Міністерство освіти і науки України
Університет митної справи та фінансів

Факультет інноваційних технологій
Кафедра комп'ютерних наук та інженерії програмного забезпечення

Кваліфікаційна робота бакалавра

на тему: Розробка мобільного додатку «StandWithUkraine»

Виконав: студент групи ІПЗ19-1
Спеціальність 121 Інженерія програмного забезпечення
Музичишин Максим Миколайович
(прізвище та ініціали)
Керівник к.ф. –м.н., доцент кафедри комп'ютерних наук та інженерії програмного
Рудянова Т.М.
(науковий ступінь, вчене звання, прізвище та ініціали)
Рецензент ФОП Гончарук О.В.
(місце роботи)
Front-End Developer
(посада)
Гончарук О.В.
(науковий ступінь, вчене звання, прізвище та ініціали)

Дніпро – 2023

АНОТАЦІЯ

Музичишин М.М. Розробка мобільного додатку «StandWithUkraine».

Кваліфікаційна робота на здобуття ступеня вищої освіти «бакалавр» за спеціальністю 121 «Інженерія програмного забезпечення». – Університет митної справи та фінансів, Дніпро, 2023.

Об'єктом дослідження є інформаційні технології та методи створення додатків для ОС Android.

Метою дослідження є розробка програмного забезпечення, що дозволяє здійснювати збір коштів з рекламодавців для підтримки України.

Методи дослідження: аналіз метрик використання додатку, тестування різних версій та імплементація його за допомогою мови програмування Dart.

Економічна ефективність: очікується підвищення надходження коштів від перегляду реклами для українській армії, постраждалого народу внаслідок бойових дій, матеріальну підтримку держави.

Ключові слова: мобільний додаток, підтримка України, залучення коштів, перегляд реклами, тест з української мови, благодійність, національна ідея.

ABSTRACTS

Muzychyshyn M.M. Development of the mobile application "StandWithUkraine".

Diploma thesis for the degree of higher education "Bachelor" in speciality 121 "Software Engineering." - University of Customs and Finance, Dnipro, 2023.

The object of research is information technology and methods of creating applications for Android OS.

The purpose of the study is to develop software that allows you to collect funds from advertisers to support Ukraine.

Research methods: analysis of application usage metrics, testing of different versions and its implementation using the Dart programming language.

Economic efficiency: it is expected to increase the flow of funds from viewing advertisements for the Ukrainian army, the people affected by the hostilities, and material support of the state.

Keywords: mobile application, support for Ukraine, fundraising, ad viewers, Ukrainian language test, charity, national idea.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОСИЛАНЬ

Аккаунт – це обліковий запис відвідувача тієї чи іншої веб-сторінки, що дозволяє гостю перейти в статус зареєстрованого користувача.

Фреймворк (англ. Framework) – це програмна платформа, яка визначає структуру програмної системи; програмне забезпечення, що полегшує розробку і об'єднання різних компонентів великого програмного проекту.

Dart – об'єктно-орієнтована мова програмування з опціонною типізацією, яка використовується для створення веб-інтерфейсів, мобільних додатків, серверних застосунків та інших програмних продуктів.

Flutter – відкрите SDK для розробки мобільних та веб-додатків з високопродуктивним інтерфейсом на основі мови програмування Dart.

Google AdMob – це мобільна рекламна платформа, яка дозволяє розробникам додатків заробляти на рекламі в своїх додатках на Android та iOS.

Скорочення:

СКБД – система управління базами даних.

МП – мова програмування.

ОС – операційна система.

ВСТУП

У сучасному світі, мобільні додатки стали невід'ємною частиною нашого життя. Завдяки їм, люди можуть бути на зв'язку з друзями та родиною, отримувати оновлення новин, здійснювати онлайн-покупки, працювати віддалено та багато іншого. Мобільні додатки дозволяють нам бути завжди під рукою, забезпечуючи зручність та швидкість доступу до необхідної інформації. За допомогою мобільних додатків можливо спростувати багато повсякденних задач для зручності людей. Також є чимало можливостей використати додатки для благодійних речей, одним із таких представлений «StandWithUkraine».

24 лютого минулого року країна агресор завдала перші ракетні удари по мирним містам України та розпочала повномасштабне вторгнення до країни. Народ України з перших днів розпочав згуртовуватися та допомагати одне-одному, країні, військовим, постраждалим внаслідок пошкодження цивільних будинків, інфраструктури, тощо. Українці розпочали добровільно йти до лав ЗСУ аби допомогти втримати міста та зміцнити армію. Але не всі можуть бути військовими, не кожна людина може перебувати безпосередньо на лініях зіткнень та тримати оборону, саме тому громадяни повинні допомагати кожен на своїй ділянці, – на заводі, лікарні, на підприємстві, сплачувати податок у важкий час або з ноутбуком в руках.

Саме в комп'ютерній сфері можна підтримувати Україну багатьма способами, одним з яких є просування мобільного додатку, всередину якого вмонтовані рекламні блоки, за допомогою яких люди без вкладень коштів можуть допомагати та донатити гроші на підтримку ЗСУ, постраждалому народу у надважкий час.

Пройшло більше року з початку повномасштабного вторгнення на територію України, за цей час було досягнуто багато успіхів, перемоги, але важливо пам'ятати, що багато людей переселених з регіонів де безпосередньо ведуться бойові дії, їхні домівки, автівки, речі зруйновані. Люди змушені переїжджати у інші міста без роботи, коштів для утримання родини. Багато

людей, котрі проживають на безпечних територіях залишилися без доходу, роботи, і це могло статися після руйнування місць роботи, скорочення штату співробітників, влучань ракет. Велика кількість народу знаходяться без надлишку коштів, якими могли би підтримати українську армію, допомогти іншим постраждалим родинам.

Саме тому, такий мобільний додаток надзвичайно необхідний, він є безкоштовним на платформі Google Play Store, за допомогою нього люди можуть без вкладень переглянути рекламні вставки, кошти з якого підуть безпосередньо у фонд підтримки ЗСУ та постраждалому народу України.

Актуальність роботи полягає вкрай необхідною допомогою збройних сил України, людям з тимчасово окупованих територій, постраждалим, дітям які залишилися без батьків, адже абсолютно безкоштовно кожна людина може допомогти, навіть якщо в неї немає на це належного фінансового стану.

Метою кваліфікаційної роботи є залучення коштів для ефективної допомоги Україні шляхом розробки програмного забезпечення з використанням мови програмування Dart та фреймворку WPF. Основними вимогами до ПЗ є розробка привабливого та зручного користувацького інтерфейсу, розробки гри для відволікання від повсякденних проблем, тестів з української мови для українців, а також для іноземців, для цього була додана англійська мова інтерфейсу. Також, метою є забезпечення стабільної роботи програмного забезпечення.

Також планується забезпечити постійне оновлення нашого додатку з новими функціями та змінами, щоб забезпечити наших користувачів найкращими можливими інструментами для вивчення української мови та відпочинку.

Об’єктом кваліфікаційної роботи є процеси роботи мобільного додатку з відтворенням рекламних блоків, тестами та грою. Актуальним питанням є автоматизований показ реклами, проходження тестувань та гри шляхом розробки програмного забезпечення. Робота спрямована на створення зручного інтерфейсу для користувачів, забезпечуючи зручну та швидку взаємодію з

додатком. Головна увага виділяється реалізації функцій, як автоматичне виведення доступних інтеграцій реклами при натиску або запуску ПЗ. Результати дослідження допоможуть визначити переваги та недоліки розробленого додатку і запропонувати рекомендації щодо подальшого вдосконалення та використання в практичних ситуаціях.

Предметом є розробка програмного забезпечення для автоматичного перегляду користувачами рекламних інтеграцій з використанням технологій Dart та Flutter. Предмет дослідження включає аналіз потреб користувачів, проектування то розробка гарного інтуїтивно-зрозумілого інтерфейсу. Предметом мобільного додатку, є його архітектура, модулі та компоненти, які визначають його функціональність та ефективність.

Для досягнення поставленої мети в роботі ставились та вирішувались наступні завдання:

– Дослідити існуючі аналоги ПЗ та їх програмні рішення.

– Проаналізувати потреби з метою визначення ключових функцій та можливостей, які має містити розроблений додаток.

– Реалізувати функціональні можливості, включаючи автоматичне виведення рекламних блоків на екран, реалізувати проходження тестування та гру.

– Розробити привабливий інтуїтивно-зрозумілий інтерфейс з підтримкою української та англійської мови, забезпечуючи зручну та ефективну роботу з додатком.

– Оцінити переваги та недоліки розробленого додатку з позицій зручності використання, функціональності та продуктивності.

– Запропонувати рекомендації щодо вдосконалення та розширення можливостей з урахуванням отриманих результатів та вимог користувача.

Для виконання поставлених завдань використовувались наступні методи та технологій: мови програмування Dart та фреймворк Flutter, включаючи роботу з базою даних, зображеннями та іншими компонентами.

Результати дослідження: у дипломній роботі розроблено програмне забезпечення, яке дозволяє:

– Забезпечити можливість перегляду користувачами рекламних блоків;

– Реалізувати можливість проходження тестування з української мови;

– Розробити зручний та інтуїтивно зрозумілий інтерфейс для користувача;

– Розробити підтримку декількох мов інтерфейсу;

– Реалізувати гру, яка буде відволікати користувачів від повсякденних проблем.

Важливим аспектом є відповідність даного продукту потребам користувачів та ринковим потребам. У сучасному світі розвиток програмного забезпечення є динамічним процесом, тому важливо проводити постійний моніторинг ринку та аналізувати досвід розробки аналогічних програмних продуктів. Даний розроблений мобільний додаток має гарні перспективи та актуальність на ринку протягом багатьох років

Отже, чітко простежується принципова необхідність рішення проблеми з розробкою мобільного додатку, який має максимально зручний дизайн, найголовніше – це можливість безкоштовно донатити на допомогу українському народу, вивчати мову та грати аби відволікатися від проблем.

Кваліфікаційна робота складається зі вступу, трьох розділів, висновку та додатків. Робота містить 64 сторінок основного тексту, 20 рисунків та 15 літературних джерел.

РОЗДІЛ 1.
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ РОЗРОБКИ МОБІЛЬНИХ ДОДАТКІВ

1.1 Дослідження предмету, цілей та особливостей додатку.

Мобільні додатки стали невід'ємною частиною нашого життя, надаючи нам зручність та доступність до різноманітних послуг та інформації в будь-який час та в будь-якому місці. Телефони знаходяться завжди під рукою в легкій доступності, який є майже у кожної людини. Розробка мобільних додатків зростає у популярності, оскільки вони дозволяють підприємствам та організаціям покращити комунікацію з клієнтами та забезпечити легкий доступ до послуг.

Основною особливістю додатку є акцент на підтримці України у важкий час, надання можливості кожному користувачеві долучитись до благодійності та підтримки країни.

Додаток розроблений для ОС Android на мові програмування Dart з використанням фреймворку Flutter. Для реклами буде використовуватися сервіс Google AdMob. Додаток забезпечить користувачів зручним, цікавим способом вивчення української мови, тадопоможе підтримати країну.

Проект розроблений з використанням сучасних технологій та мови програмування, що дозволить забезпечити найвищу якість та продуктивність додатку.

Дослідження цього проекту дозволить долучити більшу кількість людей до благодійності та підтримки України, а також забезпечить доступну та цікаву платформу для вивчення української мови.

1.2 Аналіз існуючих аналогів мобільних додатків

Існують декілька аналогів на ринку, які пропонують гру, тести з можливістю перегляду реклами, проте багато з них не вказують куди йдуть

кошти від перегляду користувачами рекламних інтеграцій. Додаток має перевагу у відношенні зручності користування та підтримки України, що робить його привабливим вибором для користувачів.

Одним з найкращих існуючих аналогів, які мають схожий функціонал, мають тести з вивченням українського діалекту, на мою думку, є Be with UA.

Be with UA – мобільний додаток, створений міжнародною IT компанією для ОС Android, яка має в багатьох містах Україні власні офіси та українських співробітників. Таким чином компанія вирішила підтримати Україну у важкий час та розробити програмний застосунок для допомоги [1].

Даний додаток дає змогу користувачам зі всього світу скористатися його зручним інтерфейсом, для користувачів поза межами України, які не розмовляють українською мовою, такий програмний засіб допомагає ознайомитися з культурою та діалектом.

В додаток інтегровані рекламні блоки, які відображаються для користувача одразу після завершення проходження тестування та відображення результатів.

Додаток має не різноманітний інтерфейс, про те це робить його зрозумілим та зручним (рис.1.1).

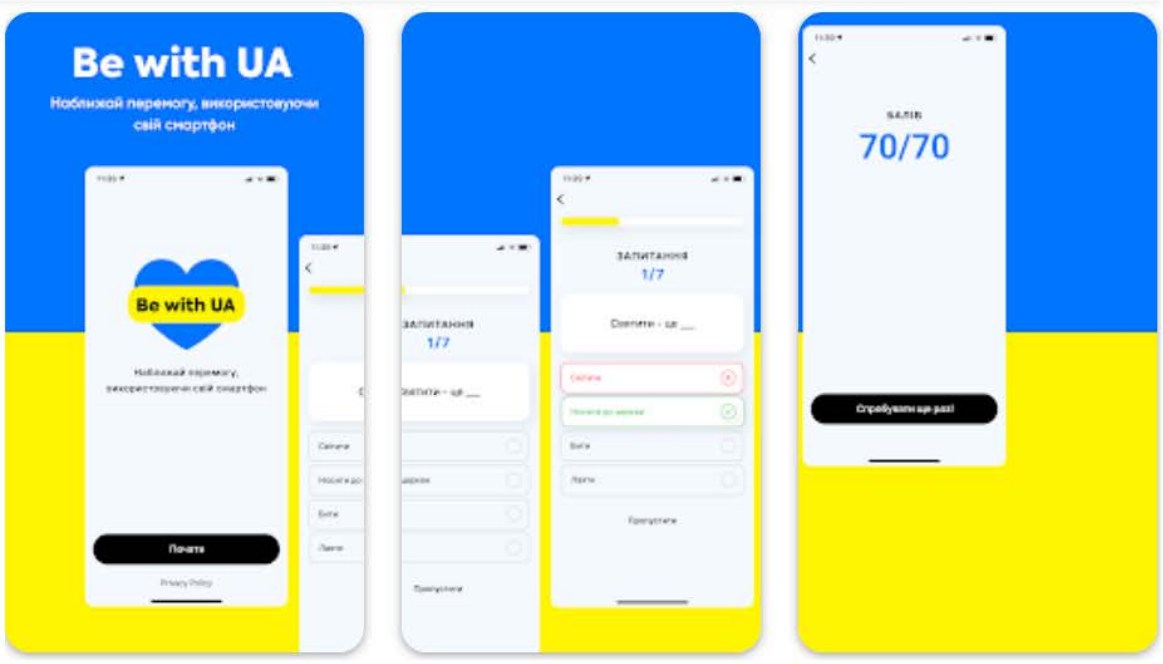

Рисунок 1.1 – Дизайн додатку Be with UA

Мобільний додаток «dymka». dymka – мобільний додаток, призначений для вивчення української мови, розділений на декілька категорій [2]. Застосунок містить можливість вивчення різних груп слів, а саме: синоніми, антоніми, омоніми, пароніми (рис.1.2).

Рисунок 1.2 – Дизайн додатку dymka

Програма – кишеньковий репетитор української мови, який дозволить вам навчатися з нуля як онлайн, так і без інтернету.

Додаток dymka стане вашим незамінним помічником у вивченні мови. Програмний засіб допоможе оновити знання та згадати те, що навчали ще у школі.

Завдяки інтерактивним карткам можливо поповнити свій словниковий запас. А різні завдання допоможуть покращити промову та розмовляти без граматичних та інших помилок.

Проте існують ще декілька інших конкурентів, наприклад мобільний застосунок – «JavelinPaint».

Мобільний додаток «JavelinPaint». Мобільний додаток "JavelinPaint" – це захоплива іграшка для тих, хто хоче відволіктися та розважитися. Гравці можуть

малювати траєкторії javelin та збивати ворожі цілі [3]. Це ідеальний спосіб відволіктися від повсякденних проблем та розслабитися (рис.1.3).

Рисунок 1.3 – Мобільний додаток «JavelinPaint»

Ігровий процес має прості правила, але вимагає деякої майстерності та стратегії. Це дозволяє гравцям відчувати виклик та розвивати свої навички, що зробило б гру популярною серед користувачів.

Загалом, це захопливий та розважальний додаток, який може привернути увагу користувачів. Крім того, додаток може стати чудовим інструментом для тренування реакції та точності, що може бути корисним для розвитку когнітивних навичок. Також, можливість зігнати злість на ворога у віртуальному просторі може допомогти людям відволіктись та зняти стрес під час нервового періоду., які шукають спосіб допомогти країні через використання мобільних додатків.

"JavelinPaint" – може бути цікавим для людей, які люблять спорт та виклики. Гра дає можливість практикуватись в точності та меткості, що може бути корисним для спортсменів, особливо тих, хто займається метанням копи. Також, гра може бути корисною для тих, хто працює в армії або військових

відділах, як спосіб тренування та підвищення меткості. В цілому, він є цікавим та корисним додатком для тих, хто цінує точність, виклики та відволікання від повсякденних проблем.

Отже, серед розробленого програмного засобу існують схожі аналоги, проте для багатьох українців підтримка країни є важливим чинником, який впливає на їх вибір продуктів і послуг. Тому, включення можливості безкоштовного перегляду реклами та перерахунку коштів на підтримку України може значно підвищити привабливість додатку, якого майже ні в кого не існує.

Додаток, в якому поєднані всі необхідні атрибути як для підвищення знань, гри , за допомогою якої можливо відволіктися від повсякденних проблем, по при це підтримка держави у складний час може стати невід'ємною частиною життя людей по всьому світу.

1.3 Методи розробки мобільних додатків

Розробка мобільних додатків включає в себе декілька методів та етапів, що дозволяють створити функціональний та ефективний продукт. Деякі з них описані нижче:

Першим етапом розробки мобільного додатку є визначення мети, яку він повинен вирішувати, та аудиторії, яка його буде використовувати. Це допоможе розробити відповідну функціональність та інтерфейс.

Розробка концепції та дизайну. Наступним етапом є розробка концепції додатку, його інтерфейсу та дизайну. Це включає в себе створення макетів та прототипів, що дозволяють перевірити ефективність дизайну та інтерфейсу.

Розробка функціональності. Після визначення мети та дизайну додатку, необхідно розробити його функціональність. Це включає в себе розробку алгоритмів, баз даних, інтеграцію з різними сервісами та інші функції.

Тестування та налагодження. Після розробки функціональності необхідно провести тестування додатку для перевірки його працездатності та виправити можливі помилки.

Реліз та підтримка. Після успішного тестування додаток можна викласти до сертифікованого магазину мобільних додатків та почати його підтримку. Це включає в себе оновлення функціональності, виправлення помилок та забезпечення безпеки.

Декілька найпоширеніших методів зручної розробки:

Нативна розробка: цей метод передбачає використання офіційних SDK (Software Development Kit) для кожної платформи окремо [4]. Для розробки мобільних додатків для Android використовується Java, Dart або Kotlin, а для iOS – Objective-C або Swift. Цей підхід дозволяє максимально використовувати можливості платформи, але вимагає розробки окремого коду для кожної платформи.

Фреймворки гібридної розробки: дозволяють розробляти додатки, використовуючи веб-технології, такі як HTML, CSS та JavaScript, і упаковувати їх у нативні контейнери для розповсюдження в магазинах додатків. Найпопулярніші фреймворки для гібридної розробки включають React Native, Flutter та Ionic [5]. Вони дозволяють швидко розробляти додатки для обох платформ, використовуючи одну кодову базу.

Кросплатформені фреймворки: ці фреймворки дозволяють розробляти додатки, використовуючи спеціальну мову програмування, яка компілюється до нативного коду для кожної платформи. Найпопулярнішими кросплатформеними фреймворками є Xamarin, Cordova та PhoneGap [6]. Вони дозволяють створювати додатки, які працюють на обох платформах, використовуючи один код.

Крім традиційних методів розробки мобільних додатків, таких як Waterfall та Agile, існує декілька інших методів, які можуть бути використані залежно від типу проекту та його особливостей. Наприклад:

Rapid Application Development (RAD): цей метод підходить для проектів, які потребують швидкої розробки та випуску додатку. Розробники використовують засоби автоматизації, щоб прискорити процес розробки та зменшити кількість ручної роботи [7].

Lean Development: цей метод зосереджений на мінімізації витрат та максимальному використанні ресурсів. Розробники зосереджуються на створенні основних функцій та функцій, які найбільше цікавлять користувачів.

DevOps: цей метод об'єднує розробку та оперативну діяльність [7]. Команди розробників та тестувальників працюють разом над розробкою додатку та його випуском. Це дозволяє швидко вносити зміни та випускати оновлення.

Hybrid Development: цей метод поєднує різні технології та підходи до розробки мобільних додатків. Розробники можуть використовувати як нативний код, так і HTML5/CSS/JavaScript для створення додатку [7].

Extreme Programming (XP): цей метод зосереджений на максимальній зручності користувача та якості додатку. Розробники створюють короткі ітерації та залучають користувачів до процесу тестування та збору відгуків [7].

Ці методи можуть бути використані окремо або поєднані між собою, залежно від потреб проекту та вимог замовника.

1.4 Інтеграція рекламних блоків у мобільних додатках

Інтеграція рекламних блоків є важливою частиною монетизації мобільних додатків. Це дозволяє розробникам отримувати прибуток зі своїх додатків, не залежно від того, чи збираються користувачі купувати платний контент або платити за додаткові функції [8].

Для цього розробники можуть використовувати різноманітні сервіси, такі як AdMob від Google, Facebook Audience Network, Unity Ads та багато інших. Ці сервіси дозволяють розробникам створювати рекламні блоки та налаштовувати різні параметри в особистих аккаутах, такі як час показу, частоту показу та таргетування аудиторії [8].

Інтеграція може бути досить зручною, оскільки дозволяє збільшити прибуток від мобільного додатку без великих зусиль. Розробники можуть використовувати рекламні сервіси для автоматичного показу в додатку, що зменшує необхідність ручного налаштування. Крім того, сервіси зазвичай

забезпечують велику кількість оферт, що збільшує шанси на вибір оптимального варіанту для конкретного додатку.

Проте, розробники повинні мати на увазі, що інтеграція рекламних блоків може впливати на користувацький досвід додатку. Якщо реклама відображається надто часто або займає занадто багато місця на екрані, це може зіпсувати враження користувачів від додатку.

Дохід може бути значним і залежить від багатьох факторів, включаючи кількість користувачів, кількість показів, тип, аудиторію додатка та бізнес-модель компанії.

Найпоширенішою формою в мобільних додатках є банер. Її дохід зазвичай розраховується за кількість показів або. Розміщення банерів може бути різним, наприклад в верхній частині екрана, внизу, між екранами тощо.

Іншою формою реклами є відеореклама, яка може бути розміщена як перед відео, так і в середині. Її дохід зазвичай розраховується за кількість переглядів.

Також популярними є інтерактивні формати, такі як ігри, опитування та інші форми взаємодії з користувачем. Дохід від таких форматів може бути різним, залежно від бізнес-моделі та цільової аудиторії.

Щодо сервісів і зручностей, платформи пропонують аналітику, можливість налаштування таргетингу та інші корисні інструменти для монетизації. На прикладі сервісу Google AdMob, використовуючи лише зареєстрований аккаунт є можливість робити всі можливі налаштування за хвилини.

Найпоширенішою бізнес-моделлю для мобільних додатків є реклама, яка є основним джерелом доходу для більшості безкоштовних додатків. За даними досліджень, в 2021 році від показу в мобільних додатках компанії заробили більше 240 мільярдів доларів [9].

Однак, дохід від може бути значно нижчим для додатків з невеликою кількістю користувачів або для додатків, які не цікаві для рекламодавців. Також важливо пам'ятати, що більшість платформ забирає частину доходу від показу тому від ціни за тисячу показів може залежати багато.

Крім того, деякі додатки можуть використовувати інші бізнес-моделі для заробітку, такі як продаж преміум-підписок або внутрішньої валюти, яка може бути використана для покупки додаткових функцій або продуктів. Такі моделі можуть бути більш прибутковими, особливо для додатків з активною базою користувачів, які готові платити за додаткові функції або переваги.

ВИСНОВКИ ДО РОЗДІЛУ 1

Аналіз предметної області розробки мобільних додатків з залученням рекламних показів показує, що реклама в мобільних додатках є важливою складовою доходу для багатьох компаній, які розробляють додатки. Інтеграція рекламних блоків у додатки стає все більш популярною, оскільки це дає можливість залучити додатковий дохід без збільшення вартості додатка для користувачів. Крім того, це дозволяє показувати користувачам рекламу, яка може бути для них цікавою та корисною, приносити користь, допомагаючи Україні у складний час.

РОЗДІЛ 2.
ВИБІР ПРОГРАМНИХ ЗАСОБІВ ДЛЯ РЕАЛІЗАЦІЇ ДОДАТКУ

2.1 Вибір архітектури для розробки продукту

При виборі архітектури для розробки мобільного додатку «StandWithUkraine» для ОС Android можна розглянути такі варіанти:

1) Model-View-Controller (MVC): ця архітектура використовується в багатьох Android-додатках і передбачає розділення коду на три частини: модель, представлення та контролер. Це дозволяє розібрати логіку додатку на більш прості та логічні компоненти [10].

2) Model-View-ViewModel (MVVM): ця архітектура побудована на підході Model-View-Controller, але з використанням моделі-вью-моделі (ViewModel), яка забезпечує зв'язок між представленням та моделлю даних. Це дозволяє підтримувати стан додатку та забезпечувати більш просту роботу з даними [10].

3) Clean Architecture: ця архітектура побудована на засадах SOLID та DRY, що дозволяє зберігати код додатку зрозумілим та легким для підтримки. Clean Architecture розбиває додаток на різні рівні (Domain, Presentation та Infrastructure) та забезпечує зв'язок між ними через інтерфейси [11].

4) Reactive Programming: цей підхід використовується для розробки додатків, які мають різні потоки даних та їх обробки. Reactive Programming дозволяє зберігати код додатку легким для підтримки та зменшити кількість коду.

При виборі архітектури для мобільного додатку можна зосередитися на тому, щоб забезпечити зручну розробку та підтримку додатку, а також його високу продуктивність та безпеку. Вибір архітектури залежить від специфіки проекту та вимог до системи.

Зважаючи на тему дипломної роботи про створення мобільного додатку, вибір архітектури є критичним етапом розробки. Він повинен бути зроблений з

урахуванням багатьох чинників, таких як вимоги функціональності, розширюваність, масштабованість, продуктивність та безпека.

Для створення додатку було обрано архітектуру MVVM, яка є однією з найбільш популярних архітектур в розробці мобільних додатків на Android. Основна ідея полягає в тому, що вся бізнес-логіка знаходиться в ViewModel, яка відокремлює представлення від моделі. Вона взаємодіє з репозиторієм та джерелом даних, виконує запити до мережі. View, у свою чергу, відповідає за відображення даних, але не знає нічого про бізнес-логіку.

Крім того, для підвищення безпеки було обрано підхід "за замовчуванням безпечності" (security by default), що передбачає застосування максимальної кількості заходів безпеки за замовчуванням та надання користувачу можливості змінювати налаштування відповідно до своїх потреб.

На цьому етапі розробки мобільного додатку, після вибору архітектури, важливо правильно обрати технології для розробки додатку.

Основні технології, які зазвичай використовуються для розробки це Java і Dart. Обидві мови програмування мають свої переваги та недоліки. Java є більш старою мовою програмування і має більшу кількість ресурсів, але Dart є більш безпечною та зручною мовою, яка може допомогти уникнути багатьох проблем, пов'язаних з Java.

Для розробки необхідно використовувати такі технології, як Android SDK, Android Studio, розробки UI (наприклад, XML), технології збереження даних (наприклад, Firebase), технології розробки технічної частини.

У виборі технологій для розробки мобільного додатку дуже важливо враховувати особливості проекту та потреби користувачів. Наприклад, якщо користувачі зазвичай використовують старіші моделі телефонів в поєднанні з новими, можливо, краще використовувати Dart.

Загальний підхід до вибору архітектури розробки мобільного додатку має бути збалансованим та зорієнтованим на конкретні потреби проекту.

2.2 Вибір мови для програмування додатку

Для розробки мобільного додатку можуть бути використані різні мови програмування, такі як Java, Kotlin, Dart C++, Python, Swift, Objective-C і багато інших. Проте, у даному випадку було обрано мову програмування Dart, як найбільш оптимальну з точки зору функціональності та ефективності для даного додатку. Крім того, для створення користувацького інтерфейсу можна використовувати різні технології, такі як XML, HTML, CSS та інші.

Мова програмування Dart:

Мова програмування Dart була розроблена компанією Google як мова для створення високопродуктивних веб-додатків та мобільних додатків. Вона має декілька переваг в порівнянні з іншими мовами програмування, такі як ефективна обробка асинхронних операцій, зручна інфраструктура для управління пам'яттю, а також потужні засоби для обробки даних та взаємодії з іншими веб-сервісами [13].

Dart – широко використовується для створення мобільних додатків за допомогою Flutter – фреймворка для розробки нативних додатків для платформ Android та iOS. Flutter, можемо продивитися приклад коду у середовищі VS Code (рис. 2.1). Це дозволяє розробникам створювати високоякісні мобільні додатки з вражаючим інтерфейсом користувача та високою продуктивністю [13].

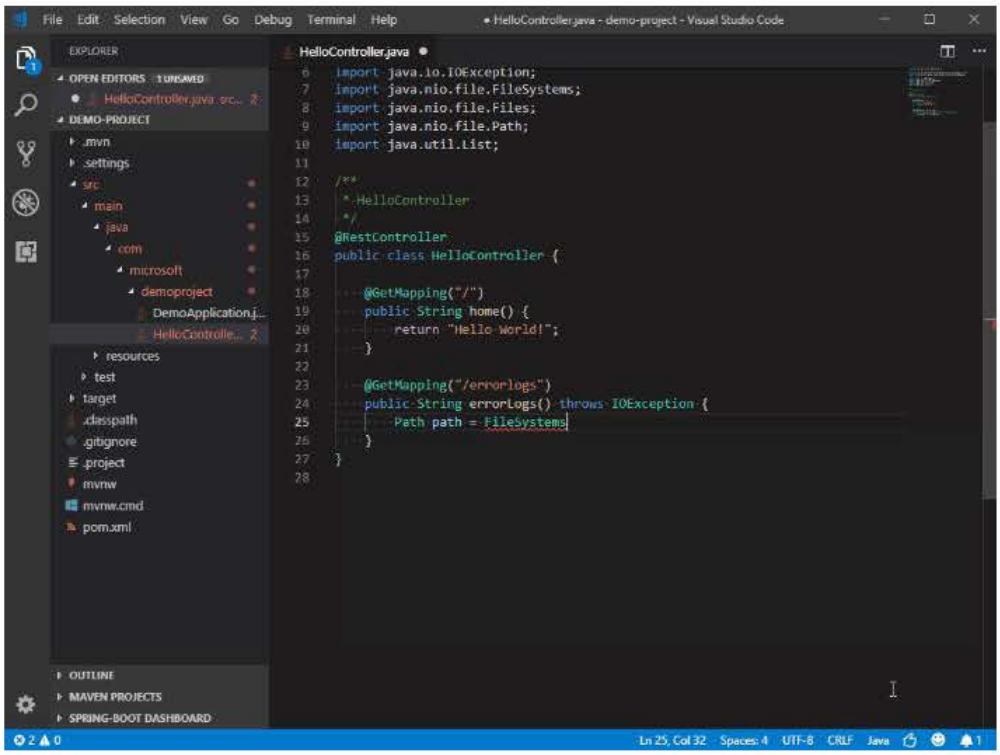

Рисунок 2.1 – Мова програмування Dart з фреймворком Flutter у середовищі VS Code

Розберемо приклад простого калькулятору. Калькулятор на Flutter та Dart складається з двох основних частин – інтерфейсу та логіки обчислення. Інтерфейс користувача може бути створений за допомогою віджетів Flutter, таких як текстові поля, кнопки, та інші. Логіка обчислення може бути реалізована за допомогою мови програмування Dart, яка може обробляти введені користувачем дані та здійснювати математичні операції.

В цьому випадку розробники можуть використовувати можливості даної мови програмування, такі як інтерполяція рядків для виводу результатів обчислення, та асинхронні операції для забезпечення швидкості та продуктивності додатку. Також можна використовувати готові бібліотеки Flutter для покращення функціональності додатку.

Однією з головних переваг цієї МП є його простота та зрозумілість. Він має синтаксис, який дуже схожий на Java та JavaScript, що дозволяє розробникам легко вивчити його. Також є статично типізованою мовою, що дозволяє зменшити кількість помилок, які можуть виникнути при розробці.

Ще один приклад простого мобільного додатку на Flutter з використанням Dart – "Hello World" додаток. У ньому створимо просту сторінку з текстом "Hello World". Для початку необхідно створити новий проект в Android Studio та додати необхідні залежності для Flutter.

Після створення проекту та додавання залежностей, можемо створити новий файл з кодом додатку. В ньому створимо новий StatefulWidget та відображатимемо текст "Hello World" на екрані:

```
dartCopy code:
import 'package:flutter/material.dart';
void main() => runApp(MyApp());
class MyApp extends StatefulWidget {
@override _MyAppState createState() => _MyAppState();
}
class _MyAppState extends State<MyApp> {
@override Widget build(BuildContext context)
{
return MaterialApp( title: 'Hello World', home: Scaffold( appBar: AppBar( title: Text('Hello
World'), ), body: Center( child: Text( 'Hello World', style: TextStyle(fontSize: 24),
),
 ),
  ),
   );
    }
```

У цьому коді створюємо новий StatefulWidget, який є основою для мобільних додатків на Flutter. У методі build повертаємо MaterialApp, який відповідає за основний дизайн додатку. У ньому вказуємо заголовок та домашню сторінку додатку.

На домашній сторінці створюємо Scaffold, який відповідає за основну структуру додатку, та створюємо AppBar та відображаємо на екрані текст "Hello World". Для відображення тексту використовуємо Center та Text Widget. Таким чином отримуємо простий проект додатку з виводом тексту.

Додатки, розроблені з використанням Dart та Flutter, мають високу продуктивність та швидкість, що дозволяє розробникам швидко створювати ефективні мобільні додатки з чудовим інтерфейсом та функціональністю. Крім того, Dart має простий синтаксис та забезпечує багатий набір інструментів для розробки [13].

Отже, Dart є відмінним вибором для розробки мобільних додатків з використанням Flutter, особливо якщо потрібна висока продуктивність та швидкість. Ця мова програмування також проста в освоєнні та надає розробникам зручні інструменти для розробки.

Мова програмування Java:

Java є однією з найпопулярніших мов програмування у світі, використовується для розробки різноманітних програм, в тому числі мобільних додатків для ОС Android. Її використання дозволяє розробникам створювати надійний і ефективний код, що працює на будь-яких платформах [14].

Одним з прикладів мобільного додатку, розробленого на Java для Android, є додаток "ToDo List", що дозволяє користувачам створювати список завдань та відстежувати їх виконання. Нижче наведений приклад створення нового проекту у середовищі Android Studio (рис.2.2) та код для створення простого списку завдань у додатку.

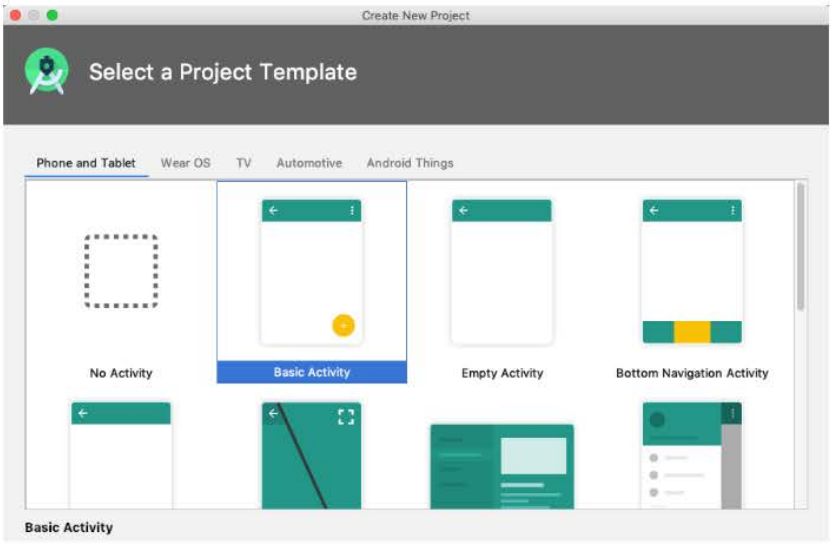

Рисунок 2.2 – Приклад створення нового проекту на Java з різними інструментами

Лістинг коду для створення "ToDo List":

```
public class MainActivity extends AppCompatActivity {
private RecyclerView recyclerView;
private RecyclerView.Adapter adapter;
private RecyclerView.LayoutManager layoutManager;
private ArrayList<String> tasks;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    tasks = new ArrayList<String>();
    tasks.add("Task 1");
    tasks.add("Task 2");
    tasks.add("Task 3");
    recyclerView = findViewById(R.id.recyclerView);
    recyclerView.setHasFixedSize(true);
    layoutManager = new LinearLayoutManager(this);
    adapter = new TaskAdapter(tasks);
    recyclerView.setLayoutManager(layoutManager);
    recyclerView.setAdapter(adapter);
  }
}
```

У даному коді створюється Activity з RecyclerView, до якого додається LinearLayoutManager та адаптер TaskAdapter. В адаптері задається список завдань tasks, які відображаються в RecyclerView.

У TaskAdapter задається ViewHolder для елементів списку, та реалізується метод onBindViewHolder, що дозволяє відображати текст завдань у відповідному TextView.

Java має велику кількість фреймворків і бібліотек, що дозволяє розробникам ефективно працювати зі звичайними задачами. Також підтримує багато поточність, що дозволяє розробляти програми з високим рівнем

паралелізму і оптимізації. Має велику спільноту розробників, яка допомагає підтримувати та розвивати мову і її інструменти.

Одним з прикладів великого та світового мобільного додатку, написаного на Java, є додаток для замовлення таксі Uber. Він має велику кількість функцій, таких як вибір місця призначення, відстеження водія на мапі, обмін повідомленнями з водієм і оплата кредитною карткою. Додаток розроблений з використанням фреймворка Android SDK, який базується на мові програмування Java.

У загальному, мова програмування Java є дуже потужним інструментом для розробки програм, яка використовується у багатьох великих і складних додатках. Вона дозволяє розробникам створювати ефективні та надійні додатки для різних платформ.

Проте код написаний на цій МП є досить громіздким та має обмеження в плані рекламних інтеграцій, що є основною задачею для роботи проекту.

Мова програмування Kotlin:

Мова програмування Kotlin – це мова програмування, що базується на Java Virtual Machine (JVM) та була розроблена компанією JetBrains. Kotlin є мовою програмування загального призначення та підтримує об'єктно-орієнтований та функціональний стилі програмування. Одним з головних переваг Kotlin є його висока сумісність з Java, що дає змогу використовувати код Java у проектах Kotlin [15].

Приклад коду, для створення простої програми, що виводить рядок тексту на екран:

```
fun main() {
    println("Hello, world!")
}
```

Цей код простий та зрозумілий, з лаконічним синтаксисом та читабельними назвами функцій. Також МП має безліч інших функцій, що

дозволяють розробникам швидко та ефективно створювати складніші програми, такі як андроїд-додатки та веб-сервіси.

Ще однією з переваг є його безпека та надійність, завдяки системі типів, що дозволяє виявляти помилки під час компіляції коду. Крім того, він підтримує корутини, що дозволяють зручно та ефективно вирішувати задачі з асинхронним програмуванням.

Загалом, Kotlin є потужною мовою програмування, що може бути використана для різних типів проектів, включаючи мобільні додатки та веб-сервіси. Його лаконічний та зрозумілий синтаксис разом з високою сумісністю робить досить привабливим вибором для розробників, нижче на рисунку 2.3 можемо спостерігати порівняння коду на цих двох мовах програмування, зліва Java, справа Kotlin .

```
// switch case
switch (option) {
  case 1:
    print('option 1');
    break;
  case 2:
    print('option 2');
    break;
  default:
    print('option is not defined');
    break;
}

// for loop
for (var i = 0; i < 5; ++i) {
  print('count: $i');
}

for (var item in list) {
  print(item);
}

// if-else
if (option == 1) {
  print('hello');
} else {
  print('bye');
}

// while loop
while (option != 0) {
  print('current option: $option');
  option--;
}
```

```
// when
when(option) {
    1 -> println("option 1");
    2 -> println("option 2");
    else -> {
        println("option is not defined")
    }
}

// for loop
for (index in 0..5) {
    println("count: $index")
}

for(item in list) {
    println(item)
}

// if-else
if (option == 1) {
  println("hello")
} else {
  println("bye");
}

// while loop
while (option != 0) {
    println("current option: $option")
    option--;
}
```

Рисунок 2.3 – Порівняння коду на двох мовах програмування

Одним з найпопулярніших прикладів використання Kotlin є розробка мобільних додатків для платформи Android. Це офіційна мова розробки Android,

і вона надає розробникам багато інструментів та можливостей для розробки швидких та ефективних додатків.

Основна перевага полягає в тому, що дозволяє розробникам писати менше коду, зменшуючи кількість помилок та спрощуючи процес розробки. Також пропонує багато функцій, що забезпечують безпеку програми, наприклад, безпечні нульові значення та безпечне приведення типів.

Крім того, може бути використаний для розробки серверних додатків, що забезпечує його універсальність та працездатність в різних областях програмування.

Код чистий і сучасний синтаксис, підтримує багато функцій програмування та має вбудовану підтримку нульової безпеки. Крім того, Kotlin має високу продуктивність, що робить його ідеальним для створення швидких та надійних мобільних додатків.

Однак, незважаючи на всі переваги Kotlin, він не так поширений, як Java. Багато бібліотек та фреймворків все ще підтримують лише Java, що може бути проблемою для деяких розробників.

Загалом, Kotlin є потужною мовою програмування з багатьма функціями та інструментами, які допомагають розробникам створювати надійні та ефективні мобільні додатки для платформи Android, але є досить молодим, навіть враховуючи його стислий код, на мою думку все одно не може конкурувати з Dart. Враховуючи все, та додаючи, що Dart розроблений компанією Google, саме від яких ми будемо підключати рекламні блоки та отримувати дохід, та є максимально сумісний з ним.

В даному випадку максимально ефективним та правильним у виборі мови програмування буде Dart з фреймворком Flutter. Данна мова є досить сучасним рішенням, легко поєднується з сервісами Google, що в даному випадку є важливим.

2.3 Вибір СКБД для управління контентом додатку

Для управління контентом додатку можна використовувати різні системи керування базами даних (СКБД), залежно від вимог проекту та його масштабів.

Одним з найбільш популярних СКБД є MySQL. Вона є безкоштовною та відкритою, має велику спільноту користувачів та широкі можливості. MySQL добре підходить для невеликих та середніх проектів зі стандартними потребами управління даними [16].

Ще однією опцією є PostgreSQL, яка також є відкритою та має безкоштовну ліцензію. PostgreSQL має високу надійність, підтримує транзакції та ACID властивості, тому вона часто використовується в більш складних проектах з багатофункціональними потребами управління даними.

Для проектів зі складною логікою та великим обсягом даних можна розглянути СКБД Oracle, яка пропонує велику кількість можливостей для роботи з даними та розширенням функціоналу.

Також можна використовувати NoSQL СКБД, наприклад MongoDB, для роботи з невструктурованими даними.

У випадку з додатком "StandWithUkraine" можна розглянути використання PostgreSQL, MySQL або Firebase оскільки вони надійні та мають широкі можливості для управління даними в режимі реального часу.

СКБД PostgreSQL:

PostgreSQL – це потужна, відкрита реляційна система керування базами даних, яка дозволяє зберігати велику кількість даних і забезпечує широкі можливості для роботи з ними. Вона підтримує багато типів даних, включаючи рядкові, числові, дати, час, географічні та інші.

PostgreSQL відомий своєю надійністю, стабільністю і безпекою. Вона має вбудовану систему захисту даних, яка забезпечує безпечне зберігання даних і захист від несанкціонованого доступу. Крім того, підтримує транзакції, що дозволяє уникнути помилок при зміні даних і забезпечує цілісність даних [16].

Однією з основних переваг – є те, що вона є вільною і відкритою системою. Це означає, що ви можете використовувати її безкоштовно, використовувати її на будь-якому обладнанні, налаштовувати її на свій смак і вносити зміни в код.

Даний СКБД є масштабованим рішенням, що дозволяє працювати з даними великих розмірів і підтримувати багато користувачів. Вона також підтримує високу продуктивність і швидкість обробки даних.

Наприклад, приклад коду для створення таблиці з даними в PostgreSQL:

```
CREATE TABLE users (
  id SERIAL PRIMARY KEY,
  name VARCHAR(50) NOT NULL,
  email VARCHAR(50) UNIQUE NOT NULL,
  password VARCHAR(100) NOT NULL
);
```

Що стосується вибору для управління контентом додатку, PostgreSQL є гідним варіантом завдяки своїм перевагам. Вона надає високу рівень надійності, безпеки та масштабованості, що особливо важливо для проектів з великою кількістю користувачів [16].

У виборі також важливо враховувати масштаб проекту та його вимоги до швидкодії. PostgreSQL може працювати з великою кількістю даних та запитів, що робить її відмінним варіантом для додатків з великою кількістю користувачів та багатофункціональними вимогами до бази даних.

Отже, PostgreSQL є гідним вибором СКБД для додатків з великою кількістю користувачів та складними вимогами до бази даних. Її можна використовувати для забезпечення надійного та безпечного управління контентом додатку з можливістю масштабування.

СКБД MySQL:

MySQL – є однією з найбільш поширених СКБД у світі, вона є відкритою і безкоштовною використовувати для більшості випадків. MySQL підтримує майже всі функції, які можна очікувати від СКБД, включаючи транзакції, індексацію, внутрішній і зовнішній ключі, процедури та функції [16].

Вона також має добре розвинену спільноту користувачів та документацію, яка є доступною для початківців. Вона також підтримує транзакції та

ізольованість, що робить її ідеальною для використання в додатках з великою кількістю запитів до бази даних.

Нижче наведено приклад коду, який демонструє, як взаємодіяти з базою даних MySQL з допомогою мови програмування Python та бібліотеки PyMySQL:

```
import pymysql
# З'єднання з базою даних
conn = pymysql.connect(host='localhost', port=3306, user='username', passwd='password', db='database')
# Створення курсора
cur = conn.cursor()
# Виконання запиту
cur.execute("SELECT * FROM users")
# Отримання результатів запиту
result = cur.fetchall()
# Виведення результатів
for row in result:
    print(row)
# Закриття курсора та з'єднання
cur.close()
conn.close()
```

У цьому прикладі підключаємося до бази даних MySQL з допомогою бібліотеки PyMySQL та виконали запит на вибірку даних з таблиці "users". Результати були виведені у вигляді рядків.

Узагальнюючи, MySQL є потужною та надійною СКБД, яка може бути використана для реалізації системи керування контентом мобільного додатку, який дозволяє забезпечувати стабільну та продуктивну роботу веб-додатків на різних операційних системах. Його вбудовані модулі та можливості дозволяють легко налаштовувати його для потреб конкретного додатку.

СКБД Firebase Realtime Database – це гнучка та масштабована система управління базами даних у режимі реального часу, розроблена компанією Google. Це одна з багатьох послуг, які пропонуються Firebase, і вона надає

розробникам можливість зберігати та синхронізувати дані у режимі реального часу між клієнтськими додатками та хмарною інфраструктурою Firebase.

Одним з ключових понять є "слухачі" (listeners). Розробники можуть створювати слухачі для відстеження змін в даних. Це означає, що коли дані змінюються, слухачі отримують сповіщення і можуть обробляти оновлені дані у режимі реального часу.

Також підтримує можливість роботи у відсутності з'єднання з Інтернетом. Коли пристрій знову отримує з'єднання, всі зміни, зроблені локально, автоматично синхронізуються з сервером.

Завдяки своїм можливостям синхронізації даних у реальному часі, Firebase Realtime Database є популярним вибором для розробки колективних додатків, чатів, групових редакторів, онлайн-ігор та багатьох інших сценаріїв, де в залежності від актуальності даних є критичними. Firebase Realtime Database також інтегрується з іншими сервісами Firebase, такими як Firebase Authentication для автентифікації користувачів, Firebase Cloud Messaging для надсилання повідомлень та Firebase Hosting для хостингу веб-сторінок і файлів.

Це і сервер, і база даних, і хостинг, і автентифікація в одній платформі. Так, Firebase надає розробникам API, який синхронізує дані користувачів і зберігає їх в хмарному сховищі.

Узагалі, Firebase Realtime Database є потужним інструментом для розробки додатків, які вимагають синхронізації даних у режимі реального часу. Він дозволяє зосередитися на розробці функціоналу додатку, не витрачаючи багато часу на розробку складних механізмів синхронізації та управління даними. Завдяки Firebase Realtime Database розробники можуть швидко створювати потужні додатки, які надають користувачам оновлені дані у режимі реального часу.

Висновком до вибору СКБД, для розробки був обраний Firebase Realtime Database, за його зручність, доступність та багатофункціональність.

2.4 Сервіс рекламного монетизації мобільних додатків від Google AdMob

Google AdMob – це сервіс рекламного монетизації мобільних додатків від Google. Він дозволяє розміщувати рекламу у додатках під Android та IOS.

Основні переваги Google AdMob:

Надійний та стабільний сервіс, який пропонує різноманітні формати, що відповідають потребам різних додатків та аудиторій.

Можливість відстеження показників ефективності в верифікованому аккаунті у режимі реального часу, приклад наведено нижче (рис. 2.5).

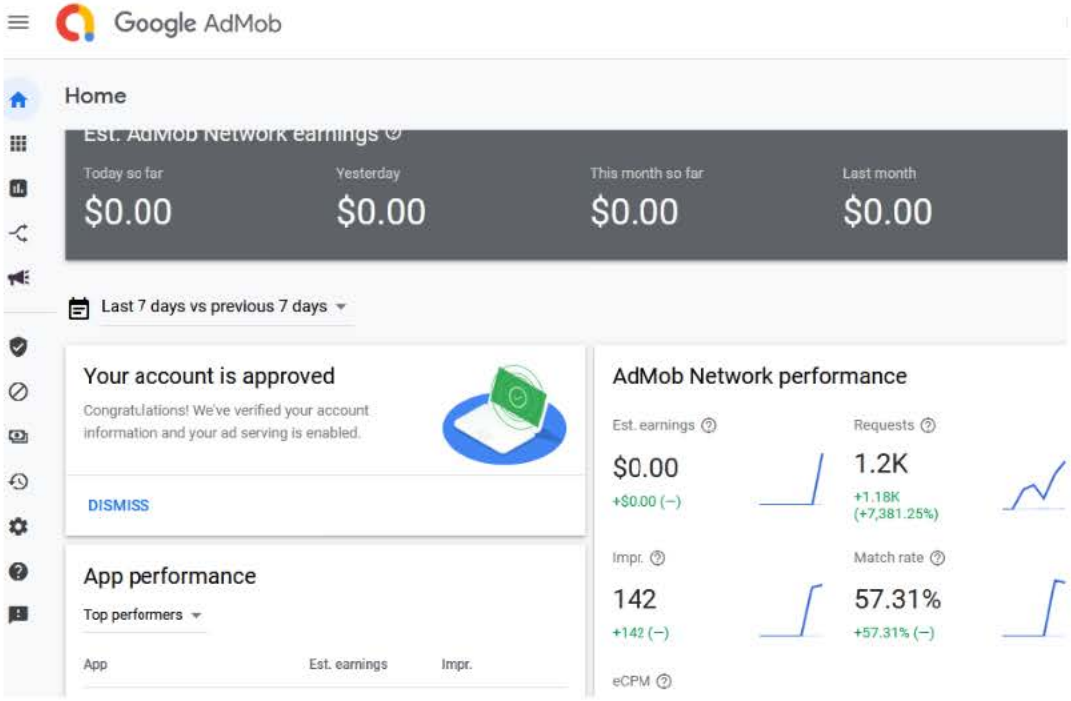

Рисунок 2.5 – Інтерфейс головної сторінки кабінету Google AdMob

Зручний інтерфейс для управління кампаніями та настройки їх параметрів.

Інтеграція з іншими продуктами Google, такими як Google Analytics та AdWords, для більш ефективного управління.

Розширена аналітика та можливість отримання детальної статистики щодо діяльності додатку та показників.

Гнучке налаштування показу на основі віку, статі, географії, інтересів користувачів та інших факторів.

Однією з головних переваг Google AdMob є його відносна легкість використання, що робить його дуже популярним серед розробників мобільних додатків. Більшість розробників вибирають Google AdMob як свій основний рекламний сервіс, оскільки він надає можливість максимально ефективно монетизувати свій додаток.

Приклад заробітку:

Припустимо, є мобільний додаток, який має більше 100 000 завантажень і користувачі активно взаємодіють з ним. Використовуємо формат "попереднього перегляду" (pre-roll), що означає, що користувачі будуть переглядати рекламу перед тим, як перейти далі.

За один перегляд заробіток 5 центів. Якщо в день додаток має 10 000 переглядів відео, заробіток складає $150 в день (10 000 * 30% * $0.05 = $150).

Якщо є бажання збільшити дохід, можна спробувати налаштувати блоки в додатку таким чином, щоб вони з'являлися під час відтворення, а не перед ним.

Загалом, з Google AdMob є найкращою пропозицією на ринку в даний момент, враховуючи зручність а також сертифіковані поки від офіційних представників Google, тому в даному мобільному застосунку буде використовуватися саме цей сервіс.

ВИСНОВКИ ДО РОЗДІЛУ 2

У розділі "Вибір програмних засобів для реалізації додатку" було проаналізовано різні інструменти та технології для створення мобільних додатків, різновиди мов програмування, вибір СКДБ для реалізації проекту а також обрано сервіс інтегрованих рекламних блоків в мобільний додаток з прикладом заробітку.

РОЗДІЛ 3.
МОДЕЛЮВАННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

3.1 Загальні можливості мобільного додатку «StandWithUkraine»

Створення мобільного додатку для ОС Android та підтримки України вимагається виконання наступних загальних вимог:

1. Розробка додатку має бути здійснена з використанням найновіших технологій та забезпечувати високу швидкість роботи додатку.
2. Дизайн додатку має бути зрозумілим та простим для користувача, з урахуванням стилю та колористики прапору України.
3. Додаток повинен бути безпечним та має мати захист від шкідливих атак, забезпечуючи конфіденційність даних користувачів.
4. Додаток має мати зручний та легкий інтерфейс для перегляду рекламних матеріалів.
5. Додаток має бути доступним для завантаження та використання на різних версіях ОС Android.
6. При розробці додатку має бути використано кращі практики програмування та забезпечено належну документацію для подальшого розвитку та підтримки додатку.

Іконка додатку. Іконка мобільного додатку це важлива складова його візуальної ідентичності, яка привертає увагу користувачів і відрізняє його від інших додатків на екрані смартфона. Відображається на головному екрані, в додатках та у списку встановлених програм. Нижче прикладена іконка мобільного додатку StandWithUkraine (рис.3.1).

Рисунок 3.1 – Іконка мобільного додатку

Інтерфейс користувача. Після того, як користувач натиснув на іконку, тим самим відкривши додаток, він перенаправляється на головний екран (рис. 3.2), де він може обрати мову інтерфейсу.

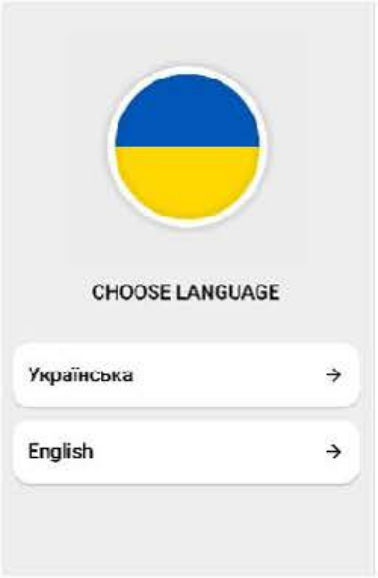

Рисунок 3.2 – Сторінка вибору мови

Потім користувач потрапляє на сторінку з головним посилом, що описує структуру та можливості додаток (рис. 3.3).

Рисунок 3.3 – Сторінка головного посилу мобільного додатку

Після цього користувач має можливість вибрати режим роботи додатку (рис. 3.4).

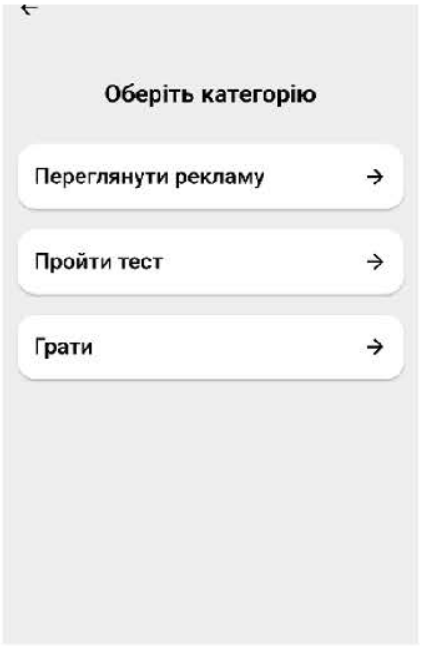

Рисунок 3.4 – Сторінка вибору категорій додатку

– Тестування з української мови;
– Гру «Fluppy Bird»;
– Перегляд реклами.

Якщо користувач обрав тестування, відкривається список питань, які користувач може вирішити (рис.3.5).

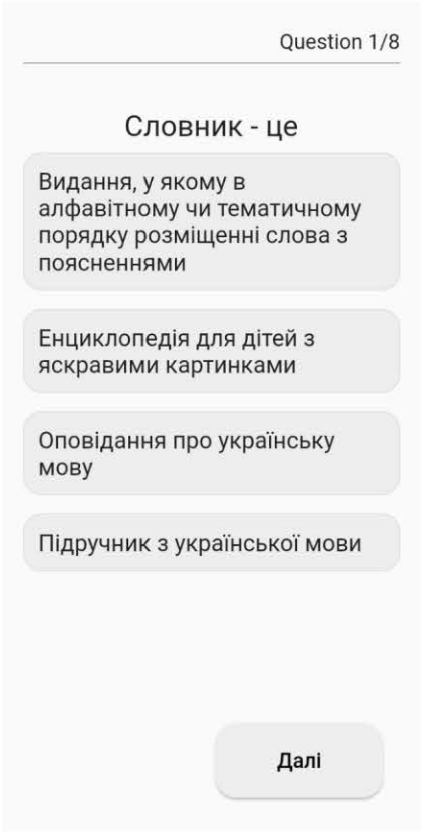

Рисунок 3.5 – Сторінка тесту з української мови

Якщо користувач обрав простий перегляд реклами, йому показується список доступних рекламних відео та інших рекламних інтеграцій, які він може переглянути (рис. 3.6):

Рисунок 3.6 – Відображення рекламного блоку

При виборі, відкривається гра, вмонтована в додаток «Fluppy Bird» (рис.3.7):

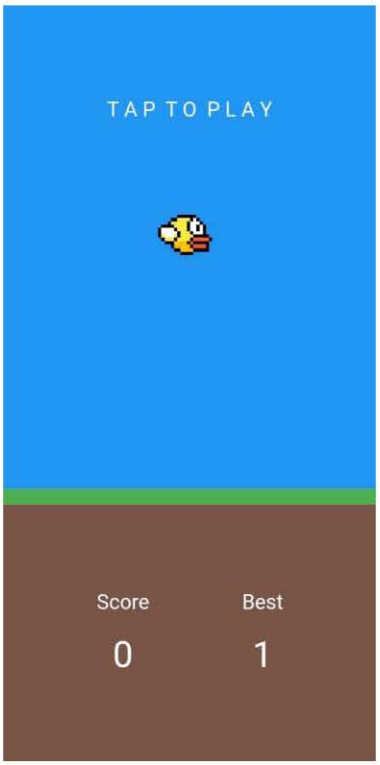

Рисунок 3.7 – Сторінка гри

3.2 Проектування та програмна реалізація мобільного додатку

Всі функції додатку доступні для користувачів безкоштовно та зручно працюють на ОС Android. Функціональні можливості користувача відображені на рисунку 3.8.

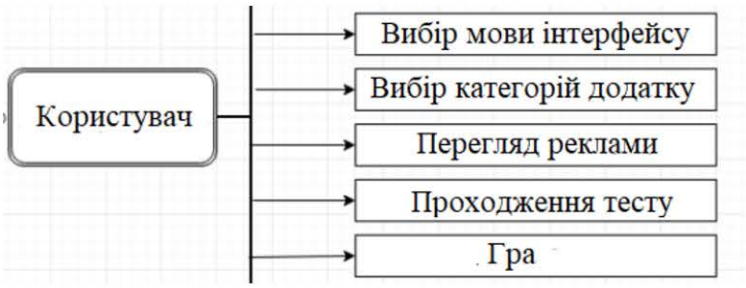

Рисунок 3.8 – Функціональні можливості користувача

Наступним етапом після визначення функціональних можливостей є розробка системи першої сторінки з вибором зручної мови інтерфейсу для користувача.

Для цього були створені файли localization.dart та AppLocalizationsDelegate, які відповідають за локалізацію додатку. В localization.dart маємо словники з рядками на англійській та українській мовах.

Для розробки першого екрану з вибором мови інтерфейсу (рис.3.9), саме української та англійської мови, будемо використовувати наступні методи:

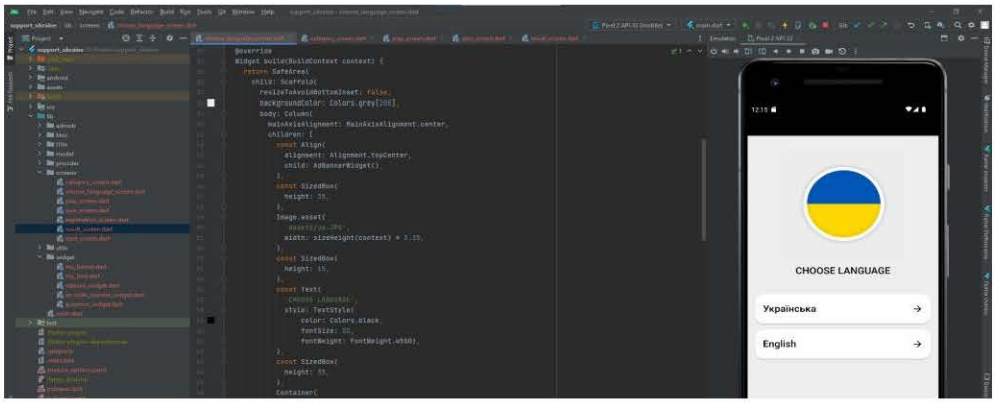

Рисунок 3.9 – Проектування першого екрану з вибором мови інтерфейсу

1. Створення файлу localization.dart для локалізації мови інтерфейсу додатку:

```
import 'package:flutter/material.dart';
import 'package:flutter/foundation.dart' show SynchronousFuture;
import 'package:flutter/widgets.dart';
class AppLocalizations {
 final Locale locale;
 AppLocalizations(this.locale);
 static AppLocalizations of(BuildContext context) {
  return Localizations.of<AppLocalizations>(context, AppLocalizations);
 }
 static Map<String, Map<String, String>> _localizedValues = {
  'en': {
   'buttonText': 'English',
  },
  'uk': {
   'buttonText': 'Українська',
  },
 };
 String get title {
  return _localizedValues[locale.languageCode]['title'];
 }
 String get buttonText {
  return _localizedValues[locale.languageCode]['buttonText'];
 }
}
class AppLocalizationsDelegate extends LocalizationsDelegate<AppLocalizations> {
 const AppLocalizationsDelegate();
 @override
 bool isSupported(Locale locale) {
  return ['en', 'uk'].contains(locale.languageCode);
 }
 @override
 Future<AppLocalizations> load(Locale locale) {
  return SynchronousFuture<AppLocalizations>(AppLocalizations(locale));
 }
```

```
  @override
  bool shouldReload(AppLocalizationsDelegate old) {
    return false;
  }
}
```

2. Підключення для використання локалізації в мобільному додатку:

```
import 'package:flutter/material.dart';
import 'localization.dart';
void main() {
  runApp(MyApp());
}
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Localizations Demo',
      localizationsDelegates: [
        AppLocalizationsDelegate(),
        GlobalMaterialLocalizations.delegate,
        GlobalWidgetsLocalizations.delegate,
      ],
      supportedLocales: [
        const Locale('en', ''),
        const Locale('uk', ''),
      ],
      home: HomePage(),
    );
  }
}
class HomePage extends StatelessWidget {
  @override
```

```
  Widget build(BuildContext context) {
   return Scaffold(
    appBar: AppBar(
     title: Text(AppLocalizations.of(context).title),
    ),
    body: Center(
     child: RaisedButton(
      onPressed: () {
       // Дії, що виконуються при натисканні кнопки
      },
      child: Text(AppLocalizations.of(context).buttonText),
     ),
    ),
   );
  }
}
```

В AppLocalizationsDelegate визначаємо підтримувані мови та методи для завантаження локалізації. У головному файлі додатка main.dart було використано AppLocalizationsDelegate як делегата для локалізації та встановлюємо підтримувані мови. У HomePage використовуємо локалізовані рядки, отримані з AppLocalizations, для відображення заголовку та тексту кнопки. Залежно від обраної мови пристрою, додаток буде відображати відповідні перекладені рядки.

Також, для використання цього підходу потрібно мати файли з рядками для кожної підтримуваної мови. Цей підхід дозволяє легко змінювати мову в додатку, забезпечуючи більш широку доступність та зручність для користувачів з різних країн та культур.

Для переходу на наступний екран, використовуємо наступний інструмент Navigator для переходу з однієї сторінки додатку на іншу. Ось приклад коду, який відповідає:

```
import 'package:flutter/material.dart';
void main() {
```

```
 runApp(MyApp());
}
class MyApp extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return MaterialApp(
   title: 'App',
   home: HomePage(),
  );
 }
}
class HomePage extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: Text('Home Page'),
   ),
   body: Center(
    child: RaisedButton(
     child: Text('Go to Next Page'),
     onPressed: () {
      Navigator.push(
       context,
       MaterialPageRoute(
        builder: (context) => NextPage(),
       ),
      );
     },
    ),
   ),
  );
 }
}
```

```
class NextPage extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: Text('Next Page'),
   ),
   body: Center(
    child: RaisedButton(
     child: Text('Go Back'),
     onPressed: () {
      Navigator.pop(context);
     },
    ),
   ),
  );
 }
}
```

Опис переходу між сторінками. Маємо дві сторінки: HomePage та NextPage. Коли користувач натискає кнопку "Go to Next Page" на HomePage, виконується перехід до сторінки NextPage. На сторінці NextPage є кнопка "Go Back", при натисканні на яку здійснюється повернення до попередньої сторінки (HomePage).

Код Navigator.push відповідає за перехід до нової сторінки, а Navigator.pop відповідає за повернення до попередньої сторінки. Обидва методи використовують контекст (context) для здійснення переходу.

На другому екрані (рис.3.10), у верхній частині екрану прапор України округлої форми та текст «Усі гроші від перегляду реклами йдуть на ЗСУ» та «Використовуй свій смартфон для допомоги ЗСУ», кнопка з текстом почати яка переносить користувача на наступний екран.

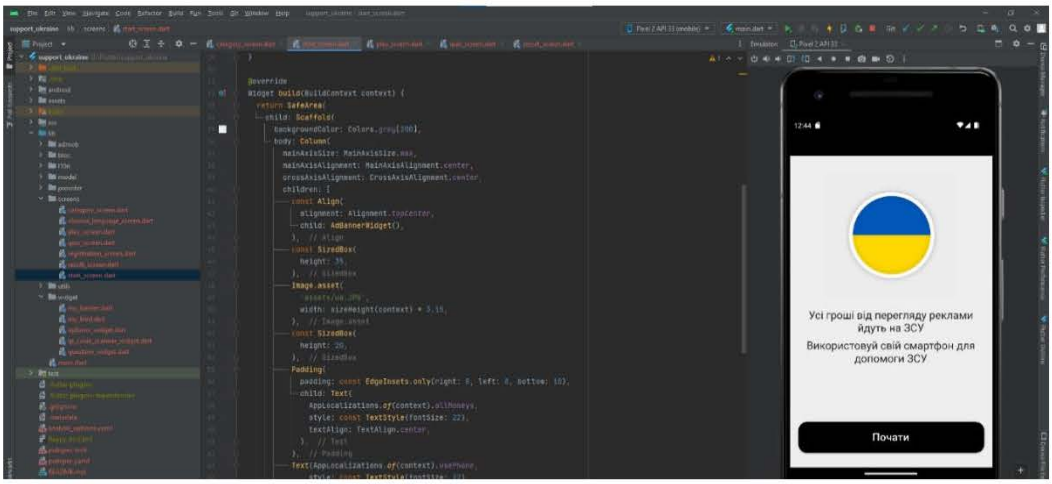

Рисунок 3.10 – Екран з головним посилом додатку

Наступним екраном являється список з вибором категорій (рис.3.11), він розроблений наступним чином:

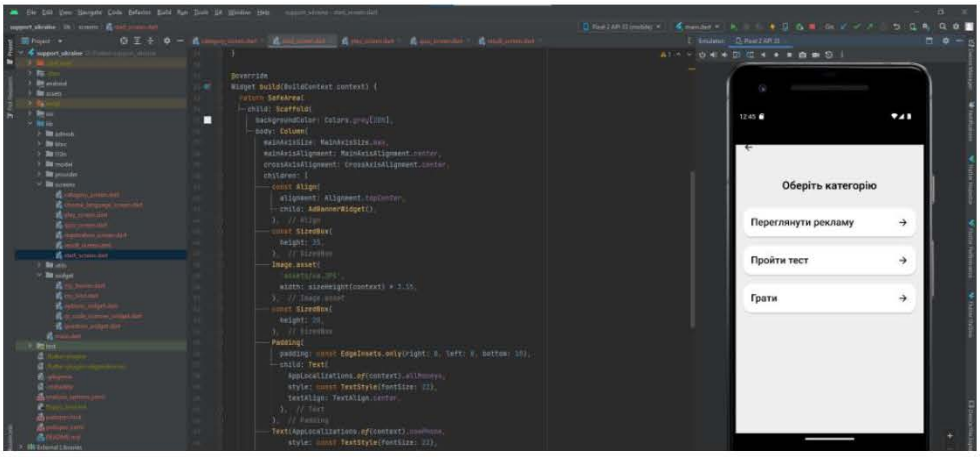

Рисунок 3.11 – Екран з вибором категорій функціоналу

Лістинг коду, який використовується для даного екрану:

```
import 'package:flutter/material.dart';
void main() {
  runApp(MyApp());
}
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'App',
```

```
      home: HomePage(),
    );
  }
}
class HomePage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Home Page'),
      ),
      body: ListView(
        children: [
          ListTile(
            title: Text('Перегляд реклами'),
            onTap: () {
              Navigator.push(
                context,
                MaterialPageRoute(
                  builder: (context) => AdScreen(),
                ),
              );
            },
          ),
          ListTile(
            title: Text('Тест'),
            onTap: () {
              Navigator.push(
                context,
                MaterialPageRoute(
                  builder: (context) => TestScreen(),
                ),
              );
            },
```

```
        ),
        ListTile(
          title: Text('Гра'),
          onTap: () {
            Navigator.push(
              context,
              MaterialPageRoute(
                builder: (context) => GameScreen(),
              ),
            );
          },
        ),
      ],
    ),
  );
 }
}
class AdScreen extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Перегляд реклами'),
    ),
    body: Center(
      child: Text('Екран перегляду реклами'),
    ),
  );
 }
}
class TestScreen extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return Scaffold(
```

```
      appBar: AppBar(
        title: Text('Тест'),
      ),
      body: Center(
        child: Text('Екран тестування'),
      ),
    );
  }
}
class GameScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Гра'),
      ),
      body: Center(
        child: Text('Екран гри'),
      ),
    );
  }
}
```

Екран списку з трьома категоріями: "Перегляд реклами", "Тест" та "Гра". Кожен пункт списку реагує на натискання та відкриває відповідний екран.

Клас HomePage є наслідником StatelessWidget і відображає ListView з трьома ListTile, які представляють категорії. Кожен ListTile має назву категорії та відповідний обробник події onTap, метод Navigator.push для переходу на відповідний екран.

Кожен екран (AdScreen, TestScreen, GameScreen) також є наслідником StatelessWidget і відображає Scaffold з заголовком в AppBar та вмістом в body. У

прикладі, для кожного екрану просто відображається текст, щоб продемонструвати різні екрани.

При натисканні на категорію "Перегляд реклами", "Тест" або "Гра", відповідний екран відкривається за допомогою методу Navigator.push, який отримує поточний контекст та MaterialPageRoute, що будує новий екран.

Розробка відображення рекламних блоків при натиску на кнопку «Переглянути рекламу»:

Фреймворк Flutter пов'язаний зі сервісом рекламної мережі AdMob, щоб відображати рекламні блоки при натисканні на кнопку «Переглянути рекламу».

Додали залежність: в файлі pubspec.yaml додано залежність firebase_admob для підключення до сервісу AdMob:

```
dependencies:
 flutter:
  sdk: flutter
 firebase_admob: ^2.0.0
```

Імпортували пакети: в файлі Dart, який містить клас-віджет, імпортовано необхідні пакети:

```
import 'package:flutter/material.dart';
import 'package:firebase_admob/firebase_admob.dart';
```

При натиску користувачем на додану кнопку у головному меню «Переглянути рекламу», подається запит на сервер AdMob, звідки приходить доступний рекламний блок та відображається на екрані (рис.3.12):

```
class MyHomePage extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: Text('My App'),
   ),
   body: Center(
    child: RaisedButton(
     onPressed: () {
```

```
            // Виклик функції показу реклами при натисканні кнопки
            showAd();
          },
          child: Text('Переглянути рекламу'),
        ),
      ),
    );
  }
}
```

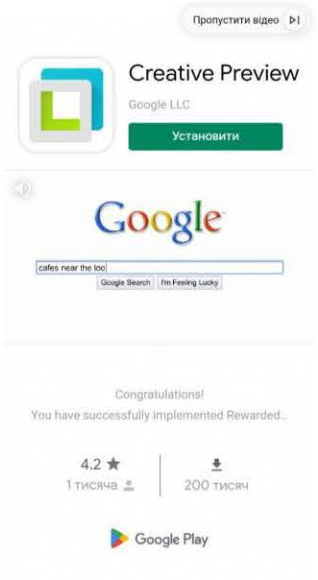

Рисунок 3.12 – Показ рекламного блоку, який відкривається після натиску на кнопку «Переглянути рекламу»

Ініціалізація та показ реклами: створено файл зі стартовим віджетом додатку, додані унікальні ідентифікатори додатку, які призначаються в аккаунті AdMob та відповідним кодом для ініціалізації та показу рекламних блоків:

```
void main() {
  // Ініціалізація AdMob
  FirebaseAdMob.instance.initialize(appId: 'pub-9176260107574133~6700231910/);
  runApp(MyApp());
}
class MyApp extends StatelessWidget {
  @override
```

```
  Widget build(BuildContext context) {
   return MaterialApp(
     title: 'My App',
     theme: ThemeData(
       primarySwatch: Colors.blue,
     ),
     home: MyHomePage(),
   );
  }
}
void showAd() {
  // Створення рекламного банера
  BannerAd bannerAd = BannerAd(
    adUnitId: ‘pub-9176260107574133/9198364756’,
    size: AdSize.banner,
    targetingInfo: MobileAdTargetingInfo(),
  );
  // Завантаження та показ рекламного банера
  bannerAd
     .load()
     .then((value) => bannerAd.show(anchorType: AnchorType.bottom));
  // Прибирання рекламного банера після певного часу
  Future.delayed(Duration
```

Таким чином розроблено запит до AdMob, рекламні блоки від якого приходять у додаток та відображаються на сторінці користувача.

Створення тесту з української мови (рис.3.13) реалізовано наступним чином:

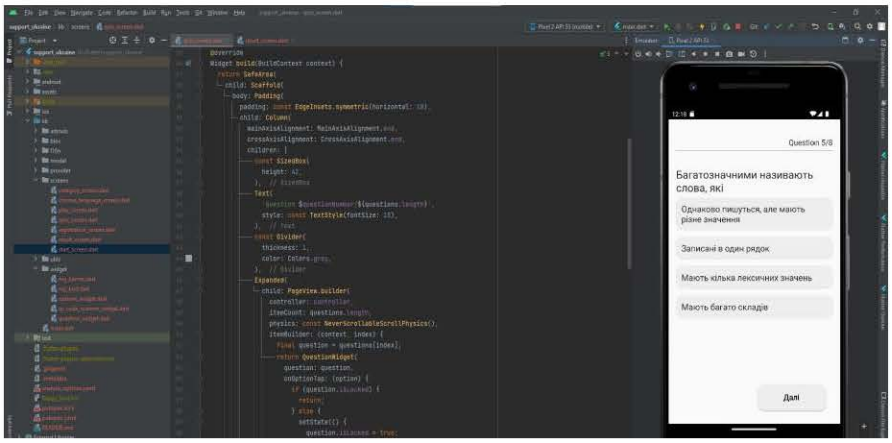

Рисунок 3.13 – Проходження тестування з української мови

Для створення тесту з українською мовою у мобільному додатку за допомогою фреймворку Flutter і мови Dart, створено відповідну модель для питань та відповідей, а також інтерфейс користувача для відображення тесту.

Код, який демонструє створення тесту з українською мовою:

```
import 'package:flutter/material.dart';
class Question {
 final String questionText;
 final List<String> options;
 final int correctAnswerIndex;
      Question({required this.questionText, required this.options, required this.correctAnswerIndex});
}
 @override
 State<QuizScreen> createState() => _QuizScreenState();
}
class _QuizScreenState extends State<QuizScreen> {
 late PageController controller;
 int questionNumber = 1;
 int score = 0;
 @override
 void initState() {
  super.initState();
  controller = PageController(initialPage: 0);
```

```
}
@override
Widget build(BuildContext context) {
 return SafeArea(
  child: Scaffold(
   body: Padding(
    padding: const EdgeInsets.symmetric(horizontal: 18),
    child: Column(
     mainAxisAlignment: MainAxisAlignment.end,
     crossAxisAlignment: CrossAxisAlignment.end,
     children: [
      const SizedBox(
       height: 42,
      ),
      Text(
       'Question $questionNumber/${questions.length}',
       style: const TextStyle(fontSize: 18),
      ),
      const Divider(
       thickness: 1,
       color: Colors.grey,
      ),
      Expanded(
       child: PageView.builder(
        controller: controller,
        itemCount: questions.length,
        physics: const NeverScrollableScrollPhysics(),
        itemBuilder: (context, index) {
         final question = questions[index];
         return QuestionWidget(
          question: question,
          onOptionTap: (option) {
           if (question.isLocked) {
            return;
```

```
            } else {
              setState(() {
                question.isLocked = true;
                question.selectedOption = option;
              });
              if (question.selectedOption!.isCorrect) {
                score++;
              }
            }
          },
        );
      },
    )
```

В даний код було додано створення моделі Question для представлення питань та відповідей. Кожне питання містить текст питання (questionText), список варіантів відповідей (options) та індекс правильної відповіді (correctAnswerIndex).

Далі, був створений становий віджет QuizScreen, який розширює StatefulWidget. У цьому віджеті визначаються змінні currentQuestionIndex і score для відстеження поточного питання та результатів тестування.

Наступним етапом було створено список питань (questions), де кожне питання є екземпляром моделі Question. Можливо додати більше питань до списку, просто додавши нові об'єкти Question з відповідними даними.

У методі checkAnswer перевіряємо, чи правильно була обрана відповідь. Якщо відповідь правильна, строка підсвічується зеленим кольором. Потім перевіряємо, чи є ще питання у списку. Якщо так, переходимо до наступного питання шляхом збільшення currentQuestionIndex.

У методі build будуємо інтерфейс користувача з використанням різних віджетів, таких як Text, Column, ListTile та інші. Відображаємо номер поточного питання, текст питання та список варіантів відповідей. При натисканні на варіант

відповіді викликаємо метод checkAnswer, передаючи йому індекс обраного варіанту.

Створення гри (рис.3.14).

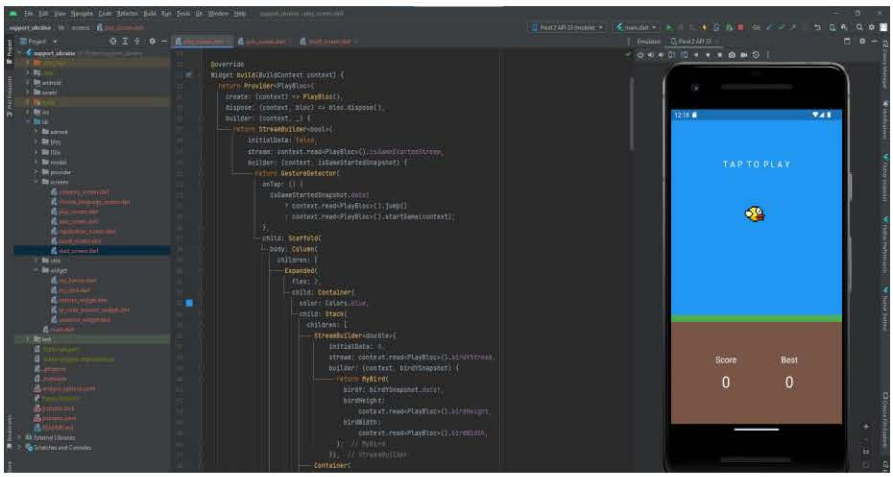

Рисунок 3.14 – Початок гри, після натиску у головному виборі категорії «Гра»

Щоб створити мобільну гру "Flappy Bird" у мобільному додатку за допомогою фреймворку Flutter і мови Dart, знадобиться відповідний віджет та деякі функції для управління гравцем та логікою гри.

Приклад коду, який демонструє створення гри "Flappy Bird":

```
import 'package:flutter/material.dart';
import 'dart:async';
void main() => runApp(FlappyBirdApp());
class FlappyBirdApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flappy Bird',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: FlappyBirdGame(),
    );
  }
```

```
}
class FlappyBirdGame extends StatefulWidget {
 @override
 _FlappyBirdGameState createState() => _FlappyBirdGameState();
}
class _FlappyBirdGameState extends State<FlappyBirdGame> {
 double birdYAxis = 0;
 double time = 0;
 double height = 0;
 double initialHeight = 0;
 bool gameHasStarted = false;
 void jump() {
  setState(() {
   time = 0;
   initialHeight = birdYAxis;
  });
 }
 void startGame() {
  gameHasStarted = true;
  Timer.periodic(Duration(milliseconds: 60), (timer) {
   time += 0.04;
   height = -4.9 * time * time + 2.8 * time;
   setState(() {
    birdYAxis = initialHeight - height;
   });
   if (birdYAxis > 1) {
    timer.cancel();
    gameHasStarted = false;
   }
  });
 }
 @override
 Widget build(BuildContext context) {
  return GestureDetector(
```

```
onTap: () {
 if (gameHasStarted) {
  jump();
 } else {
  startGame();
 }
},
child: Scaffold(
 body: Column(
  children: [
   Expanded(
    flex: 3,
    child: Container(
     color: Colors.blue,
     child: Center(
      child: Text(
       'TAP TO JUMP',
       style: TextStyle(
        fontSize: 20,
        fontWeight: FontWeight.bold,
        color: Colors.white,
       ),
      ),
     ),
    ),
   ),
   Expanded(
    flex: 6,
    child: Container(
     color: Colors.green,
     child: Center(
      child: Container(
       width: 60,
       height: 60,
```

```
                decoration: BoxDecoration(
                  image: DecorationImage(
                    image: AssetImage('assets/bird.png'),
                  ),
                ),
              ),
            ),
          ),
        ),
        Expanded(
          flex: 1,
          child: Container(
            color: Colors.brown,
          ),
        ),
      ],
    ),
  ),
 );
 }
}
```

Створюємо клас FlappyBirdApp, який реалізує MaterialApp і встановлює головний віджет FlappyBirdGame як домашню сторінку додатку.

Визначаємо становий віджет _FlappyBirdGameState, який розширює StatefulWidget. У цьому віджеті оголошуємо різні змінні, такі як birdYAxis (положення пташки по вертикалі), time (час гри), height (висота пташки), initialHeight (початкова висота пташки) та gameHasStarted (прапорець, що вказує, чи розпочалась гра).

У методі jump встановлюємо значення time на 0 і зберігаємо початкову висоту пташки. Це дозволяє пташці здійснити стрибок.

Метод startGame запускає гру. Використовуючи Timer.periodic, оновлюємо положення пташки в залежності від часу. Формула -4.9 * time * time + 2.8 * time визначає шлях руху пташки за законами фізики. Постійно оновлюючи стан, переміщаємо пташку вгору і вниз. Якщо пташка опускається нижче дна, гра закінчується.

У методі build створюємо інтерфейс гри, використовуючи різні віджети, такі як GestureDetector, Scaffold, Container, Text і Image. За допомогою GestureDetector дозволяємо користувачеві взаємодіяти з екраном, натискати для стрибка пташки або розпочати гру. Віджет Text відображає текст "TAP TO JUMP", а віджет Image відображає зображення пташки.

3.3 Робота користувача додатку

Притримуючись шаблону головної сторінки та принципів дизайну, була розроблена головна сторінка системи. Всі сторінки містять зручне меню навігації зверху сторінки, яке дозволяє перейти до деяких розділів системи:

1) вибір мови додатка;

2) екран головної мети мобільного застосунку;

3) вибір категорій;

4) перегляд гри;

5) перегляд реклами;

6) проходження тестування.

Початкова сторінка додатку при його відкритті (рис. 3.2).

Після натиску на кнопку «Почати» користувач потрапляє на наступний екран з вибором категорій (рис. 3.3).

Після натиску на вибір мови застосунку, користувач переходить на інший екран мобільного додатку, з головним посилом додатку (рис.3.4):

Таким чином, користувач може обирати між трьома категоріями функцій додатку, а саме «Перегляд реклами» (рис.3.6), при натиску на який користувачу відкривається можливість перегляду рекламних оголошень. При натисканні на

кнопку «Пройти тест» (рис.3.5), користувач може обрати та пройти тест на орфографічні знання або знання української мови. При натиску на кнопку «Грати», користувачу відкривається гра «Fluppy Bird» (рис.3.7):

Flappy Bird (Fluppy Bird) – це мобільна гра, в якій гравці клікають по екрану, щоб керувати птахом і проходити через серію труб, уникнувши зіткнення. Простота та складність гри призвели до її масштабної популярністю.

Гра має просте управління, але вимагає від гравців високої реакції та точності, що робить її дуже викликом. Ця гра стала легендою серед мобільних ігор і зберігає свою популярність досі.

Тестування з української мови (рис.3.5) може бути корисним для визначення рівня знань, оцінки потреб у покращенні мовної компетенції та допомоги в підвищенні якості комунікації з україномовними співрозмовниками.

Розроблений мобільний додаток відповідає всім сучасним стандартам розробки додатків, реалізує та відповідає шаблону проектування MVC, має відповідний до стандартів, визначених в дизайн, надає зручні можливості для роботи користувачів, та реалізує всі необхідні функції.

Повний лістинг розробленого мобільного застосунку наведений у додатку.

ВИСНОВКИ ДО РОЗДІЛУ 3

У розділі "Моделювання предметної області" було розглянуто основні функції та вимоги до мобільного додатку а також було розроблено сценарії взаємодії користувачів з додатком.

На основі аналізу предметної області було визначено, що додаток має бути зручним у використанні. Також важливо було врахувати можливість додавання реклами, зароблені кошти з якої будуть перераховані на підтримку України у важкий час.

ВИСНОВКИ

У результаті роботи було розроблено мобільний додаток, який був написаний на мові програмування Dart з фреймворком Flutter. Створений додаток, дозволяє користувачам переглядати рекламу та допомагати Україні. Не менш важлива перевага це інтуїтивно-зрозумілий та привабливий інтерфейс. Крім того, додаток також містить тести з української мови і гру, які допомагають покращити знання користувачів та розважитись, що э основною перевагою. В цілому, мобільний додаток StandWithUkraine є важливим інструментом в боротьбі за підтримку та розвиток України.

У загальному, додаток це дуже корисний інструмент для тих, хто бажає підтримати Україну, переглядаючи рекламу. Користувачі можуть з легкістю переглядати рекламу та збирати кошти на підтримку України. Крім того, наявність тестів з української мови та гри збагачує додаток і дозволяє користувачам розвивати свої знання про Україну.

Розроблений програмний продукт, в рамках виконання дипломної роботи, може знайти своє застосування в різних ситуаціях, як, наприклад, допомога переселеним жителям тимчасово окупованих територій, підтримка ЗСУ, та будь-які інші проблеми громадян України, де потрібна фінансова підтримка.

Одними з основних технічних рішень, що були запропоновані в роботі, є інтуїтивно-зрозумілий та привабливий інтерфейс, це автоматичного перегляду рекламних блоків за допомогою підключеного сервісу Google AdMob, проходження тестування та вивчення мови . Ці рішення значно допомагають українцям у важкий військовий час.

На підставі отриманих результатів можна запропонувати покращення продукту, наприклад, вдосконалення тестування, згодом додати нові можливі варіанти проходження, щоб уникнути можливих збоїв. Також, можливе додавання вивчення різних мов, задля приваблення нових користувачів. Використання цього продукту можна рекомендувати для багатьох проблем, де він може бути корисним.

Оцінюючи розроблений продукт, можна зазначити його численні переваги, і його недоліки, такі як:

Стабільність додатку під час його користування. Попри вмонтовані тести з української мови та гру, додаток працює стабільно, в випадку проходження немає підв'язки до мережі, власне саме через це можливість перебою майже неможливе, що дає максимальну зручність в користуванні.

Можливість збоїв під час відображення рекламних блоків. Це може виникнути через багато причин, наприклад: вимкнений інтернет у користувача, проблеми з мережею, недостатність блоків у Google AdMob,. При тому, що у сервісу який надає відображені рекламні блоки всередині додатку є багато можливостей налаштувань та варіацій, але можуть бути певні обмеження у показі для тієї чи іншої країни. Для вирішення цих проблем необхідно аналізувати причини та вживати відповідні заходи.

На базі отриманих висновків можна запропонувати наступні рекомендації:

1. При розробці мобільного додатку, необхідно забезпечувати максимальну стійкість до помилок, що виникають при роботі.

2. При проектуванні необхідно враховувати можливість росту та обсягів потенційних користувачів, а також запобігати збоїв у роботі показу реклами шляхом оптимізації запитів.

3. Для підвищення стійкості роботи рекомендується використовувати програми-моніторингу, які відслідковують стан, кількість користувачів та оперативно сповіщають про можливі проблеми.

4. Необхідно забезпечувати регулярне оновлення застосунку та швидке вирішення проблеми, щоб у разі виникнення помилки можна було відновити роботу з мінімальними втратами.

У результаті дослідження та розробки було успішно досягнуто поставлені завдання. Програмне забезпечення було розроблене з урахуванням вимог до привабливого та зручного користувальницького інтерфейсу. Була впроваджена необхідна функціональність.

В цілому, розробка програмного забезпечення є великим внеском у розвиток технологій та підтримку України. Цей проект показує, як технології можуть використовуватися не лише для комерційних цілей, а й для досягнення важливих суспільних мет.

ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Be with UA [Електронний ресурс] — Режим доступу: https://play.google.com/store/apps/details?id=com.chi.bewithua
2. dymka [Електронний ресурс] – Режим доступу: https://play.google.com/store/apps/details?id=com.ribsky.dymka
3. JavelinPaint [Електронний ресурс] — Режим доступу: https://play.google.com/store/apps/details?id=com.Avirise.JavelinPaint
4. Фреймворки гібридної розробки [Електронний ресурс] — Режим доступу до ресурсу: https://wezom.com.ua/ua/blog/luchshie-frejmvorki-dlja-krossplatformennoj-razrabotki-mobilnyh-prilozhenij
5. Кросплатформені фреймворки [Електронний ресурс] — Режим доступу до ресурсу: https://avada-media.ua/ua/services/kross-apps/
6. Rapid Application Development (RAD) | Definition, Steps & Full Guide [Електронний ресурс] — Режим доступу до ресурсу: https://kissflow.com/application-development/rad/rapid-application-development/
7. Інтеграція рекламних блоків [Електронний ресурс] — Режим доступу до ресурсу: https://support.google.com/analytics/answer/6183286?hl=uk
8. Google AdMob [Електронний ресурс] — Режим доступу до ресурсу: https://developers.google.com/admob?hl=ua
9. Model-View-Controller [Електронний ресурс] — Режим доступу до ресурсу: https://www.codecademy.com/article/mvc
10. Clean Architecture [Електронний ресурс] — Режим доступу до ресурсу: https://www.north-47.com/the-practical-guide-part-3-clean-architecture/
11. Reactive Programming [Електронний ресурс] — Режим доступу до ресурсу: https://www.mulesoft.com/lp/whitepaper/api/reactive-programming
12. Dart – найбільш зручна мова для веб-програмування [Електронний ресурс] – Режим доступу: https://metanit.com/dart/tutorial/1.1.php

13. Мова програмування Java [Електронний ресурс] – Режим доступу: https://java.lviv.ua/chomu-java-najpopulyarnisha-mova-programuvannya-u-sviti
14. Мова програмування Kotlin [Електронний ресурс] – Режим доступу: https://brander.ua/technologies/kotlin
15. APACHE що це? [Електронний ресурс] – Режим доступу: https://freehost.com.ua/ukr/faq/wiki/apache-chto-eto/