КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп'ютерних наук та кібернетики
Кафедра інтелектуальних програмних систем

Кваліфікаційна робота
на здобуття ступеня бакалавра

за спеціальністю

121 Інженерія програмного забезпечення: програмна інженерія

на тему:

РОЗРОБКА МОБІЛЬНОГО ВЕБ-СЕРВІСУ «ІКТА»

Виконав студент 4-го курсу
Віталій ДАЦЮК _____
(підпис)

Науковий керівник:
доцент, кандидат фіз.-мат. наук
Лариса КАТЕРИНИЧ _____
(підпис)

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

Студент _____
(підпис)

Роботу розглянуто й допущено до захисту на засіданні кафедри інтелектуальних програмних систем
«25» травня 2022 р.,

Протокол №10
Завідувач кафедри
Олександр ПРОВОТАР _____
(підпис)

Київ – 2022

РЕФЕРАТ

Обсяг роботи 35 сторінок, 10 використаних джерел, 14 рисунків, 5 таблиць.

Ключові слова: мобільний веб-сервіс, додаток, фреймворк, сервіс, JavaScript, React Native, Android, IOS, JSX, Google Cloud Platform.

Об’єктом роботи є реалізація процесу передачі потрібної інформації від хмарного сервісу(серверу) до клієнта, який в свою чергу підтримує весь запланований функціонал для обробки та правильного відображення даних.

Предметом роботи є мобільний додаток, який у собі містить авторизацію та сервіс, що написаний мовою JavaScript та за допомогою фреймворку React Native, який підтримує операційні системи Android та IOS.

Метою випускної кваліфікаційної роботи є створення мобільного веб-сервісу, який демонструє користувачу панорамну фотографію випадково згенерованої локації у завчасно обраному місті України і ставить перед користувачем завдання вгадати, де вона знаходиться.

Інструментом розробки додатку було обрано Visual Studio Code – текстовий редактор, що не забирає багато пам'яті. Завдяки широкому вибору плагінів та високій швидкості роботи, дане середовище повністю задоволняє поставлені задачі.

Результати роботи: при виконанні випускної кваліфікаційної роботи було розглянуто різні підходи до проєктування та реалізації мобільних застосунків, проаналізовано додатки аналоги та сервіси, схожі за ідеєю. У ході розробки роботи було реалізовано практичну частину: мобільний веб-сервіс.

Програмну реалізацію продукту можна розширювати та використовувати у майбутньому, додаючи різні типи рівнів, міст та функціонал. Додаток може бути корисним широкому спектру користувачів і використаним як у різних навчальних програмах, так і для проведення вільного часу.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

HTTP – протокол передачі даних, що використовується в комп'ютерних мережах.

HTTPS – покращена версія протоколу HTTP, що підтримує шифрування і є більш безпечною.

ООП – об'єктно-орієнтоване програмування.

JS – мова програмування JavaScript.

JSX – розширення React до синтаксису мови JavaScript.

GCP – Google Cloud Platform – хмарний веб-сервіс, що надає велику кількість функціоналу, зокрема надає доступ до панорамних фотографій певних локацій.

RN – фреймворк React Native.

API – прикладний програмний інтерфейс або протокол, що надає можливість взаємодії зі сторонніми сервісами для створення програмного забезпечення.

ОС – операційна система.

MVC – Model-View-Controller – архітектурний шаблон, який використовують для проєктування та імплементації програмного забезпечення.

SDK – набір інструментів, який дозволяє створювати прикладні програми для певної технології або платформи.

IDE – інтегроване середовище розробки, що дозволяє розробляти програмне забезпечення.

ВСТУП

Оцінка сучасного стану об’єкта розробки. У 21 столітті людство знаходиться у стані науково-технічної революції, в якості матеріальної основи якої служить електронно-обчислювальна техніка. На базі цієї техніки з'являється новий вид технологій − інформаційні. До них належать процеси де «вихідним матеріалом» і «продукцією» є інформація. У сучасній прогресуючій системі світу, наявність переносного мобільного пристрою є якщо і не обов’язковим засобом для ведення нормального життя, то як мінімум дуже і дуже бажаним. Зважаючи на те становище, в якому людство знаходиться останніх п’ять років – пандемія Covid-19 та війна – можливість звернутися в службу підтримки одразу через телефон чи придбати або замовити продукти онлайн у мережі, щоб їх доставили тобі додому значно полегшує або навіть рятує життя багатьом людям кожного дня. Саме тому мобільні пристрої є такими популярними, що ними користується майже кожен. Але окрім певних службових цілей, у яких нам допомагає наш смартфон, телефон також може повністю вирішити проблему з інформаційними та розважальними потребами людини, використовуючи різні інтернет-сервіси та застосунки. Я вирішив зупинитися якраз на інформаційно-розважальному мобільному застосунку, який допоможе людям дізнатися більше про своє рідне місто або про місто у нестандартній, цікавій формі.

Актуальність роботи та підстави для її виконання. Мобільні додатки стали одним з головних трендів у розвитку інформаційних технологій в останні роки. Основною перевагою мобільних додатків є те, що користувач може отримати доступ до них, перебуваючи в будь-якому місці та в будь-якій ситуації. Варто також відмітити, що у сучасної людини є досить актуальна проблема, з якою вона зіштовхується майже кожного дня: достатньо часто знаходяться короткі проміжки часу, у які вона вільна від роботи і просто витрачає час не продуктивно: як то поїздка з роботи у метро або ж тривале очікування в черзі куди-небудь. Задля того, щоб надати деяку користь саме таким проміжкам часу у мене і виникнула ідея

розробити додаток ІКТА, що дозволить людині, яка має при собі мобільний пристрій, провести цей час більш продуктивно та врятує її від нудьги.

Мета й завдання роботи. Беручи до уваги сучасні тенденції та розвиток технологій, метою моєї бакалавратської кваліфікаційної роботи є створення та розробка мобільного сервісу, який, використовуючи функцію Google Street View, демонструє користувачеві панорамну картинку певної локації у місті: нехай то вулиця, провулок або шосе. Це ставить перед користувачем задачу вгадати, що це конкретно за місце та, відмітивши його на карті Google Maps, дізнатися, наскільки далеким чи близьким є здогадка користувача.

Відповідно до мети ставляться такі завдання:

1. Дослідити існуючі засоби та технології для розробки мобільного додатку
2. Аналіз архітектурних шаблонів для розробки мобільного додатку.
3. Розробити структурну організацю мобільного додатку
4. Розробити алгоритм роботи мобільного додатку
5. Виконати програмну реалізацію мобільного додатку
6. Провести тестування програми та проаналізувати отримані результати

Об’єкт, методи й засоби дослідження та розроблення. Розробці практичної частини передувало ознайомлення із створенням мобільних додатків та концепціями створення сервісів для взаємодії із сервісами Google Maps та допоміжними, їх структуризації та аналізу додаткових допоміжних бібліотек та служб.

На початку розробки мобільного додатку враховувалися сучасні тенденції та рекомендації, які можна знайти на веб-сайтах для розробників. Зокрема, опрацьовано статті, в яких зроблена оцінка щодо популярності та доцільності використання тої чи іншої системи для взаємодії мобільного додатку з іншим сервісом, що забезпечував би повний функціонал.

Об'єктом дослідження є створення мобільного інформаційно-розважального сервісу, де користувачеві потрібно вгадати розташування певної локації чи місця, помітивши його на карті Google Maps.

Методами дослідження є побудова клієнт-серверних додатків, UML-діаграми, методи ООП, теорія проєктного менеджменту.

Інструментами для створення мобільного додатку було обрано мову програмування JavaScript, кросплатформенний фреймворк React Native. Для передачі інформації між клієнтом і сервером було використано бібліотеку axios.

Можливі сфери застосування. «ІКТА» — це мобільний додаток про географічні дослідження. Він буде цікавий тим, хто любить дізнаватися про нові місця, змагатися та навчатися одночасно. Ви потрапляєте у якесь невідоме місце вашого міста і маєте якнайближче здогадатись, де саме Ви знаходитеся.

Цей додаток підійде як і дорослим, так і дітям. Головний плюс цього застосунку – він не потребує багато часу. Це означає, що користуватись «ІКТА» можна майже у всіх сферах буденного життя: під час перерви на обід, у школі як навчаючу програму або ж як частиною змагального процесу на вечірці географів. Додаток призначений для того, щоб закривати інформаційні і розважальні потреби людини, тому сфери застосування цього додатку майже нескінченні.

РОЗДІЛ 1. ОБГРУНТУВАННЯ ДОЦІЛЬНОСТІ РОЗРОБКИ МОБІЛЬНОГО ВЕБ-СЕРВІСУ «ІКТА»

1.1 Аналіз предметної галузі розробки інформаційно-розважального мобільного додатку.

У сучасної людини, яка жива в багатофункціональному, прогресуючому та активному світі є безліч усіляких проблем та завдань, які вона вирішує кожного дня. Сьогодні, такі речі як тайм-менеджмент або продуктивність не є чимось екзотичним, а навпаки – вони здають все більш і більш розповсюдженими. Ці два фактори допомагають не тільки триматися на плаву, розвиватися як особистість та покращувати свій рівень життя, але й дають змогу на гідний відпочинок та рятують людей від того, щоб вони випадково не з'їхали з глузду у такі активні, насичені часи. Тому питання тайм-менеджменту, продуктивності та відпочинку для людини 21-сторіччя є нагальним.

Також, не слід забувати, що наразі усе намагається глобалізуватись та урбанізуватись, тому у міста приїжджає все більше і більше людей, вулиці переповнені, а транспорт у години час-пік взагалі майже не рухається. І хоч зараз уся світова спільнота тягнеться до того, щоб перенести усі бізнес-процеси, роботу та завдання до мережі інтернет, автоматизувати її, дистанціюватися від неї, щоб можна було керувати бізнесом не встаючи зі свого дивану, але прогрес цей не настільки швидкий, тому проблема перенасиченості міст людьми та автомобільним транспортом досі відкрита.

Все це призводить до того, що люди проводять багато часу, витрачаючи його на якісь побічні речі: довга дорога на роботу і назад, тривале очікування в усіляких чергах, простій у пробках на дорозі або ж просто обідня перерва. Усе це той час, який персона неодмінно витрачає кожного дня, зменшуючи свою продуктивність. У додачу до всього, ці проміжки часу хоч і здаються короткими, але для людини вони можуть тягнутися ну дуже довго.

Зважаючи на перелічені раніше проблеми та обставини, виникає ідея

створення певних мобільних застосунків, які ще прийнято в певних соціальних колах називати *таймкіллерами*.

1.2 Огляд схожих за ідеєю мобільних застосунків та веб-сервісів аналогів.

Як зазначалося вище у мене, як у людини яка зіштовхується майже з усіма уже наведеними проблемами кожного дня, виникла ідея розробити щось таке, що дозволить проводити ці самі короткі проміжки часу з користю. Тому я вирішив створити інформаційно-розважальний додаток, що також можна назвати достатньо поширеним сучасним словом *таймкіллер*.

Таймкіллер – (з англійської time – час та killer – вбивця. Дослівно – убивця часу) – заняття (або у випадку програмної індустрії – додаток), який дозволяє або провести неминуче втрачений час з певною користю, або додаток, що значно зменшує «рівень нудьги» та хоч якось прикрашає проведений період життя, що також можна назвати деяким відпочинок від рутинних справ. Але якщо слово таймкіллер зазвичай використовують до додатків, які забирають прямо кажучи левову частину життя сучасної людини, що тільки допомагає та навіть змушує людину прокрастинувати та понижувати свою працездатність і продуктивність, у моєму випадку я хотів розробити додаток, що буде, якщо так можна виразитись «вбивати час» тих проміжків життя, коли втрата цього часу не є бажаною, але є невідворотньою. Тобто для свого застосунку я хотів використати це слово саме у позитивному ключі.

На сьогоднішній час додатки-таймкіллери це не новизна. Наразі прикладів таких додатків можна приводити хоч цілий день і вони бувають на різний смак: усілякі мобільні ігри та платформери, чати та форуми, сервіси для переглядів фільмів, відео, тощо. Навіть більшість соціальних мереж наразі молодь та підлітки використовують зовсім не як засіб комунікації, поширення інформації та добування корисної інформації, а використовують їх за призначенням типового таймкіллеру. Але у цьому розділі ми будемо розглядати самі ті

додатки, що допомагають «вбити час», які дозволяють провести короткі проміжки життя саме з користю та у граючій, веселій формі. Варто зазначити, що для підбору аналогів та схожих застосунків я спирався на суб'єктивну свою думку та думку моїх друзів-однолітків. Тому я не претендую на те, щоб кожен погоджувався з твердженням про те, що додатки, які я наведу нижче є цілком інформаційно-корисними.

При дослідженні інтернет ресурсів, я знайшов декілька додатків, які схожі за ідеєю та один додаток-аналог:

Duolingo – електронна платформа, створена для вивчення мови і краудсорсингового перекладу тексту. Мають два види акаунтів: безкоштовний і платний. Структура сервісу побудована на тому, що за мірою проходження уроків по обраній мові та здобуваючи певні знання мови, користувачі також допомагають перекладати вебсайти, статті та інші документи. Процес навчання проходить в цікавій, ненав'язливій формі, що допомагає вивчати матеріал значно легше. Тому даний додаток несе не тільки персональну користь користувачеві, але й допомагає людям та організаціям зі всього світу отримувати належний переклад з однієї мови на інші. Мобільний додаток написаний на мові програмування Scala.

Рис. 1.2.1 – Приклад інтерфейсу мобільного додатку Duolingo

Quiz Planet – мобільний застосунок – змагання, яке дозволяє позмагатися з другом чи незнайомцем у знаннях стосовно тисячі різних тем та фактів. Додаток побудований у вигляді певної гри, у якому користувачеві потрібно відповісти на поставлене питання правильно, маючи 4 варіанти відповідей. Відповідаючи на три питання, які разом складають раунд, користувач чекає на результати відповідей його візаві. Відповідно той, хто відповів на більшу частину питань правильно – є переможцем. Така форма «прокачування» ерудиції в змагальній формі дозволяє запам’ятовувати цікаві факти та речі набагато легше й краще, ніж через читання певних енциклопедій. Мобільна частина застосунку написана завдяки фреймворку на базі JavaScript – React Native.

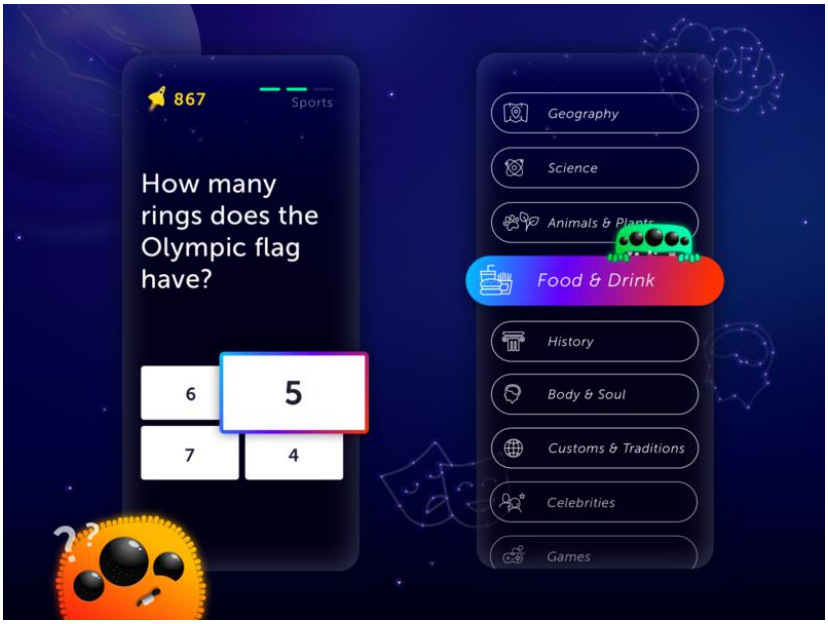

Рис. 1.2.2 – Приклад інтерфейсу мобільного додатку Quiz Planet

GeoGuessr – прямий аналог мого застосунку. Це певна вебгра, у якій використовуються напіввипадково згенеровані локації з усього світу, які можна побачити у панорамному вигляді завдяки функції Google Street View для платних підписників та Mapillary для безкоштовного використання. Гра вимагає від користувача вгадати місцезнаходження локації, що була продемонстрована, та відмітити це місце на карті. Це дозоляє дізнатися, наскільки правильним була здогадка користувача. Вебсайт отримав сотні тисяч унікальних відвідувачів лише

у перший тиждень існування гри. Мобільна версія додатку була написана пізніше та завдяки фреймворку на базі Javascript – React Native.

Рис. 1.2.3 – Приклад інтерфейсу мобільного додатку GeoGuessr

Таблиця 1.2.1 – Переваги та недоліки існуючих інформаційно – розважальних додатків

Додаток	Переваги	Недоліки
Duolingo	Простота використання. Не забирає багато часу. Зрозумілий інтерфейс.	Обмежений функціонал для безкоштовних користувачів.
Quiz Planet	Не забирає багато часу. Цікава, оригінальна форма подачі інформації.	Для гри завжди потрібен другий гравець.
Geoguessr	Багатий функціонал. Можливість створювати свої локації та рівні	Розповсюджується на платній основі. Обмежений час для безкоштовних користувачів.

1.3 Вимоги та задачі веб додатку.

У результаті аналізу, проведеного мною на основі переглянутих вище додатків, схожих за ідеєю та додатку-аналогу, я чітко зрозумів, які характерні риси повинні бути у мого мобільного веб-сервісу:

1) Зручний, зрозумілий та простий для користувача інтерфейс, з точки зору UI/UX дизайну

2) Висока швидкодія застосунку

3) Додаток повинен не забирати у користувача багато часу за один сеанс.

4) Додаток має нести інформаційну цінність для користувача.

5) Додаток має бути створений у розважальній, простій формі

6) Додаток має займати небагато пам'яті і не сильно навантажувати операційну систему девайсу

7) Додаток має бути повністю безкоштовний для користувача

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА МЕТОДИ РОЗВ'ЯЗАННЯ ЗАДАЧІ СТВОРЕННЯ МОБІЛЬНОГО ВЕБ-СЕРВІСУ

2.1 Вибір методів, що застосовуються для вирішення поставленої задачі.

Веб-додаток - розподілений додаток, в якому клієнтом виступає браузер, а сервером – веб-сервер. У випадку мобільного веб-додатку, клієнтом буде виступати не браузер, а конкретно додаток, який буде завантажений на відповідний мобільний пристрій. Цей додаток буде реалізацією так званих тонких клієнтів – логіка застосунку зосереджується на сервері, а функція мобільного застосунку полягає переважно у відображенні інформації, завантаженої мережею з сервера, і передачі назад даних користувача. Однією з переваг такого підходу є той факт, що наразі існують мови програмування та різні фреймворки, які дозволяють розробнику писати програму одразу під дві найбільш поширені операційні системи для мобільних девайсів – Android та IOS. Для написання мобільного застосунку, мною було вибрано саме такий фреймворк, тому у моєму випадку клієнти не залежать від конкретної операційної системи, яка використовується на їхньому смартфоні. Отже можна сказати, що додаток “ІКТА” є міжплатформеним сервісом.

Унаслідок цієї універсальності і відносної простоти розробки мобільні застосунки, написані на універсальних фреймворках, стали широко популярними починаючи з 2016 року. Істотною перевагою побудови такого мобільного додатку є саме те, що функції будуть виконуватися незалежно від операційної системи користувача. Це економить багато часу, ресурсів та зусиль як на етапі написання проєкту, так і у період його підтримки. Замість того, щоб писати різні версії застосунку для операційних систем Android та IOS, застосунок створюється один раз для довільно обраної платформи і на ній розгортається. Цього підходу вистачає для того, щоб написати основний функціонал застосунку та реалізувати дизайн. Проте різна реалізація вбудованих нативних елементів та різна стилізація компонентів залежно від системи може викликати проблеми при розробці мобільних застосунків і подальшої підтримки. Крім того, можливість користувача

налаштовувати багато параметрів мобільного пристрою (наприклад, розмір шрифту, кольори, відключення підтримки дозволів на геолокацію) може перешкоджати коректній роботі застосунку. Тому при розробці потрібно враховувати усі можливі сценарії та варіанти.

Мобільний клієнт формує запит і відправляє його на сервер, де у свою чергу виконуються певні обчислення. Після виконаних операцій на сервері, які можуть бути майже будь-якими: починаючи від найпростіших і закінчуючи навантаженими, довготривалими операціями, які можуть займати певну кількість часу. Після певних обчислень, сервер повертає клієнту відповідь. Цей процес «спілкування» між сервером та клієнтом відбувається завдяки протоколу відправки та отриманню повідомлень HTTP. І хоча цей протокол раніше був достатньо поширеним і популярним, сучасні додатки та веб-сервіси використовують більш захищену та покращену версію цього протоколу – HTTPS. Це розширення HTTP дозволяє підтримувати шифрування даних, в цілях безпеки та кращого збереження інформації, яка передається від клієнту до сервера і навпаки. У моєму додатку використовується саме HTTPS протокол.

На сьогоднішній день існують такі шаблони розробки веб-сервісів та мобільних веб-додатків:

1) *Клієнт-серверний шаблон*

Даний шаблон складається з двох частин: сервера та великої кількості клієнтів. Серверний компонент забезпечує службами клієнтські компоненти. Клієнти роблять запит на послуги на сервер, а він, в свою чергу, забезпечує їх тими самими послугами. Більше того, сервер продовжує «слухати» клієнтські запити. В основному даний шаблон застосовується при проєктуванні веб-додатків та мобільних застосунків (рис.2.1).

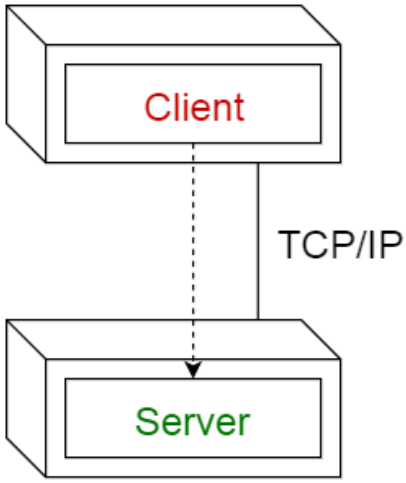

Рисунок 2.1.1 – Структурна схема клієнт-серверного шаблону проєктування

2) *Шаблон модель-представлення-контролер.*

Даний шаблон також відомий як MVC – шаблон. Основним принципом шаблону є розділення інтерактивних прикладних програм на 3 частини:

1. Модель – містить ключові дані і функціонал;
2. Представлення – відображає інформацію користувачу (можна задавати більше, ніж одне представлення);
3. Контролер – займається обробкою даних користувачів.

Це робиться з ціллю розділення внутрішнього представлення інформації від способів її представлення та прийняття від користувача. Ця схема ізолює компоненти та дозволяє ефективно реалізувати повторне використання коду. В основному даний шаблон застосовується при проєктуванні архітектури систем, написаних за допомогою основних мов програмування (рис. 2.2).

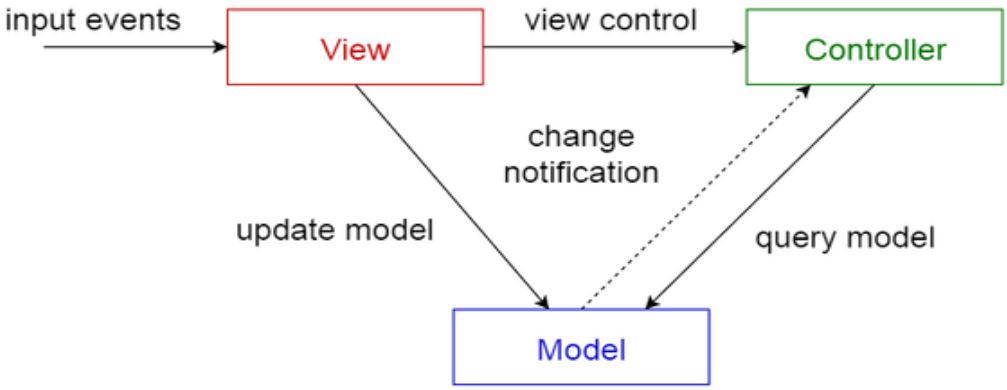

Рисунок 2.1.2 – Структурна схема шаблону «модель-представлення-контролер»

Це далеко не усі шаблони для розробки веб-сервісів та мобільних додатків, але я навів ті, які є найбільш популярними. Для свого застосунку я вирішив використати суміш цих двох підходів проєктування, які на мою думку, є найдоцільнішими та найзручнішими, для поставленої мною задачі.

2.2 Варіантний аналіз засобів розробки системи.

На даний момент часу існує чимала кількість різних фреймворків та бібліотек для розробки мобільних додатків. Більшість з них дозволяє писати застосунки для телефону, які будуть підтримувати тільки якусь одну конкретну операційну систему засобу. Для написання додатку для платформи Android зазвичай використовують мову програмування Java або Kotlin, а для написання та налагодження коду використовують Android Studio. В свою чергу для написання додатку для платформи IOS використовують мови програмування Swift або Objective-C і середовище написання коду xCode. Але такий підхід розробки мобільного додатку, коли розробник пише платформенно залежний код на різні платформи має свої переваги та недоліки:

Таблиця 2.2.1 – Переваги та недоліки написання програмного застосунку одразу під декілька мобільних операційних систем

Переваги	Недоліки
Більш чіткий та деталізований підхід до розробки додатку, що підходить більш великим та потужним застосункам	Великі затрати по ресурсам на вивчення мов, тестування застосунку та його підтримки
Гнучкість та великий спектр можливостей при написанні програмних компонентів	Неможливість відтворити ідентичну поведінку застосунку на різних платформах
Повна сумісність між старими та новими версіями операційних систем	Достатньо довга, повільна розробка застосунку

В ході аналізу вищепереглянутих переваг і недоліків я вирішив відійти від ідеї написання застосунку на двох різних мовах програмування під відповідні операційні системи, адже мій додаток не має потреби у заглибленні у такі деталі реалізації нативних компонентів. Тому, враховуючи дану інформацію, я зупинився на фреймворках, які дозволяють написання програмного коду одразу під декілька мобільних операційних систем. У моєму випадку – під системи Android та IOS, так як вони є найбільш популярними і становлять на сьогоднішній день майже 99 відсотків від усіх мобільних ОС.

2.3 Розгляд та вибір кросплатформенного фреймворку для розробки мобільного додатку.

На сьогоднішній день у великому світі мобільної розробки виникло чимало фреймворків, які дозволяють писати програмний код, не залежачи від операційної системи смартфону. У цьому розділі я проведу предметний аналіз наявних інструментів розробки та аргументую свою мотивацію зупинитися на конкретному фреймворку.

Перший фреймворк, який будемо розглядати має назву **Flutter.** Flutter – безкоштовний набір засобів розробки мобільного користувацького інтерфейсу,

створений компанією Google і випущений у травні 2017 року. Цей фреймворк є open-source(з відкритим початковим кодом) проєктом, тому це дозволяє будь-якому бажаючому розробнику вносити певні зміни або додавати новий функціонал до програмного коду інструментарію, що дає значну гнучкість та великий спектр можливостей при розробці, але за наявності достатньо широкого community(кола людей або розробників, які підтримують та розвивають проєкт). Flutter дозволяє створювати власні мобільні додатки з одним масивом коду – тобто маючи одну базу написаної програмної поведінки, на виході розробник отримує застосунок, який можна використовувати на смартфонах з операційними системами IOS та Android. На відміну від багатьох інших відомих на сьогоднішній день мобільних платформ Flutter не використовує мову JavaScript ні в якому вигляді. В якості мови програмування для цього фреймворку була використана мова програмування, що знову ж таки була створена компанією Google – *Dart*. Dart компілюється у бінарний код, за рахунок чого і досягається швидкість виконання операцій, що може зрівнятися з мовами: Objective-C, Swift, Java або Kotlin. Також варто зазначити, що Flutter не використовує нативні компоненти взагалі, що звільняє розробника від написання зайвих слоїв коду для комунікації з ними. Для побудови UI у Flutter використовується такий декларативний підхід, який дозволяє отримати ще більше приросту швидкості роботи, адже кожен віджет(компонент) перемальовується тільки при необхідності – коли у ньому дійсно щось змінилося.

Наступний фреймворк, який я розглядав для розробки свого мобільного веб-сервісу – **Xamarin**. Xamarin – зручний набір інструментів для розробки кросплатформенних мобільних додатків на мові програмування C# з використанням модульної платформи створення програмного забезпечення .NET. Цей фреймворк підтримує IOS, Android та Windows Phone. Для розробки на основі Xamarin не потрібно знати досконально специфічні мови для відповідної операційної системи. У додаток до всього, при роботі з конкретною платформою у розробника буде доступ до всіх можливостей пакета SDK(software development kit – комплект для розробки програмного забезпечення) цієї платформи, а також до вбудованих механізмів написання користувацьких інтерфейсів. Таким чином Xamarin дозволяє

створювати застосунки, які майже не мають відмінностей з нативними аналогами, а значить спокійно підходять для поширення написаного мобільного додатку через офіційні інтернет магазини цих головних платформ – Google Play (для застосунків на базі Android) та App Store (для застосунків на базі IOS). Також варто відмітити, що по словам розробників застосунок, написаний завдяки Xamarin не буде поступатися в плані продуктивності нативним інструментам розробки.

Останній фреймворк, який я розглядав у ході свого аналізу для написання мого мобільного додатку є **React Native**. React Native (також відомий у світі програмування як RN) – популярна платформа розробки мобільних додатків на основі JavaScript, яка дозволяє створювати застосунки з власним інтерфейсом для IOS та Android. Він дає змогу писати програмний код для двох платформ, використовуючи одну й ту ж саму кодову базу. React Native був вперше випущений компанією Facebook з відкритим початковим кодом у 2015 році. На базі цього фреймворку написані такі популярні мобільні застосунки як: Instagram, Facebook та Skype. Даний інструментарій був побудований на основі досить популярної JavaScript бібліотеки – React. Це дозволило зберегти розробникам основний функціонал цієї бібліотеки, перенести його та покращити вже у React Native. Це все призвело до того, що багато веб-розробників, які володіли знаннями React змогли швидко перейти до написання мобільного програмного коду і створювати надійні, готові до роботи додатки для смартфонів. Як і React, RN використовує суміш JavaScript і спеціально створеної мови розмітки JSX, який хоч і має таку саму логіку поведінки, має певні відмінності від бібліотеки. Варто зазначити, що React Native відійшов від використання популярних мов CSS та HTML, але він дозволяє використовувати власні елементи мобільного користувацького інтерфейсу.

Порівняємо згадані фреймворки:

Таблиця 2.3.1 – Переваги та недоліки існуючих кросплатформенних фреймворків, для написання мобільних застосунків

	Flutter	Xamarin	React Native
Мова програмування	Dart	C#, .NET	JavaScript + React
Підтримувані операційні системи	Android, IOS, Google Fuchsia, Web, Desktop	Android, IOS, UWP	Android, IOS, UWP
Community фреймворку	Невелике, але швидко розвивається	Велике	Дуже велике
Інструменти для написання інтерфейсу та зовнішнього вигляду	Вбудовані в фреймворк віджети	Xamarin IOS/Android та Xamarin.Forms	Нативні та Declarative UI компоненти
Продуктивність готового застосунку	Висока	Висока	Дуже висока, наближається до продуктивності нативних мов програмування

Враховуючи до уваги усі переваги та недоліки наведених вище інструментаріїв, для розробки свого мобільного веб сервісу я вирішив обрати фреймворк React Native, адже він виявився найбільш швидким у порівнянні з іншими, має дуже велику спільноту розробників та людей, які підтримують проєкт та достатньо легкий, інтуїтивний у використанні.

2.4 Розробка схеми алгоритму роботи системи.

Схему алгоритму функціонування мобільного веб-сервісу доцільно розділити на два окремих алгоритми, а саме: алгоритм функціонування клієнтської частини та поведінковий алгоритм серверної частини.

Розглянута система, як уже було зазначено в попередньому розділі, містить у собі клієнтську частину реалізовану за допомогою фреймворку React Native. Нижче представлено схему взаємодії компонентів клієнтської частини системи (рис.

2.4.1). На основі якої було розроблено алгоритм роботи клієнтської частини системи (рис. 2.4.2).

На початку роботи системи здійснюється ініціалізація системи, та рендеринг початкового зовнішнього вигляду та дизайну з початковими станами, та інформаціє. з початкової моделі даних.

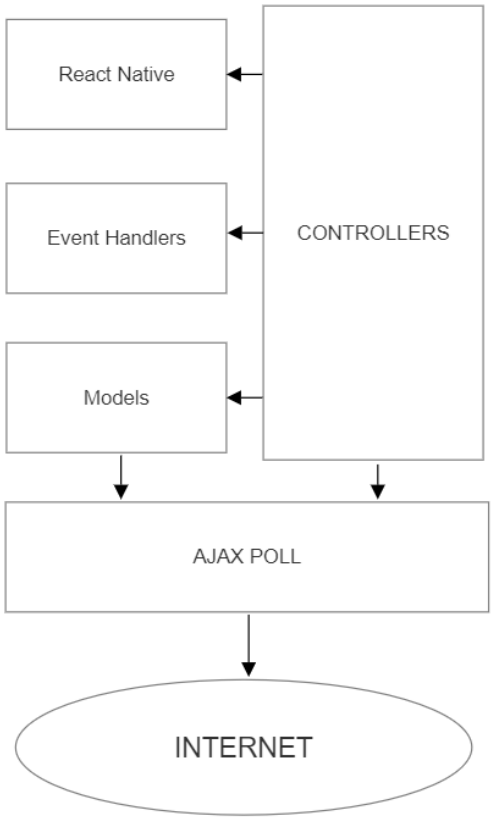

Рисунок 2.4.1 – Схема взаємодії компонентів клієнтської частини системи

Нижче наведено опис алгоритму функціонування клієнтської частини мобільного веб-сервісу:

1. Підстановка початкових даних у користувацький інтерфейс основним керуючим контролером;
2. Обробка конкретної інтеракції користувача з графічним інтерфейсом основними керуючими контролерами, з подальшою підміною даних та заміною користувацького інтерфейсу;

3. Встановлення основними контролерами початкових даних, з подальшою їх заміною на підходящі, в залежності від взаємодії користувача з користувацьким інтерфейсом

4. Звернення до сторонніх API та серверної частини мобільного веб-сервісу за допомогою ajax poll.

Згідно зі схемою взаємодії компонентів клієнтської частини системи було розроблено алгоритм роботи клієнтської частини системи.

Графічне представлення алгоритму функціонування клієнтської частини системи управління проєктами в режимі онлайн представлено на рис. 2.4.2.

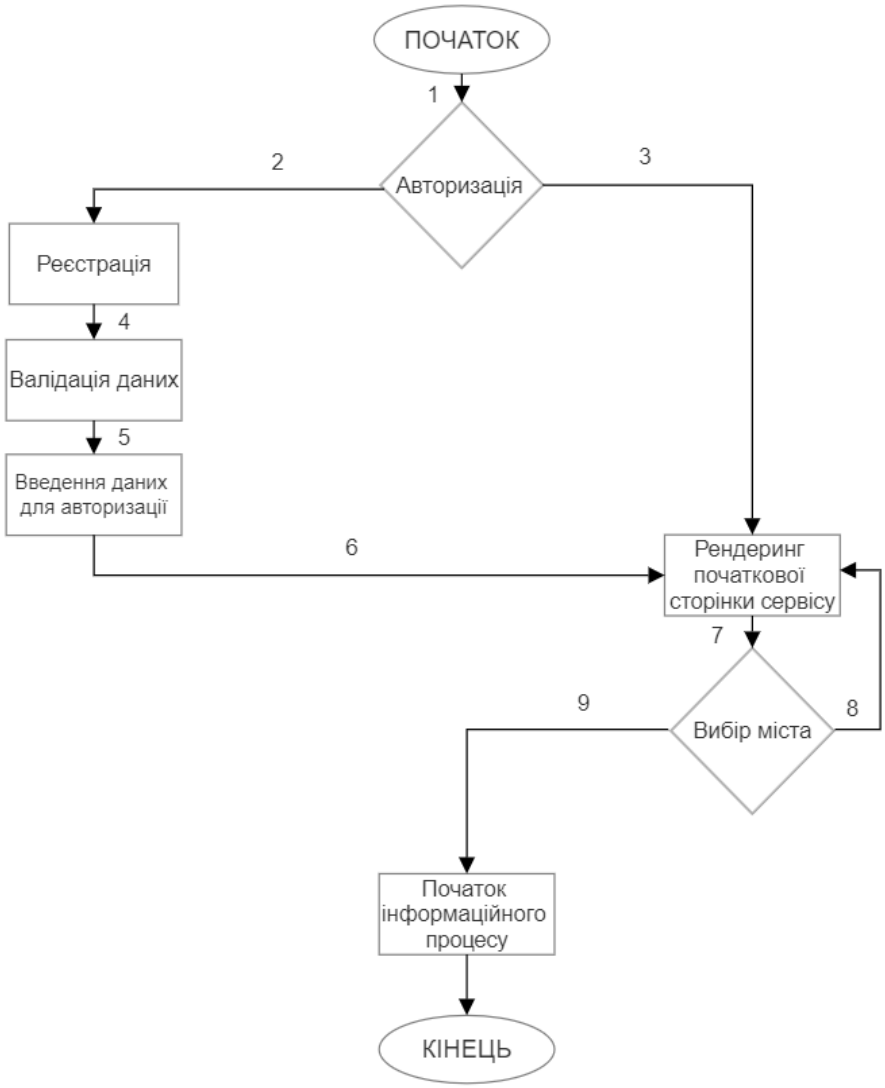

Рисунок 2.4.2 – Алгоритм функціонування клієнтської частини мобільного веб-сервісу

Алгоритм включає в себе такі етапи:

1. Завантаження авторизаційної сторінки мобільного веб-сервісу;
2. У разі, якщо користувач ще не має облікового запису – він повинен створити його, заповнивши форму для реєстрації;
3. Якщо користувач має обліковий запис і він не виходив із нього, то він одразу потрапить до етапу номер 7.
4. У разі реєстрації проводиться валідація(перевірка на дійсність) логіну та паролю користувача.
5. Наступний етап – введення даних для авторизація та також їхня подальша валідація.
6. Якщо валідація логіну та паролю пройшла успішна – відбувається авторизація користувача під введеною інформацією.
7. Завершивши процес авторизації, користувач попадає на початкову сторінку мобільного веб-сервісу, де можна переглянути дані профілю та місто, яке вибране для початку інформаційного процесу.
8. У користувача є можливість змінити обране місто, для якого будуть згенеровані панорамні зображення. Після зміни міста, користувач знову потрапить до етапу номер 7.
9. Користувач може розпочати вивчення локацій для обраного міста, натиснувши на відповідну клавішу.

У якості серверних засобів та інструментів у моєму мобільному веб сервісі було використано платформу **Google Cloud Platform**, що в свою чергу є частиною Google Cloud. Google Cloud Platform(скорочено GCP) – набір хмарних служб, які надає компанія Google і які виконуються на тій самій інфрастуктурі, які сама Google використовує для своїх же продуктів. Ці продукти призначені для своїх кінцевих споживачів, наприклад Google Search і YouTube. Окрім інструментів для керування, GCP також надає ряд таких хмарних служб, як: хмарні обчислення, зберігання даних, аналіз даних та машинне навчання.

Спектр можливостей та інструментів Google Cloud Platform досить широкий. Платформа надає доступ майже до усіх своїх функціональних досягнень та сервісів,

як то BigQuery, SQL, Cloud Functions чи App Engine. Усе це дає програмним розробникам великі можливості для створення, розширення та модифікування нового або вже готового застосунку чи додатку. Єдине, що потрібно зробити – створити акаунт розробника та зареєструвати компанію для проєкту.

Також GCP надає обмежений, але широкий доступ до свого API (програмного інтерфейсу застосунку). Це дозволяє користуватися такими API, як: Street View API, Cloud Translations API, Maps SDK API та багатьома іншими. У своєму мобільному веб-сервісі я використовував саме Street View Api та Google Maps SDK. Завдяки функціоналу Street View API я отримав можливість генерувати випадкові локації, перевіряти чи згенеровані місця існують в базі даних Google та правильно відображати панорамні фотографії локації, адже даний інтерфейс надає усю потрібну інформацію для цього. Використовуючи Google Maps SDK у мене з’явилася можливість знайти правильні межі міста, потрібні мені як розробнику для того, що нові згенеровані локації не виходили за рамки потрібної місцевості.

2.5 Узагальнення.

У даному розділі було визначено основні складові створеного мобільного веб-сервісу «ІКТА». Проведений ретельний аналіз та опис методологій для зручного підходу щодо розробки мобільних веб-сервісів, сформований перелік основних функцій застосунку.

Розроблено діаграму взаємодії основних компонентів клієнтської частини системи.

Розроблено алгоритм функціонування клієнтської частини системи.

Описаний поведінковий алгоритм взаємодії клієнтської частини з серверною платформою.

РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ МОБІЛЬНОГО ВЕБ-СЕРВІСУ «ІКТА»

3.1 Обгрунтування вибору мови програмування.

У якості мови програмування для написання програмної реалізації мобільного веб-сервісу було використано JavaScript. JS – це об’єктно-орієнтована мова програмування, але варто зазначити, що прототипування, яке використовують в даній мові обумовлене значними відмінностями в роботі з традиційними клас-орієнтованими мовами програмування. Також Javascript має ряд певних властивостей, які притаманні функціональним мовам програмування – функції, у якості об’єктів першого класу, об’єкти як списки, наявність анонімних функцій та замикань, що надають мові більшої гнучкості.

Javascript підтримує такі функціональні стилі: об’єктно-орієнтований, імперативний та функціональний.

За допомогою даної мови програмування відкриваються так можливості:

- У будь-який спосіб змінювати сторінку, писати на ній текст, додавати, змінювати або видаляти стилі,
- Можливість реагувати на певні події: скрипт може чекати, коли станеться певна подія(наприклад натискання, закінчення завантаження сторінки) і реагувати на ці події виконанням певних програмних дій;
- Можливість виконувати запити на сервер та завантажувати певні дані без перезавантаження сторінки. Цей процес називають "AJAX";
- Можливість валідувати дані, демонструвати повідомлення і багато іншого.

Хоча Javascript не позбавлена таких недоліків як, наприклад, нестрога типізація, що часто є причиною для неявного преведення типів. Однак усі мінуси перекриваються поширеністю та кросплатформенністю мови.

Головні архітектурні риси Javascript:

- прототипне програмування;
- динамічна, але слабка типізація;

- автоматичне керування пам’яттю;

Зважаючи на те, що синтаксис Javascript є схожим із синтаксисом мови C, у JS є корінні відмінності:

- функції, як об’єкти першого класу;
- автоматичне приведення типів;
- автоматизована збірка сміття;
- анонімні функції та замикання.

3.2 Обґрунтування вибору середовища розробки.

Сьогодні наявна величезна кількість середовищ, які дозволяють розробляти мобільні додатки. Їх можна розділити на дві основних категорії:

- *Текстові редактори* – достатньо прости, але швидкі інструменти для розробки. Вони не потребують багато ресурсів, які потрібні для повноцінних IDE. Цими засобами зазвичай користуються у тих випадках, коли потрібно швидко відкрити конкретний файл для внесення певних малих змін. Але на сьогоднішній день такі редактори можуть мати у своєму інструментарії велику кількість різних плагінів, надбудов та засобів, тож наразі різниця між повноцінними IDE та текстовими редакторами плавно стирається. Під мій аналіз підпали наступні редактори: Visual Studio Code, Brackets, Vim та Sublime Text. Усі вони поширюються на безкоштовній основі, і у кожному є підтримка розпізнавання синтаксису Javascript та JSX.

- *IDE (Integrated Development Environment)* – «інтегроване середовище розробки». Ці інструменти мають у собі набагато ширший функціонал, в порівнянні з редакторами тексту. Але в той самий час вони потребують більших ресурсів задля їх правильної роботи, що зумовлює значний програш у швидкості роботи. Інтегровані середовища розробки мають у собі величезний спектр додаткового функціоналу, як то наприклад дебагер, системи контролю версій та багато інших. Зазвичай IDE завантажує весь проєкт загалом, що дозволяє забезпечувати автодоповнення функціями у будь-якому місці проєкту, легку та

просту навігацію по файлам проєкту та інші. Типовими прикладами IDE для розробки є: Visual Studio, WebStorm, PHPStorm і т.д. Більшість з них розповсюджується на платній основі, але у всіх ним є також підтримка синтаксису Javascript та JSX.

Для створення свого мобільного веб-сервісу я використовував текстовий редактор Visual Studio Code. Дякуючи великій кількості додаткових плагінів різниця функціоналу між VSCode та будь-яким IDE майже відсутня. Але порівнюючи з IDE, швидкість редактору є набагато більша, він набагато гнучкіший та простіший у використанні.

3.3 Огляд інтерфейсу та тестування інформаційно-розважального мобільного веб-сервісу «ІКТА»

Для можливості використання мобільного додатку необхідні наявність персонального мобільного пристрою або схожого переносного девайсу(планшет). Також умовою для користування застосунку є наявне з'єднання з мережею інтернет.

Варто зазначити, що наведені скріншоти не є остаточним результатом дизайну, адже він може змінитися.

Для правильного тестування мобільного веб-сервісу було використано тест кейси(таблиці 3.1 та 3.2)

Таблиця 3.1 − Тест кейс №1. Вибір конкретного міста для початку інформаційного процесу

Тестовий сценарій	Вибір міста для початку гри
Кроки	Відкрити застосунок Натиснути кнопку вибір міста Вибрати місто
Очікуваний результат	Після вибору міста, інформаційний процес має розпочинатися для вибраного міста

Результати тестування тест кейсу номер 1 описані на рис. 3.1 – 3.4

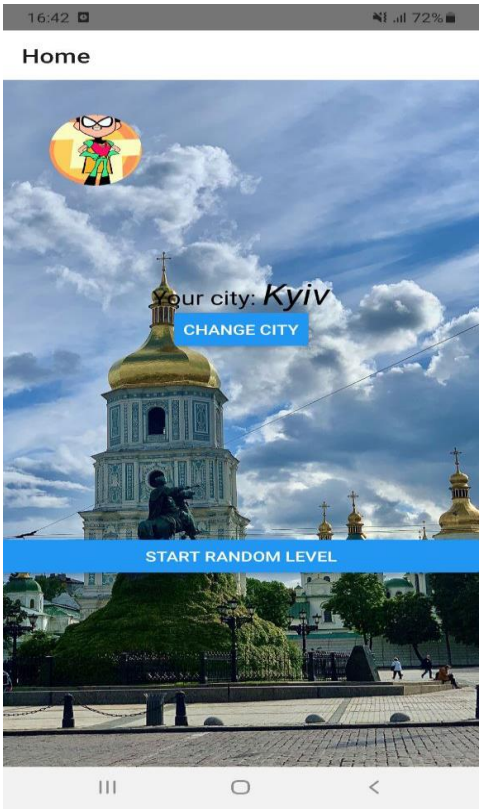

Рисунок 3.1 – Початкове вікно

Рисунок 3.2 – Вікно вибору екрану

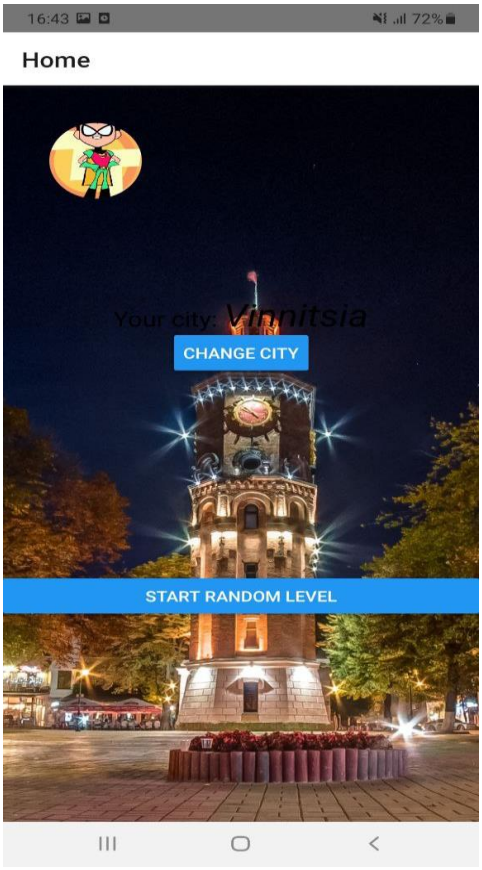

Рисунок 3.3 – Початкове вікно після вибору міста

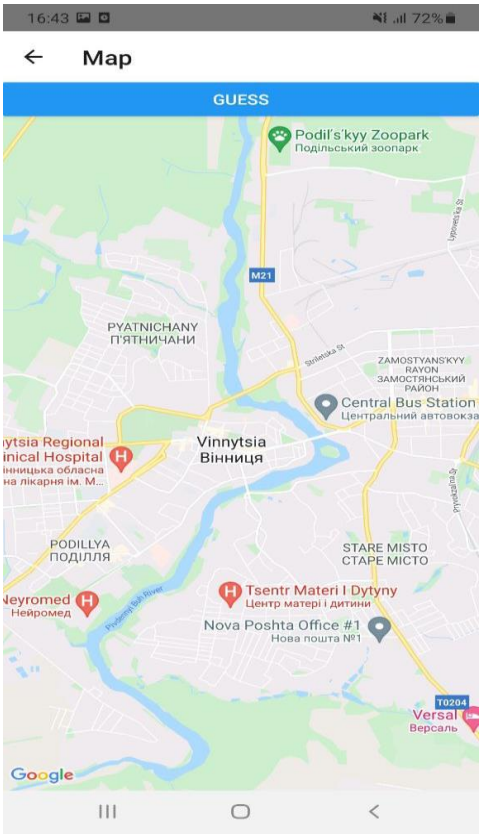

Рисунок 3.4 – Вікно позначення попередньо згенерованої панорамної локації

Таблиця 3.2 – Тест кейс №2. Проведення раунду інформаційної гри

Тестовий сценарій	Демонстрування згенерованої локації та вгадування її розташування
Кроки	Відкрити застосунок Натиснути кнопку почати гру Переглянути згенеровану локацію Відмітити на карті здогадку розташування Отримати оцінку
Очікуваний результат	Після перегляду локації та відмітки її розташування користувач має отримати оцінку його відповіді

Результати тестування тест кейсу номер 2 описані на рис. 3.5 – 3.8

Рисунок 3.5 – Вікно демонстрування згенерованої локації у місті

Рисунок 3.6 – Вікно відгадування продемонтсрованної локації на карті

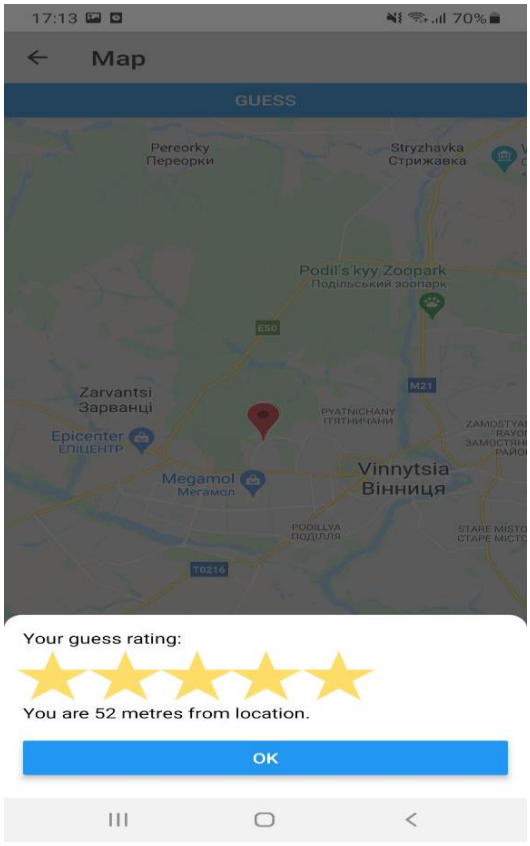

Рисунок 3.7 – Вікно демонстрування результату відгадування користувача

Розроблений мобільний веб-сервіс окрім описаних вище протестованих сценаріїв, передбачає також можливість авторизації з валідацією даних та створення профілю користувача. Також у додатку є можливість перегляду персоналізованої статистики для користувача, що базується на його попередніх відповідях, зроблених у застосунку.

3.5 Висновок.

У даному розділі було проведено та проаналізовано вибір середовища розробки для написання коду, обґрунтовано вибір мови програмування, яку найдоцільніше було використовувати для розробки. Систему також було проаналізовано відповідно з функціональними характеристиками та протестовано для створених тест кейсів. Додаток цілком відповідає поставленим вимогам.

ВИСНОВКИ

Під час виконання випускної кваліфікаційної роботи була поставлена задача створення інформаційно-розважального мобільного веб-сервісу.

При роботі над поставленою задачею, було перевірено актуальність обраної теми, ретельно проаналізовано предметну область сервісу, об'єкт проєктування, переглянуто існуючі додатки-аналоги та додатки, схожі за підходом і ідеєю, їх функціонал та призначення.

Визначено предмет та об'єкт дослідження, сформовано мет дослідження.

Було охарактеризовано основні складові мобільного інформаційно-розважального веб-сервісу та їх призначення. Проаналізовано та описано вимоги щодо існуючого та використаного програмного забезпечення, сформовано список основного функціоналу додатку. Розроблено структуру та алгоритм працездатності мобільної географічної гри.

В процесі розробки даного додатку був проведений аналіз відповідних засобів для реалізації застосунку. Розглянуто мови програмування, бібліотеки та фреймворки. Було порівняно їхні відмінності, плюси та мінуси та обрані конкретні задля написання програмної частини застосунку. Для написання поставлених вимог на стороні клієнту було використано мову програмування Javascript, використано фреймворк React Native. Для взаємодії з серверною частиною було використано Google Cloud Platform та AJAX запити.

Після вибору усіх засобів імплементації, систему було протестовано на конкретних тест-кейсах. Ророоблений додаток відповідає поставленим задачам.

Технічна реалізація системи відбувалась у середовищі Visual Studio Code.

Списки використаних посилань

[1] https://reactnative.dev/

[2] Boduch A. React and React Native. – Packt Publishing Ltd, 2017.

[3] https://gs.statcounter.com/os-market-share/mobile/worldwide

[4] https://technostacks.com/blog/mobile-app-development-frameworks

[5] https://dotnet.microsoft.com/en-us/apps/xamarin

[6] https://flutter.dev/

[7]https://www.raywenderlich.com/247-react-native-tutorial-building-android-apps-with-javascript

[8] Danielsson W. React Native application development //Linköpings universitet, Swedia. – 2016. – Т. 10. – №. 4.

[9] https://cloud.google.com/docs

[10] Grove R. F., Ozkan E. The MVC-web design pattern //International Conference on Web Information Systems and Technologies. – SCITEPRESS, 2011.