.11 – Статистика тестування відтворення екрану

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. 3.12). 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.

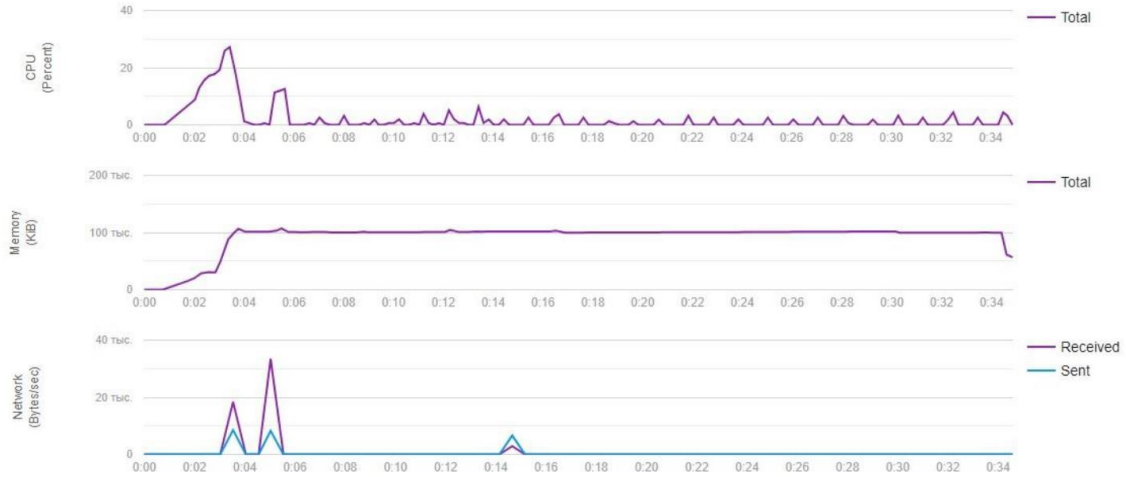

Рисунок 3.12 – Метрика продуктивності додатка

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.

3.8.2 Функціональне тестування

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 мобільного додатка.

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. 3.13).

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. 3.14).

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.

3.8.3. Інтеграційне тестування

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.

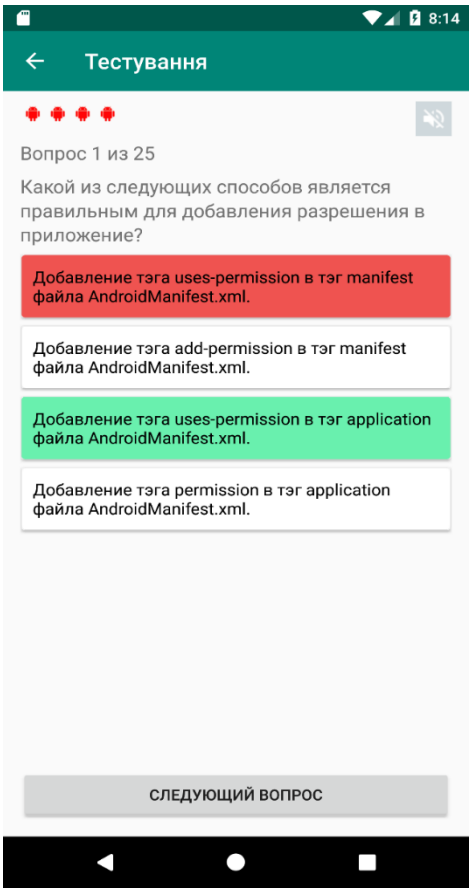

Рисунок 3.13 – Результат проходження тесту роботи відображення обраної та коректної відповіді в тесті.

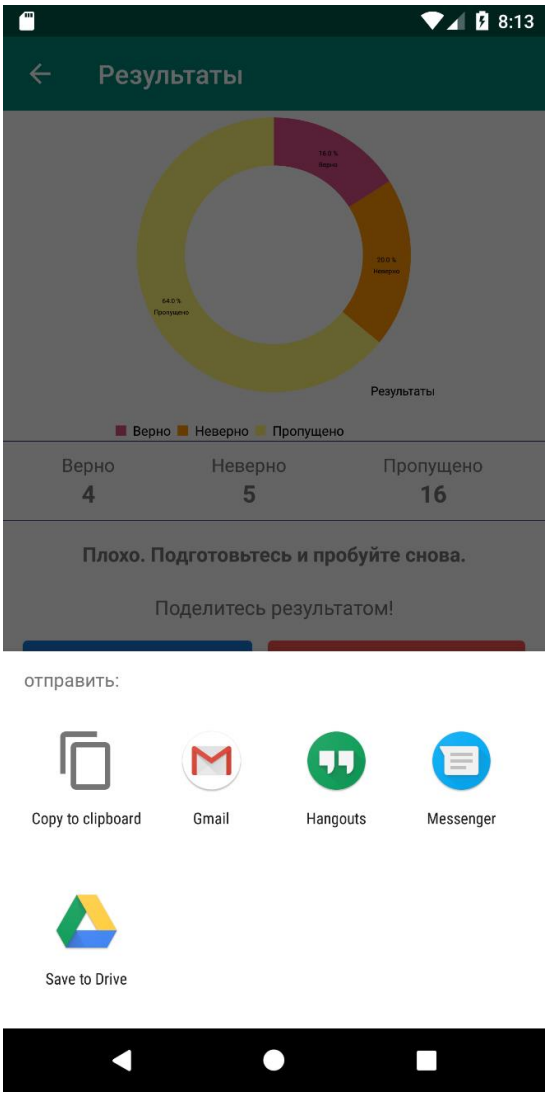

Рисунок 3.14 – Результат проходження тесту коректної роботи вікна результату тесту та можливості відправлення цього результату

3.9 Висновок

У розділі було визначено які функціональні можливості інформаційної технології оцінювання якості знань має реалізовувати.

В ході дослідження було розроблено алгоритм роботи програмного засобу оцінювання якості знань, побудовано його графічне представлення.

Розроблено UML-діаграму класів програми для оцінювання якості знань.

Під час розроблення мобільного додатку було спроектовано оптимальний на вигляд і для застосування інтуїтивно зрозумілий інтерфейс.

У процесі розробки програми проводилося поетапне тестування з метою виявлення програмних помилок і невідповідностей ТЗ. Тестований програмний продукт послідовно запускався, його поведінка аналізувалась, і при необхідності по результатам аналізу вносилися зміни в програмний код.

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

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Оцінювання комерційного потенціалу розробки (технологічний аудит розробки)

Метою проведення технологічного аудиту є оцінювання комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності [41]. В результаті оцінювання робиться висновок щодо напрямів (особливостей) організації подальшого її впровадження з врахуванням встановленого рейтингу.

Для проведення технологічного аудиту залучено 3-х незалежних експертів (Е1, Е2). Оцінювання комерційного потенціалу розробки здійснено за 12-ю критеріями, наведеними в таблиці 4.1.

Таблиця 4.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Бали (за 5-ти бальною шкалою)					
Критерій	0	1	2	3	4
Технічна здійсненність процесії:					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижча за ціни аналогів	Ціна продукту значно нижча за ціни аналогів

Продовження таблиці 4.1

Бали (за 5-ти бальною шкалою)					
Критерій	0	1	2	3	4
Ринкові переваги (недоліки):					
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування

Продовження таблиці 4.1

	Бали (за 5-ти бальною шкалою)				
Критерій	0	1	2	3	4
Практична здійсненність					
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловом у комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використо-вуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідна отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання комерційного потенціалу розробки зведено у таблицю 4.2.

Таблиця 4.2 — Результати оцінювання комерційного потенціалу розробки

Критерії	Експерти	
	Озеранський В.С.	Сілагін О.В.
	Бали, виставлені експертами:	
1	2	2
2	2	2
3	3	3
4	3	4
5	2	2
6	3	3
7	2	2
8	4	3
9	4	3
10	4	4
11	4	4
12	4	3
Сума балів	37	35
Середньоарифметична сума балів СБ	36	

За даними таблиці 4.2 можна зробити висновок, що рівень комерційного потенціалу розробки вище середнього, так як середньоарифметична сума балів всіх експертів становить 36, що знаходиться в межах від 31 до 40.

4.2 Прогнозування витрат на виконання науково-дослідної (дослідно-конструкторської) роботи

Прогнозування витрат на виконання науково-дослідної чи дослідно-конструкторської роботи складається з таких етапів:

1. Розрахунок витрат, які безпосередньо стосуються виконавців даного розділу НДДКР.
2. Розрахунок загальних витрат на виконання даної НДДКР.
3. Прогнозування загальних витрат на виконання та впровадження результатів НДДКР.

4.2.1 Розрахунок витрат, які безпосередньо стосуються виконавців розділу

Розрахунок витрат здійснено за такими статтями та формулами:

1. Основна заробітна плата кожного із розробників (формула 4.1):

$$З_о = \frac{М}{T_p} \cdot t \text{ грн.}, \quad (4.1)$$

М – місячний посадовий оклад конкретного розробника, грн. Оклад інженера програміста в нашому випадку складатиме 9000 грн., місячний оклад керівника проекту складає 10000 грн.

$T_р$ – кількість робочих днів у місяці. Середнє значення цього показника за місяці розробки складає 22 дні. t – число робочих днів роботи розробника – 41 день, число робочих днів роботи керівника проекту – 10 днів.

Зроблені розрахунки для керівника і програміста занесемо до таблиці 4.3.

Таблиця 4.3 – Розрахунки основної заробітної плати

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
Керівник проекту	10000	454.54	10	4545.4
Інженер-програміст	9000	409.09	41	16772.6
Всього				21318

2. Додаткова заробітна плата $З_д$ всіх розробників розраховується як (10…12%) від суми основної заробітної плати всіх розробників:

$$З_д = (0{,}1 \ldots 0{,}12) \cdot З_о = 0{,}1 * 21318 = 2131{,}8 \text{ (грн.)}$$

3. Нарахування на заробітну плату $Н_{зп}$ для працівників бюджетної сфери становить 22 % від суми основної та додаткової заробітної плати:

$$Н_{зп} = (З_о + З_д) \cdot \frac{\beta}{100} = (З_о + З_д) \cdot \frac{22}{100} = (21318 + 2131{,}8) * 0{,}22 = 5158{,}9 \text{ (грн.)}$$

4. Амортизація обладнання, комп'ютерів та приміщень, які використовувались при розробці програмного продукту розраховується за формулою 4.2:

$$А = \frac{Ц * Н_а}{100} * \frac{Т}{12} \text{ грн.}, \qquad (4.2)$$

де Ц – балансова вартість персонального комп'ютера, Ц = 18000 грн.;

$Н_а$ – річна норма амортизаційних відрахувань, $Н_а$=10%;

Т – термін використання комп'ютера, Т = 1 місяць.

$$А = \frac{18000 * 10}{100} * \frac{1}{12} = 150 \text{ грн.}$$

5. Розрахуємо витрати на матеріали. Витрати на матеріали розрахуємо за формулою 4.3:

$$K = \sum_1^n Н_і \cdot Ц_і \cdot К_і, \qquad (4.3)$$

де *n* – кількість комплектуючих;

Нi - кількість комплектуючих і-го виду;

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

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

Витрати на матеріали, що були використані на розробку програмного продукту наведені в таблиці 4.4. Загальна сума витрат $В_м$ = 95 грн.

Таблиця 4.4 - Розрахунок витрат на матеріали

Найменування матеріалу	Ціна, грн.	Витрачено, шт.	Вартість витрачених матеріалів, грн.
1. Папір друкарський	80,00	1	80,00
2. Компакт-диск	7,50	2	15,00
Всього		M = 95,00	

6. Витрати на силову електроенергію $В_с$ розраховуються за формулою 4.4:

$$В_с = В*П*Ф*К_п \text{ грн,.} \quad (4.4)$$

де В – вартість 1 кВт-год. електроенергії, В = 1,44 грн/кВт;

П – потужність комп'ютера, П = 0,5 кВт;

Ф – фактична кількість годин роботи комп'ютера та інших пристроїв при створенні програмного продукту, Ф = 41 * 5 = 205 (год.);

Кн – коефіцієнт використання потужності, Кн = 0,7.

Отже, витрати на силову електроенергію будуть такі:

$$B_e = 1,44*0,5*205*0,7 = 73.1 \text{ грн.}$$

7. Інші витрати $В_{ін}$ приймають як 100-300% від суми основної заробітної плати розробників (формула 4.5):

$$В_{iн} = (1 \dots 3) * (З_о + З_р) \quad (4.5)$$

$$В_{ін} = 2 * (21318 + 2131,8) = 46899,6 \text{ (грн.).}$$

8. Загальні витрати на розробку програмного продукту дорівнюють сумі всіх витрат (формула 4.6):

$$В = З_о + З_д + Н_{зп} + А + М + В_е + В_{ін} \text{ [грн.], (4.6)}$$

$$В = 21318 + 2131,8 + 5158,9 + 150 + 95 + 73,1 + 14363,32 = 43290,12 \text{ (грн.).}$$

4.2.2 Розрахунок загальних витрат на виконання даної роботи

Оскільки здійснено повністю всю роботу і в подальшому вона не буде продовжена, то $В_{заг} = В = 43290,12$ грн.

4.2.3 Прогнозування загальних витрат на виконання та впровадження результатів роботи

Прогнозування загальних витрат на виконання та впровадження результатів магістерської кваліфікаційної роботи здійснено за формулою 4.7:

$$ЗВ = \frac{В_{заг}}{\beta}, \quad (4.7)$$

де β – коефіцієнт, який характеризує етап виконання магістерської кваліфікаційної роботи, 0,9.

$$ЗВ = \frac{43290,12}{0,9} = 48100,1 \text{ (грн.)}$$

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки

Спрогнозуємо отримання прибутку від реалізації результатів нашої розробки. Зростання чистого прибутку можна оцінити у теперішній вартості грошей. Це забезпечить підприємству (організації) надходження додаткових коштів, які дозволять покращити фінансові результати діяльності .

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

$$\Delta П_i = \sum_1^n (\Delta П_я \cdot N + П_я \Delta N)_i \quad (4.8)$$

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

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

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

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

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

В результаті впровадження результатів наукової розробки витрати на виготовлення інформаційної технології зменшяться на 40 грн (що автоматично спричинить збільшення чистого прибутку підприємства на 40 грн), а кількість користувачів, які будуть користуватись збільшиться: протягом першого року – на 200 користувачів, протягом другого року – на 130 користувачів, протягом третього року – 90 користувачів. Реалізація інформаційної технології до впровадження результатів наукової розробки складала 500 користувачів, а прибуток, що отримував розробник до впровадження результатів наукової розробки – 400 грн.

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

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

$$\Delta П_1 = 40 \cdot 500 + (400 + 40) \cdot 200 = 90000 \text{ грн.}$$

Протягом другого року:

$$\Delta П_2 = 40 \cdot 500 + (400 + 40) \cdot (200 + 130) = 147200 \text{ грн.}$$

Протягом третього року:

$$\Delta П_3 = 40 \cdot 500 + (400 + 40) \cdot (200 + 130 + 90) = 186800 \text{ грн.}$$

Отже, протягом трьох років підприємство може розраховувати на збільшення чистого прибутку від реалізації наукової розробки.

4.4 Розрахунок ефективності вкладених інвестицій та період їх окупності

Загальні витрати на виконання та впровадження результатів роботи дорівнює 48100,1 грн. Результати вкладених у наукову розробку інвестицій почнуть виявлятися через два роки.

Ці результати виявляться у тому, що у першому році підприємство отримає збільшення чистого прибутку на 76250 грн., у другому році – збільшення чистого прибутку на 129375 грн., у третьому році – збільшення чистого прибутку на 171875 грн.

Рисунок, що характеризує рух платежів (інвестицій та додаткових прибутків) буде мати вигляд, рисунок 4.1.

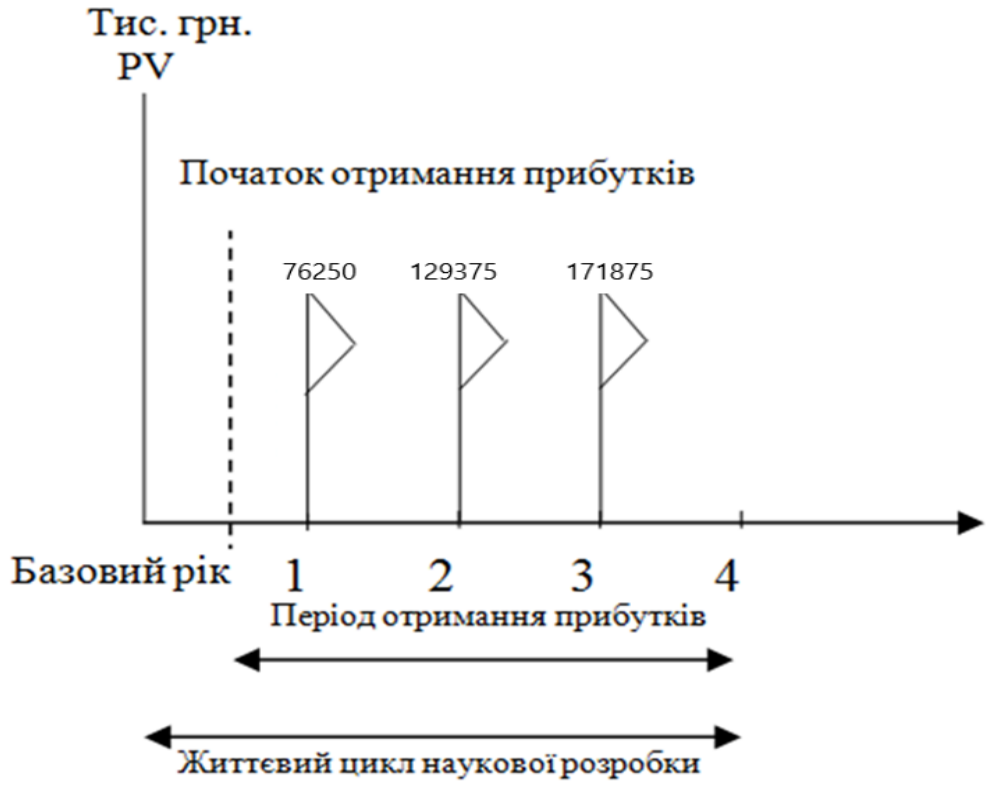

Рисунок 4.1 – Вісь часу з фіксацією платежів, що мають місце під час розробки та впровадження результатів НДДКР

4.4.1 Розрахунок абсолютної ефективності вкладених інвестицій

Для розрахунку абсолютної ефективності вкладених інвестицій користуються формулою 4.9:

$$Е_{абс} = (ПП – PV), \quad (4.9)$$

де ПП – приведена вартість всіх чистих прибутків, які отримає підприємство (організація) від реалізації результатів наукової розробки, грн.; PV – теперішня вартість інвестицій PV = ЗВ = 48100,1 грн.

У свою чергу, приведена вартість всіх чистих прибутків ПП розраховується за формулою 4.10:

$$ПП = \sum_1^T \frac{\Delta П_i}{(1+\tau)^t} \quad (4.10)$$

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

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

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

t – період часу (в роках) від моменту отримання чистого прибутку до точки «0».

Якщо $Е_{абс} < 0$, то результат від проведення наукових досліджень та їх впровадження буде збитковим, і вкладати кошти в проведення цих досліджень ніхто не буде.

Якщо $Е_{абс} > 0$, то результат від проведення наукових досліджень та їх впровадження принесе прибуток, але це також ще не свідчить про те, що інвестор буде зацікавлений у фінансуванні даного проекту (роботи).

Користуючись формулою 4.10, розрахуємо приведену вартість усіх чистих прибутків:

$$ПП = \frac{48100,1}{(1+0,1)^0} + \frac{43300}{(1+0,1)^2} + \frac{30800}{(1+0,1)^3} + \frac{18500}{(1+0,1)^4} = 360660,7$$

Тепер, використовуючи формулу 4.9, розрахуємо абсолютну ефективність вкладених інвестицій:

$$Е_{абс} = 360660,7 - 28157,4 = 332503,3 грн$$

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

4.4.2 Розрахунок відносної ефективності вкладених у наукову розробку інвестицій

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $Е_в$. Для цього використаємо формулу 4.11:

$$Е_В = \sqrt[Т_ж]{1 + \frac{Е_{абс}}{PV}} - 1\ , \qquad (4.11)$$

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

PV –теперішня вартість інвестицій PV = ЗВ, грн; $Т_ж$ – життєвий цикл наукової розробки, роки.

$$Е_в = \sqrt[3]{1 + \frac{332503,3}{48100,1}} - 1 = 0,99 \text{ або } 99\ \%$$

Розраховану величину $Е_в$ порівняємо з мінімальною (бар'єрною) ставкою дисконтування $\tau_{мін}$, яка визначає ту мінімальну дохідність, нижче за яку інвестиції вкладатися не будуть. У загальному вигляді мінімальна (бар'єрна) ставка дисконтування $\tau_{мін}$ визначається за формулою 4.12:

$$\tau = d + f\ , \qquad (4.12)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; d = (0,14...0,2);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина f = (0,05...0,1).

Якщо величина $E_в > \tau_{мін}$, то інвестор може бути зацікавлений у фінансуванні даної наукової розробки. В іншому випадку фінансування наукової розробки здійснюватися не буде.

Тому розрахуємо мінімальну ставку дисконтування:

$$\tau = 0{,}17 + 0{,}1 = 0{,}27$$

Оскільки $E_в$ = 0,99 = 99 % $>$ $\tau_{мін}$ = 0,27 = 27 %, то у інвестора буде зацікавленість вкладати гроші в наукову розробку, оскільки значно більші прибутки він отримає від неї, ніж якби просто поклав свої гроші на депозит у комерційний банк.

4.4.3 Розрахунок терміну окупності інвестицій

Термін окупності вкладених у реалізацію наукового проекту інвестицій $T_{ок}$ розраховується за формулою 4.13:

$$T_{ок} = \frac{1}{E_в}. \qquad (4.13)$$

Якщо $T_{ок}$ < 3…5-ти років, то фінансування наукової розробки є доцільним.

$$T_{ок} = \frac{1}{0{,}99} = 1{,}01 \text{ року}$$

Отже, фінансування наукової розробки є доцільним, оскільки $T_{ок} < 5$ років.

4.5 Висновок

Результати проведених розрахунків дають можливість зробити висновок про доцільність розробки та впровадження нашої наукової роботи.

Це підтверджують такі показники як:

– абсолютна ефективність вкладених інвестицій, яка дорівнює 332503,3 грн., що є більшим 0 і вказує на те, що інвестор може бути зацікавленим у нашій розробці;

– відносна ефективність наукової розробки становить 99%, що є вищим за мінімальну ставку дисконтування (27%), тому вкласти кошти у нашу розробку є вигідніше, ніж покласти кошти на депозит;

– термін окупності вкладених у реалізацію наукового проекту інвестицій складе 1,01 року, що є менше 5-ти і вказує швидку окупність вкладених інвестицій.

Крім того, розраховано, що наукова розробка приноситиме підприємству додатковий прибуток протягом 3-х років за рахунок покращення її якості порівняно з існуючими аналогами.

ВИСНОВКИ

В ході виконання магістерської кваліфікаційної роботи розроблено інформаційна технологію оцінювання якості знань.

Проведено аналіз предметної області та досліджені основні комунікаційні тенденції в науковій та освітній сферах. У ході аналізу було зазначено, що інформаційні технології в вигляді мобільного додатку має високий педагогічний потенціал та дає змогу суттєво різноманітити методи організації та реалізації ефективного навчального процесу

Виконано порівняння існуючих додатків оцінювання знань. В ході аналізу об’єкту проектування було визначено вимоги до інформаційної технології.

Описано та проаналізовано існуючі сучасні мови програмування та середовищ розробки в яких можна розробляти додатки. В результаті порівняльного аналізу об'єктно-орієнтованих мов на основі об'єктивних переваг було обрано мову програмування Dart. Для розробки обрано інтегроване середовище розробки Android Studio для роботи з платформою Android, яке задовольняє усім потребам при розробці заданого програмного забезпечення.

Було удосконалено спосіб побудови тестових завдань, і доведено доцільність модифікації.

В ході дослідження було розроблено алгоритм роботи програмного засобу для оцінювання якості знань, побудовано його графічне представлення.

Розроблено UML-діаграму класів додатку для оцінювання якості знань.

Під час розроблення мобільного додатку було спроектовано оптимальний на вигляд і для застосування інтуїтивно зрозумілий інтерфейс користувача.

На основі проведених досліджень розроблено програмне забезпечення функціонування інформаційна технологію оцінювання якості знань.

Проведене тестування розробленого мобільного додатку підтвердило ефективність і правильність функціонування його роботи. Це дозволяє користувачам отримувати правильні результати оцінювання.

Встановлено що розроблений програмний засіб з удосконаленим підходом генерації тестів забезпечив зменшення на 3-4% затрати часу для створення тестових завдань і ймовірність недобросовісного проходження тесту.

Проведено економічні розрахунки витрат та прибутків від впровадження розробки, терміни окупності та визначено її комерційний потенціал. Встановлено, що в сучасної технічної ситуації розробка являється конкурентоспроможною на ІТ-ринку. Щорічна ефективність вкладених в наукову розробку інвестицій склала 99%, а термін окупності - 1,01 року.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

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.

Британський В.А., Озеранський В.С. Порівняльний аналіз сучасних підходів розробки мобільних додатків // Матеріали XLIX науково-технічної конференції підрозділів ВНТУ. Вінниця,2020.

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

Л. Гриневич та ін., Нова українська школа. Концептуальні засади реформування середньої школи, 36 с., 2016. [Електронний ресурс]. Доступно: https://mon.gov.ua/storage/app/media/zagalna%20serednya/nova-ukrainska-shkola-compressed.pdf

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

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

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

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/

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

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

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/

Голощапов А. Google Android: программирование для мобильных устройств. — СПб. БХВ-Петербург, 2010. — 448 с. — ISBN 978-5-9775-0562-8.

Дэйв Марк и др. iOS 6 SDK. Разработка приложений для iPhone, iPad и iPod touch = Beginning iOS 6 Development Exploring the iOS SDK. — М.:«Вильямс», 2013. — 672 с. — ISBN 978-5-8459-1852-9.

Кроссплатформенная разработка | AppTractor [Електронний ресурс]. – Режим доступу: https://apptractor.ru/develop/cross-platform-development

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

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

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

Елизаренко Г.Н. Проектирование компьютерных курсов обучения: концепция, язык, структура. НТУУ «КПИ». Киев, 2001.

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

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

Документация Android SDK. [Электронный ресурс] URL: https://developer.android.com/studio (дата обращения: 23.01.2019)

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

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

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

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

Арлоу Дж., Нейштадт А. UML 2 и унифицированный процесс. – М.: Символ-Плюс, 2007. – 624 р.

Буч Г., Рамбо Дж., Джекобсон А. Язык UML. Руководство пользователя. – СПб.: Издательство «Питер», 2003. – 432 с

Что делать, если под рукой нет Android-устройства? Обзор Android-эмуляторов [Електронний ресурс] – Режим доступу: https://habr.com/ru/post/218739/

Firebase Guides. [Електронний ресурс] – Режим доступу: https://firebase.google.com/docs/guides

Buildfire [Електронний ресурс] – Режим доступу: https://github.com/BuildFire/sdk/wiki

Apache Cordova [Електронний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/Cordova

PhoneGap [Електронний ресурс] – Режим доступу: http://docs.phonegap.com/

Ionic Framework [Електронний ресурс] – Режим доступу: https://github.com/ionic-team/ionic-framework

Framework7 [Електронний ресурс] – Режим доступу: https://framework7.io/docs/

NativeScript, что за зверь и для чего он нужен? [Електронний ресурс] – Режим доступу: https://habr.com/ru/post/318950/

Jasonette [Електронний ресурс] – Режим доступу: https://docs.jasonette.com/

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

Тестування. [Електронний ресурс]. – Режим доступу: http://arhivstatey.pp.ua/index.php?newsid=26947

Software as a service https://en.wikipedia.org/wiki/Software_as_a_service

What Is Mobile Backend As A Service (MBaaS)? Електронний ресурс]. – Режим доступу: https://backendless.com/what-is-mobile-backend-as-a-service-mbaas/

Грабовецький Б. Є. Економіка підприємства. / Б. Є. Грабовецький, Т.М. Пілявоз – Вінниця: ВНТУ, 2009 – 248 с