ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Навчально–науковий інститут Інформаційних технологій

Кафедра інженерії програмного забезпечення

Пояснювальна записка

до бакалаврської роботи
на ступень вищої освіти бакалавр

на тему «**СТВОРЕННЯ МОБІЛЬНОГО ЗАСТОСУНКУ ДЛЯ ВЕДЕННЯ СТАТИСТИКИ ПО ЛОВЛІ РИБИ НА ФРЕЙМВОРЦІ REACT NATIVE**»

Виконав: студент 4 курсу, групи ПД-42
спеціальності
121 Інженерія програмного забезпечення
(шифр і назва спеціальності)
Кравченко Юрій Васильович
(прізвище та ініціали)
Керівник Гаманюк І.М.
(прізвище та ініціали)
Рецензент
(прізвище та ініціали)

КИЇВ – 2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Навчально–науковий інститут Інформаційних технологій

Кафедра Інженерії програмного забезпечення
Ступінь вищої освіти «Бакалавр»
Спеціальність підготовки Програмна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного
забезпечення
_____О.В.Негоденко
“____” _____ 2021 року

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУЮ РОБОТУ СТУДЕНТУ

Кравченку Юрію Васильовичу
(прізвище, ім’я, по батькові)

1.Тема роботи: «Створення мобільного застосунку для ведення статистики по ловлі риби на фреймворці React Native»

Керівник роботи Гаманюк І.М, с.в.
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від «12» березня 2021 року № 625

2. Строк подання студентом роботи «1» червня 2021року

3. Вихідні дані до роботи: мобільні застосунки для ведення статистикипо ловлі риби. методи проектування архітектури проекту, методи розробки мобільних застосунків

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити).

4.1 Огляд предметної області
4.2 Вибір засобів реалізації
4.3Проектування застосунку
4.4 Реалізація застосунку

5. Перелік графічного матеріалу (презентація)

6. Дата видачі завдання__1.06.2021 року__

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврсбкої роботи	Строк виконання етапів роботи	Примітка
1	Отримання завдання на бакалаврську роботу	19.04.21	
2	Огляд предметної області	20.04.21	
3	Вибір інструментальних засобів реалізації	25.04.21	
4	Проєктування застосунку	28.04.21	
5	Реалізація застосунку	30.04.21	
6	Тестування застосунку	06.05.21	
7	Тестування програмної системи	10.05.21	
8	Написання та оформлення пояснювальної записки	15.05.21	
9	Розробка графічних та презентаційних матеріалів	01.06.21	
10	Захист бакалаврської роботи		

Студент _____ Ю.В. Кравченко
(підпис) (прізвище та ініціали)

Керівник роботи _____ І.М. Гаманюк
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина бакалаврської роботи: 52с., 26 рис, 1 дод., 3 табл., 15 джерел.

Ключові слова: React, SPA, MATERIAL UI, GOOGLE MAP, FORMIK, GIT, FRONT-END, FRAMEWORK. YUP, FIREBASE, REDUX.

Об’єкт дослідження – процес аналізу продуктивності риболова.

Предмет дослідження - аналіз продуктивності риболова за допомогою мобільного застосунку.

Мета роботи - розробка мобільний застосунку для риболовів за допомогою фреймворку React Native. Для реалізації мети було сформульовано та вирішено наступні завдання:

1. Аналіз існуючих мобільних додатків.
2. Розробка React-Native компонентів для візуалізації створеного додатка.
3. Підключення логічної частини компонентів до Redux.
4. Інтегрування клієнтської та серверної частин мобільного застосунку.
5. Вибір та вивчення необхідних для проекту node_modules.

Методи дослідження - методи проектування архітектури проекту, методи розробки мобільних додатків.

Під час виконання роботи був проведений аналіз існуючих мобільних додатків, таких як «Fishing Times Free», «Нотатки риболова» , що мають зхожий функціонал. В ході розгляду цих додатків встановленні їх переваги та недоліки.

ЗМІСТ

ВСТУП

Мобільні телефони давно перестали бути чимось незвичним. В останні роки функціонал та можливості гаджетів значно збільшилися і вони вже стають не просто як засоби зв'язку, а щось набагато більше.

Спочатку мобільні програми пропонувались для інформаційних цілей та для підвищення продуктивності, включаючи електронну пошту, календар, контакти, калькулятор та інформацію про погоду. Завдяки швидкому збільшенню технології та перспектив користувачів, розробники впроваджують їх в інші категорії, такі як мобільні ігри, GPS, банкінг, покупка квитків, соціальні медіа, відео-чати, автоматизація підприємств, послуги на основі місцезнаходження, фітнес-програми та нещодавно мобільні медичні програми.

Кількість людей які полюбляють займатися любительським та спортивним риболовством стає з кожним роком все більше і більше, а популяція риби навпаки стрімко скорочується. Тому зараз є актуальним створення мобільних застосунків де б люди могли обмінюватись між собою інформацією що до минулих риболовль а також знаходити для себе нові місця.

Об’єкт дослідження – процес аналізу продуктивності риболова.

Предмет дослідження - аналіз продуктивності риболова за допомогою мобільного застосунку.

Мета роботи - створення мобільний застосунку для риболовів за допомогою фреймворку React Native.

Методи дослідження - методи проектування архітектури проекту, методи проектування та розробки мобільних додатків, методи проектування та розробки користувацьких інтерфейсів.

Практична значущість результатів полягає в використанні розробленого мобільного додатку за для статистичного аналізу та відкриттю нових локацій.

Під час виконання роботи був проведений аналіз існуючих мобільних додатків, таких як «Fishing Times Free», «Нотатки риболова» , що мають зхожий функціонал. В ході розгляду цих додатків встановленні їх переваги та недоліки.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ТЕОРЕТИЧНІ ОСНОВИ

1.1.Операційні системи та дизайн

1.1.1. Вибір найбільш актуальної операційної системи

Говорячи про «розробку застосунків», ми як правило маємо на увазі розробку для мобільних пристроїв таких як смартфони та планшети.

Конкретніше, ми розглянемо розробку мобільних застосунків для двох основних опареційних систем: iOS та Android.

У всьому світі двійку лідерів операційних систем на яких працюють мобільні присторї займають iOS та Android.

Гаджети на iOS, особливо телефони iPhone - це продукція компанії Apple. У той час як інші пристрої на Android - це результат діяльності інженерів Google. Ця ОС використовується в різних моделях від багатьох виробників. Багато в чому схожий інтерфейс діючих операційних систем. Однак користувачам Android не складе труднощів змінити і встановити нові програми, а також підлаштувати систему під власні потреби. В iOS всі зміни суворо контролюються. У більшості випадків вони виявляються поверхневими, а користувачеві доводиться працювати з існуючими шаблонами.

Спочатку може здатися, що Android біль цікава операційна система.
Адже в GooglePlay представлені тисячі різноманітних програм. Однак це можна віднести і до плюсів, і до мінусів.

Справа в тому, що щодня для Android створюються сотні додатків. Ретельно перевірити кожне з них практично неможливо, чим і користуються недобросовісні розробники. Часто в GooglePlay зустрічається контент, переповнений рекламою і вірусами.

В iOS за цим стежать набагато серйозніше. У AppStore заливають меншу кількість програм, і кожну з них ретельно перевіряють. Тому ризик

завантаження неякісного контенту зводиться до мінімуму. При цьому передбачено софт під різні завдання, необхідні конкретному користувачеві.

Згідно зі статистикою за початок 2021 року Android займає 70% всього ринку мобільних телефонів, а більшість решти займає IOS зі своїми пристроями iPhone та iPad.

Тому мій вибір при розробці мобільного застосунку впав саме на операційну систему Android. Так як вона є найросповсюдженіша в світі. Також для розповсюдження власного застосунку під операційну систему Android мені не обов'язково купляти платний аккаунт розробника, який наразі вартує близько 25$, адже програму можна встановити з файлу формату APK, в той час як на IOS всі сторонні файли будуть автоматично заблоковані.

1.1.2. Проблематика мобільного дизайну

Однією з найбільших проблем при створенні мобільного застосунку це розробка дизайну. Адже сучасні моделі смартфонів мають різні розширення та співвідношення сторін дисплеїв. Зазвичай розробники мають окремі версії відображення під певні параметри такі як: розмір, операційна система та орієнтація.

Зараз є декілька типів мобільного веб-дизайну:

– Адаптивний дизайн – почав серйозно застосовуватись з появою мобільних пристроїв, використовуючи переваги бізнес-логіки, розміру та орієнтації CSS-медіа-запитів[1]. Його ідея полягає у реагуванні на різні статичні та мінливі особливості відображення програми.

– Адаптивний дизайн - як правило, відноситься до фіксованих, дискретних вимірювань адже він розміщує контент згідно розміру екрану. В адаптивному дизайні ви бачите більш точні розміри для кількох роздільних здатностей дисплея.

– Прогресивний дизайн - відноситься до програм які прогресивно завантажують контент, працюють в офлайн-режимі за відсутності мережі[2]. Простими словами цей вид дизайну заміщає при оновленні чи підгрузці даних блоки програм ідентичними за формаю та розміром заглушками

В своєму застосунку я буду використовувати змішаний тип дизайну, що має в собі фрагменти адаптивного та прогресивного відображення.

1.2. Аналіз існуючих мобільних застосунків

При будь-якій розробці хоч то мобільний застосунок чи веб сайт одним з перших етапів має йти аналіз існуючих рішень. Адже може виявитися так, що ваша ідея вже кимось була в повній мірі реалізована і не має ніякого сенсу створювати ще один такий продукт. Хорошою практикою зараз є доопрацювання готових ідей та створення на базі цього нових додатків.

Наразі для риболовлі є не так багато мобільних застосунків. Основна їх частина це календарі клювання та блокноти для записуваня нотаток. Провівши пошук в Google play за ключовим словом «риболовля» я зміг віднайти тільки два додатки які б в якійсь би мірі мали відношення до моєї ідеї.

1.2.1. Щоденник риболова.

Щоденник риболова - це програма для обліку уловів і риболовних трофеїв. Застосунок містить в собі симбіоз календаря для риболовлі та записника своїх минулих риболовель.

Переваги сервісу:

- календар для риболовлі;
- безкоштовність та доступність;
- зрозумілий інтерфейс;
- немає реклами;

Недоліки сервісу:

- Відсутність повноцінної статистики по минулих риболовлях;
- Неможна поділитися результатами зі своїми друзями;
- Застарілий дизайн;
- Немає ніякої взаємодії користувача з картами;
- Немає української локалізації.

1.2.2. Fishing Times Free

Fishing Times Free – ця програма пропонує легкий для читання календар риболовлі Solunar, час припливів та інформацію про сонце / місяць для планування ваших риболовецьких поїздок. Якщо ти знаєш, що робиш, це допоможе тобі знайти добрі часи та дні, щоб збільшити свої шанси ловити рибу. В преміум версії цього застосунку можна додавати свої риболовлі та отримувати статистику по них.

Переваги сервісу:

- Календар для риболовлі;
- Зрозумілий інтерфейс;
- Статистика по попередні риболовлях

Недоліки сервісу:

- Для повного функціоналу необхідно купляти преміум пакет.
- Неможна поділитися результатами зі своїми друзями;
- Застарілий дизайн;
- Велика кількість реклами;
- Немає української локалізації

1.2.3. Порівняльна таблиця з аналогами

В даній таблиці табл 3.2.1. представлене порівняння мого застосунку з аналогами. Після порівняння можна зробити висновки, що до можливостей кожного з застосунків та побачити, що можна було б згодом покращити.

Таблиця 3.2.1 – Порівняння з аналогами.

Найменування	**Мій застосунок**	**Fishing Times Free**	**Блокнот риболова**
Українська локалізація	+		
Взаємодія з картами	+	+	
Календар риболова		+	+
Статистика по ловлі риби	+	+	
Можливість поділитися інформацією з друзями	+		
Інформація що до зимувальних ям, та заборонених місць лову	+		

1.3.Постановка задачі

Задачею даної роботи є реалізація мобільного застосунку для ведення статистики по ловлі риби основуючись на перевагах та недоліках вже готових рішень. Застосунок повинен задовольняти наступні користувацькі вимоги :

- надавати користувачеві інформацію по минулим риболовлям у вигляді графіку за допомогою якого він зможе аналізувати та робити висновки з їх результативності ;

- надавати користувачеві можливість створювати та переглядати пости які будуть стосуватися лише риболовлі;
- застосунок обов’язково повинен мати мапу, за допомогою якої користувач зможе зберігати місця риболовель.

Проаналізувавши уже готові рішення та використавши деякі з них на практиці, можна зробити висновок, що зараз більш актуально буде зробити застосунок з невеликою кількістю функціоналу, проте весь він має бути дійсно доцільний та корисний для простого риболова.

1.4.Висновки до розділу

В даному розділі було розібрано основні операційні системи, їх переваги та недоліки. Зроблено вибір під яку саме ОС розробляти мобільний додаток. Розглянуто основні підходи створення дизайнів для мобільних застосунків та обрано два з них. Також було проведено аналіз існуючих аналогів, встановлені їх недоліки та переваги один над одним. Після проведення аналізу існуючих аналогів були зформовані та описані для майбутнього застосунку.

2 ФОРМУВАННЯ ВИМОГ ДО ІНТЕРФЕЙСУ. ВИБІР СТЕКУ ТЕХНОЛОГІЙ

2.1.Досвід взаємодії з користувачем

Досвід взаємодії з користувачем (від англ. user experience, перекладається як досвід користувача) - це розробка досвіду взаємодії користувача з інтерфейсом. Досвід користувача зазвичай включає всі емоції, переконання, переваги, , фізичні та психологічні реакції[3], відчуття поведінку і досягнення, які виникають до, під час і після використання системи. Досвід користувача поєднує образ торгової марки, спосіб представлення, функціональність, продуктивність системи, інтерактивну поведінку і допоміжні можливості інтерактивної системи, фізичний і психологічний стан користувача, що є результатом попереднього досвіду звичок, навичок і індивідуальності.

Цей термін чато застосовується в інформаційних технологіях для опису суб’єктивного ставлення, що виникає в користувача в процесі використання як окремою частиною, так і програмним продуктом в цілому.

Досвід користувача залежить від такого поняття як людино-комп’ютерна взаємодія і і usability(зручність користування). Розробники завжди приділяють велику частину часу який відведений для розробки на проектування досвіду користувача.

П'ять рівнів - це модель, яку запровадив Дж. Гарреттом для повноцінного проектування досвіду користувача для програмних[4] продуктів. Модель пропонує основу для обговорення проблем, пов'язаних з досвідом користувача, а також можливих шляхів і засобів їх вирішення. Розробка програми починається з верхнього рівня (стратегії), на цьому рівні досить розпливчасто описується майбутній програмний продукт зі сторони як замовника так і користувача . В ході роботи над проектом - тобто просування вниз по рівнях - рішення, пов'язані з досвідом користувача стають конкретніше і знаходять більш високий ступінь деталізації. Кожен наступний рівень тісно пов'язаний з попереднім (верхнім.

Очевидно, при такому варіанті розробки діапазон можливих рішень скорочується з кожним натсупним переходом на більший рівень. Втім, це не означає, що всі рішення на конкретному рівні повинні бути прийняті до переходу на наступний рівень.

Описати рівні можна таким чином:

1) Перший рівень називається поверхневих та відповідає за візуальну частину майбутнього продукту Це самий нижчий та самий деталізований рівень. На ньому описуються такі дрібні деталі дизайну як крапки, текст, картинки, кнопки та інші.

2) Далі йде рівень компонування, описує конкретну реалізацію абстрактно зформованої структурної частини майбутнього застосуноку.На ньому описується та вирішуються завдання найкращого розташування елементів які відповідають за відображення(UI елементи).

3) На третьому рівні – так званому структурному відбувається опис взаємного розташування сторінок, вікон та інших. До цього рівня можна поставити запитання "куди", "як" та "звідки", кінцевий користувач зможе переходити. Хороша структура значно полегшує навігацію та робить її значно зрозумілішою для кінцевого користувача.

4) Четвертий рівень являється простим переліком функціональних частин, котрі користувач зможе використовувати.

5) П’ятий, останній та самий розмито описаний рівень це рівень стратегії. На ньому рівні потрібно подумати та дати відповідь на запитання які стосуютсья очікувань від майбутнього продукту, як зі сторони кінцевого користувача так і замовника. Ці відповіді будуть представлені у вигляді певного списку.

2.2.Формування вимог до інтерфейсу

Ключовою частиною кожного програмного продукту є саме інтерфейс. Враховуючи поставлені вимоги до мобільного застосунку веб інтерфейс буде розроблятися окремими компонентами. За допомогою такого підходу реалізується певна автономність та незалежність компонентів один від одного, що в свою чергу збільшить швидкодію, стійкість та масштабованість. Однак для користувача не є принциповою архітектура на якій побудований застосунок, тому він створює свої вимоги:

1) Можливість реєстрації
2) Адміністратор повинен мати змогу підтвердити/заблокувати обліковий запис
3) Користувач повинен легко авторизуватись
4) Користувач повинен мати можливість створювати свої та переглядати пости створені іншими користувачами
5) Користувач повинен мати можливість завантажити/відвантажити фотографії
6) Користувач повинен мати можливість отримання статистики по ловлі риби
7) Має бути можливість редагування профілю.

2.3.Вибір фреймворку

На теперішній час вибір фреймворку для кросплатформенної мобільної розробки є досить складним питанням, адже кожен з ним має свої плюси та мінуси та використовується в певних випадках. Перед написанням роботи було розглянуто 5 з них: React Native, Flutter, lonic, Xamarin, PhoneGap

2.3.1. React Native

React Native – наразі самий популярний з всіх кросплатформерних фреймворків для мобільної розробки[5]. Він був створений в 2013 році на внутрішніх змаганнях з програмування світового гіганту – компанії Facebook. Перша ж офіційна версія побачила світ в січні 2015 року.

Популярність цьому фреймворку приніс його архітектурний підхід до написання коду. « Напиши один раз, використовуй повторно » - головний принцип який передбачає використання одного й того ж самого коду для різних платформ. Також в Native вбудована функція Hot Reloading яка дозволяє додавати новий код і вносити правки до старого прямо під час виконання програми, що є дуже зручним при налаштуванні користувацького інтерфейсу. Ще однією з його особливостей є те, що після ініціалізації проекту фреймворк вже надає велику кількість готових компонентів, хоч деякі з них і потребують адаптації під різні платформи.

Так як React Native націлений на результат який можна зпівставити з нативною розробкою, в погоні за швидкодією частіше всього надають перевагу цьому фреймворку. Native також дає можливість використовувати кастомні модулі на мовах програмування для нативної розробки, але їх доведеться писати для кожної платформи окремо.

З моменту запуску React nativ пройшло близько 7 років, через це його підтримують майже всі IDE. Вивчати цей фреймворк та писати на ньому доволі легко, завдяки використанню JavaScript (звісно якщо ви знаєте JavaScript)

Плюси React Native

– Функція Hot Reloaded, що дозволяє значно зекономити час на перекомпіляції коду

– Можливість використання кастомних модулів написаних на інших мовах нативної розробки.

– Велика кількість модулів які вже вбудовані у фреймворк.

– Величезне комюніті розробників, до яких можна звернутися з будь якими питаннями.

Мінуси React Native

– Доволі часті оновлення. Застосунок не можна залишити на рік, а потім одразупочати додавати новий функціонал. Потрібен буде деякий час для оновлення залежних до фреймворку бібліоткек

– Складний алгоритм пошуку помилок по додатку.

2.3.2. Flutter

Flutter вперше був показаний компанією Google в 2015 році. Реліз першої стабільної версії відбувся 4 грудня 2018 року. Розробка на цьому фреймворку ведеться на об’єкто орієнтованій мові програмування Dart.

Хоч Flutter і є відносно новим фреймворком, проте він вже здобув хорошу репутацію в кросплатформенній розробці[6]. В його принципі закладене створення застосунків с однією кодовою бозою для мобільних платформ, веб-додатків та десктопа. Проте якщо вам необхідні різні стилі для різних операційних систем, вам доведеться трохи попрацювати, а все через те, що замість нативних компонентів (підходу який використовується в React Native) Flutter використовує свій властний графічний двигун. В останній версіях фреймворку також з’явилася підтримка функції Hot Reloading (додавання та рефакторинг коду без повторної збірки проекту). Ще одна з особливостей даного фреймворку є можливість випускати застосунки для різних версій Android без додаткових маніпуляцій з кодом.

Плюси Flutter

– Функція Hot Reloaded, що додалась до фреймворку в недавніх оновленнях
– Властний графічний двигун.
– Інтерфейс легко розбивається на окремі модулі.

Мінуси Flutter

– Нестабільність – фреймворк лиш відносно недавно вийшов з beta, і в ньому де не де зустрічаються баги

– Досить не велика кількість бібліотек в порівнянні з нативною розробкою

– Мова розробки Dart, хоч вона і є продуктивною, проте для її вивчення знадобиться досить значний часовий ресурс.

– Не велике комюніті розробників. Для вирішення якогось не зрозумілого питання, інколи потрібно чекати по місяцю для, того щоб отримати відповідь

2.3.3. Lonic

Перший офіційно випущений фреймворк для розробки гібридний мобільних додатків. Перша версія була випущена в 2013 році та в ті часи мала підтримку AngularJS та Apache Cordova.

Останні версії даного фреймворку пропонують концепції єдиної бази коду лоя різних платформ, проте на новуму рівні. Всі його компоненти автоматично адаптуються до тіє платформи на котрій запускається застосунок – а значить і

розробка стає швидшою та більш продуктивною. Ще однією особливість Lonic є можливість написання окремих модулів на різних фреймворках таких як Angular, React, Vue.

Плюси Lonic

– Швидкість розробки.

– Для розробки можна використовувати браузер.

– Велика кількість вбудованих в фреймворк готових компонентів.

Мінуси Lonic

– Швидкодія – дуже важливий пункт, адже застосунки розроблені на базі Lonic програють в декілька разів по продуктивності застосункам розробленим React Native.

– Не велика кількість бібліотек. Та не можливість використання деяких з ним про роботі з модулями написаними на інших фреймворках.

– При білді застосунку він може ламатися без причини при цьому ніяк не дававши наводки для розробника по помилках.

2.3.4. Xamarin

Xamarin є досить популярний фреймворком для розробки мобільних додатків. Він був випущений компанією Microsoft в 2013 році. Однією з його особливостей є можливість розробки під Windows [7].

У Xamarin є два основні інструменти Xamarin.Android/iOS та Xamarin.Forms. По частині кросплатформеннх додатків даний фреймворк пропонує використовувати API Xamarin.Essentials. Xamarin.Android та Xamarin.IOS можуть наділяти застосунки тіми ж самими можливостями, що і нативні рішення. У випадку з Xamarin IOS програма компілюється безпосередньо в машинний код, тоді як в Xamarin.Android спочатку в відбувається компіляція в байт-код, який потім інтерпретуєтсья в віртуальний машинний, що не є хороший рішенням в плані оптиміцації робити застосунків.

Плюси Xamarin

– Швидкість розробк. (одна база коду).

– Технічна підтримка від Microsoft.

– Можливість розробки під Windows.

Мінуси Xamarin

– Майже не має оновлень.

– Не велика спільнота розробників.

– Складно розробляти застосунки у які мають входити тяжкі графічні елементи.

– Досить не велика продуктивність.

2.3.5. PhoneGap

PhoneGap вийшов в реліз в 2013 році. Як і Lonic, PhoneGap дозволяє використовувати веб-технології в розробці мобільних додатків. Він являє собою дистибутив Apache Cordova.

Застосунки розроблені на PhoneGap по суті являють собою набір HTML-сторінок, обгорнутих в нативну[8] оболочку, що по факту не дає користувачеві користувацького досвіду як від повноцінного нативного застосунку. Сторінки в ньому зберегаються в локальному каталозі або в хмарних сховищах, а під час запуску вони отримують доступ до функцій смартфону через плагіни. Цей фреймворк також не може похвастатись значною швидкодією в порівнянні зі своїми аналогами, а все через обгортання кожного зі сторінок в нативну обгортку.

Плюси PhoneGap

– Не велика вага застосунку
– Єдина база коду для мобільних та веб застосунків.

Мінуси PhoneGap

– Майже не має оновлень.
– Не велика спільнота розробників.
– Складність інтеграції графічних анімацій.
– Для розробки потрібно використовувати не популярні та для більшості розробників не знайомі IDE.
– Не має підтримки Hot Reloaded.

2.3.6. Порівняльна таблиця фреймворків

В таблиці табл.2.3.6 представлені всі найпопулярніші фреймворки для розробки мобільних застосунків. Після порівняння можна зробити висновки щодо доцільності використання того чи іншого фреймворку для свого проетку.

Таблиця 2.3.6 – Порівняння фреймворків для кросплатформерної розробки

Найменування	**React Native**	**Flutter**	**Lonic**	**Xamarin**	**PhoneGap**
Підтримка функції Hot Reloading	+	+	-	-	-
Мова програмування	JavaScript + React	Dart	JavaScript, HTML, CSS + Angular, React, Vue	C# + .NET	JavaScript, HTML, CSS
Спільнота розробників	Дуже велика	Не велика	Не велика	Не велика	Не велика
Відкритий код фрейворку	Так	Так	Так + платні пакети	Так + платні пакети	Так
Швидкодія	Висока, наближена до нативної	Висока, наближена до нативної	Середня	Не велика	Середня
Вбудовані компоненти	Так	Ні	Ні	Так	Ні

2.3.7. Висновок щодо вибору фреймворку.

Проаналізувавши дані з порівняльої таблиці 2.3.6 я можу зробити вибір на користь React Native, адже він дійсно вібрав в себе весь той функціонал та можливості які зараз необхідні для повноцінної розробки мобільних застосунків.

2.4.Вибір CLI для розробки

CLI – це візуальний інтерфейс, де програмісти можуть запускати команди для розробки програм. Початок роботи з React-Native розпочинається саме з вибору CLI, і це той етап на якому майже всі помиляються. React-Native підтримує дві CLI, і під час вибору одного з них ви маєте бути абсолютно впевненими.

2.4.1. React Native CLI

Перше, що дуже важливо відзначити, це те, що після того як ви вирішили вибрати React Natie CLI або хочете перейти на нього з Expo, дороги назад вже немає, оскільки це незворотній процес. Особливістю[9] React Natie CLI є те, що він дає розробнику повну свободу впровадження будь-яких сторонніх плагінів та можете використовувати окремі модулі написані на Java/Objective-C. Також ви не обмежені мінімальними версіями ОС(**1)** та жодною версією React Native.

Плюси React Native CLI

– Ви можете використовувати властні модулі написані на Java/Objective-C.

– Є підтримка функції Bluetooth.

– Підтримка всіх версій React Native.

– Можливість виконати build застосунку у будь-який час.

Мінуси React Native CLI

– Потребує Android Studio та Xcode для запуску проектів.

– Ви не зможете розробляти під IOS без Mac.

– Пристрій має бути підключений до комп’ютера через USB, щоб його можна було використовувати для тестування.

– Шрифти потрібно імпортувати окремо через Xcode.

– Вірне налаштування робочого процесу включаючи налаштування пристороюю для тестування є доволі складним заняттям та може зайняти певну кількість часу.

2.4.2. Expo CLI

Expo – це набір інструментів побудованих поверх React Native. Ці інструменти залежать від одного ключового моменту, якого ми притримуємось притримуєится при розробці за допомогою[10] Expo: більшість застосунків можна створювати за допомогою власного коду при умові, що у нас є повний набір необхідних API-інтерфейсів JavaScript.

Це важливо, тому що в React Native ви завжди можете перейти до нативного коду. Інколи це надзвичайно користно, але це несе за собою важкість в тестуванні.

Expo надає хмарний компілятор коду за допомогою якого ви можете встановити їхній додаток собі на телефон та з легкістю приєднатись до нього, що значно спрощує процес розробки. Також для того, щоб надати доступ тестувальнику до проекту вам вже не обхідно робити білд проекту та надсилати йому apk файл, а лиш просто поділитись ссилкою на хмарний компілятор.

Для того щоб виконати білд застосунку Expo надає свій властний сервіс, але слід зазначити, що ви маєте відстояти в черзі для виконання білду, обійти це можна шляхом придбання преміум підписки на сервіси Expo.

Плюси Expo CLI

– Налаштування проекту досить просте і може бути виконане за декілька хвилин.

– Поділитися проектом досить легко, достатньо лиш надати ссилку або QR-код, вам не потрібно надсилати весь файл apk.

– Не потребує збірки для запуску проекту, все відбувається в хмарних сховищах.

– Одразу інтегрує деякі базові бібліотеки в стартовий проект (Push-сповіщення, Asset Manager)

– Expo може створювати apk та ipa файли, а також допомагає завантажувати їх в інтернет магазини накшталт Play Market.

Мінуси Expo CLI

– Ви не можете додавати нативні модулі.

– Ви не можете використовати бібліотеки котрі використовують код написаний на Objective-C / Java.

– Прості застосунки накшталт виведення на екран Hello World на екран можуть займати близько 25МБ через вбудовані бібліотеки.

2.4.3. Порівняльна таблиця CLI

В таблиці 2.4.2 можемо наглядно побачити плюси та мінути використання в своєму проекті тієї чи іншої CLI.

Таблиця 2.4.2 – порівняння CLI

Найменування	**Expo CLI**	**React-Native CLI**
Можливість налаштувати проект за декілька хвилин	+	-
Можливість поділитися проектом за допомогою силки або QR-коду.	+	-
Вага тільки ініціалізованого застосунку	25-мб	7-мб

Одразу встановлені JS API, такі як : Push-сповіщення, Asset Manager	+	-
Є підтриика Bluetooth	-	+
Можливість використоувати властні модулі написані на Java/Objective-C.	-	+

Виходячи з таблиці 2.4.2 можна зробити вибір в користь Expo CLI так як в даній роботі у мене не має потреби у використанні власних модулів написаних на навивних мовах, але необхідна можливість тестування та дебагу застосунку в хмарних середовищах.

2.5.Вибір необхідних бібліотек.

Виходячи з поставлений задач до мобільного застосунку тепер постає питання з необхідних для розробки додаткових бібліотек. Хоч React Native має вже досить великий вбудований функціонал але для його розширення та для більш зручної розробки можна обрати да довстановити деякі з бібліотек.

Formik & Yup - так як в застосунку передбачена досить велика кількість форм, то доцільно використовувати бібліотеку яка б зробила роботу з ними більш зручними. Formik є дійсно хорошим вибором для роботи з формамм, поперше він покращує швидкодію застосунку завдяки тому, що після зміни одного з полів форми він не робить повного оновлення компоненту на відміну від вбудованого інтсрументу. Ще одним плюсом використання Formik є його сумістність з такою бібліотекою для валідації як Yup. В парі ці два інструменти дають розробнику можливість зберегти час написання коду, та покращити оптимізацію майже в два рази.

Material Ui – це досить об’ємна бібліотека з великою кількістю вже готових React комонентів. При розробці мобільного дадатку це може зекономити досить багато часу. В основі MateialUi закладий JSS (стилізація css за допомогою мови програмування JavaScript).

React Navigation – одна з найголовніших бібліотек при створенні будь-якого з мобільних застосунків. Завдяки цій бібліотеці розрозбинк може з легкістю налаштовувати переходи між сторінками та підключати функції лінивого завантажування, що позитивно впливає на швидкодію застосунку.

React Native Maps – бібліотека яка надає можливість використовувати весь спектор функціоналу Google Maps у випадку розробки під Android та Aple Maps при розродці під операційну систему IOS.

2.6. Архітектура розробки застосунку

2.6.1. Flux

Flux – це архітектура відповідальна за створення слою даних в JavaScript застосунках та за розробку серверної частини в веб-застосунках[11]. Flux дпопвнює презентаційні компоненти в React використовуючи однонаправлений потік даних авдяки чому він є простим для дебагу помилок. Даніз в ньому проходять через прямий потік вашого застосунку.

Цей архітектурний підхід передбачує 4 головних компоненти рис 2.6.1:

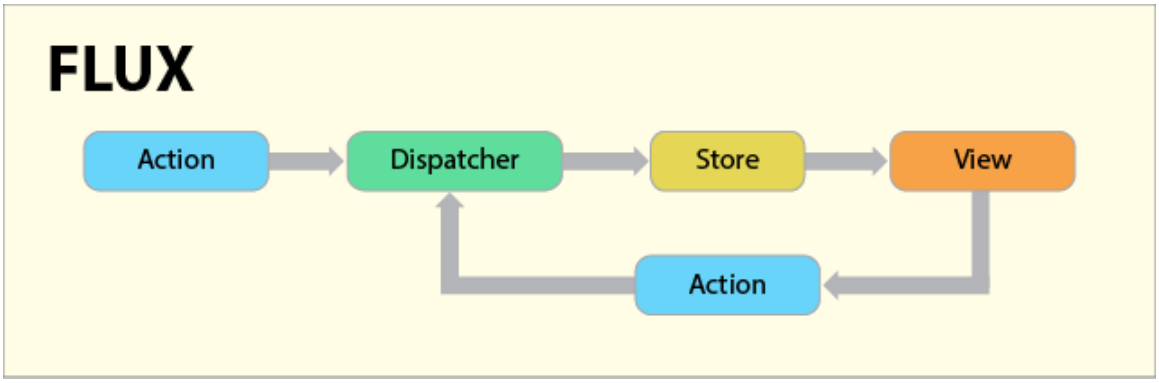


Рисунок 2.6.1 – Складові частини Flux архітектури

– Дія (Action) – помічник який передає дані в діспатчер.

– Діспатчер (Dispatcher) – отримує дії та передає їх до сховища.

– Сховище (Store) – діє як контейнер для зберігання стану та логіки застосунку. Справжня робота застосунку відбувається в сховищі. Сховища зарезервонані для прослуховування дій діспатчеру.

– Відображення (View) – React компоненти захоплюють стан із сховища та передають дочірнім компонентам.

Коли відбуваєтсья дія, діспатчер відправляє дані в сховище, яке зарезервоване для прослуховування саме цієї події. Тепер в сховищі необхідно обновити відобращення, що в свою чергу викличе чергу зі змін. Компоненти перегляду розповсюджують дії через центральний діспатчер і це буде відправлено в розні сховища. Ці сховища і будуть відповідати за логіку застосунку та за інші дані. Сховище оновлює всі компоненти перегляду. Це підтверджує те, що Flux слідує за однонаправленним потоком даних а Action, Dispatcher, Store та View є незалежними вузлами з конкретними вхідними та вихідними даними.

2.6.2. Redux

Виходячи з того, що для розробки було обрано саме Flux архітектуру, то для її реалізації буде використовуватись найбільш з розповсюджена її бібліотека – Redux. Якщо коротко говорити – то ця бібліотека займається станом додатку[12]. Хоч у React і є свій властний метод управління станом, він погано маштабується через необхідність передавати цей стан від батьківського компоненту до дочірнього, а якщо вкладеність є великою, то це призводить до «Prop drilling» - процесу який потрібно пройти щоб отримати дані до частин дерева React Component. Redux же надає можливість брати та викликати стан в будь якій його частині без передачі по компонентах.

Структура Redux рис 2.6.2 схожа до структури Flux, проте є невеликі відмінності.

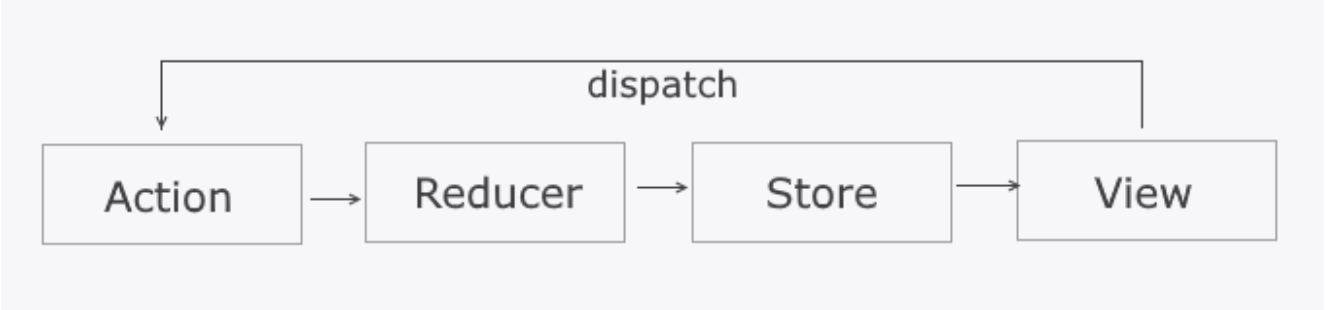


Рисунок 2.6.2 – Складові частини Redux

Відмінність заключається тільки в додаванні Reducer. По суті Reducer – це функція яка отримує дію і в відповідності до неї змінює стан сховища.

В Redux спільний стан додатку представляється один об’єктом JavaScript – state або state tree[13] (дерево стану). Незмінне дерево стану доступне тільки для читання, змінювати його напряму категорично забороняється. Зміни можливі тільки за допомогою відправки дії (action). Передача дії з потоком даних відбувається через виклик методу dispatch() в сховищі. Саме сховище передає дії в reducer і генерує наступний стан, а потім оновлює стан і сповіщає про це всіх слухачів.

2.7. Висновки до розділу

В даному розділі було проведено аналіз існуючих фреймворків для кросплатформерної розробки та зроблено вибір на користь одного з них. Також були сформовані вимоги до інтерфейсу та обраний пригідний CLI. І на кінець обрано архітектуру побудови застосунку та всі необхідні бібліотеки для зручної та розробки.

3 РОЗРОБКА МОБІЛЬНОГО ЗАСТОСУНКУ

3.1.Ініціалізація та розгортання проекту

3.1.1. Ініціалізація проекту за допомогою Expo CLI.

Так як для розробки було обрано Expo CLI, то ініціалізація проекту займе не велику кількість часу. Для того щоб розпочати роботу нам необхідно виконати декілька кроків:

1) Переглянути версію nodeJs рис 3.1.1, якщо вона застаріла, то оновити до останніх версій

```
node -v
```

Рисунок 3.1.1 – Команда для встановлення останньої версії nodeJs

2) Встановлення Expo CLI за допомогою команди рис

```
npm install -g expo-cli
```

Рисунок 3.1.2 – Команда для встановлення Expo CLI

3) Створення нового проекту рис 3.1.3

```
expo init my-project
```

Рисунок 3.1.3 – Команда для створення нового проекту

4) Вибір варіанту попереднього налаштування проекту рисунок 3.1.4

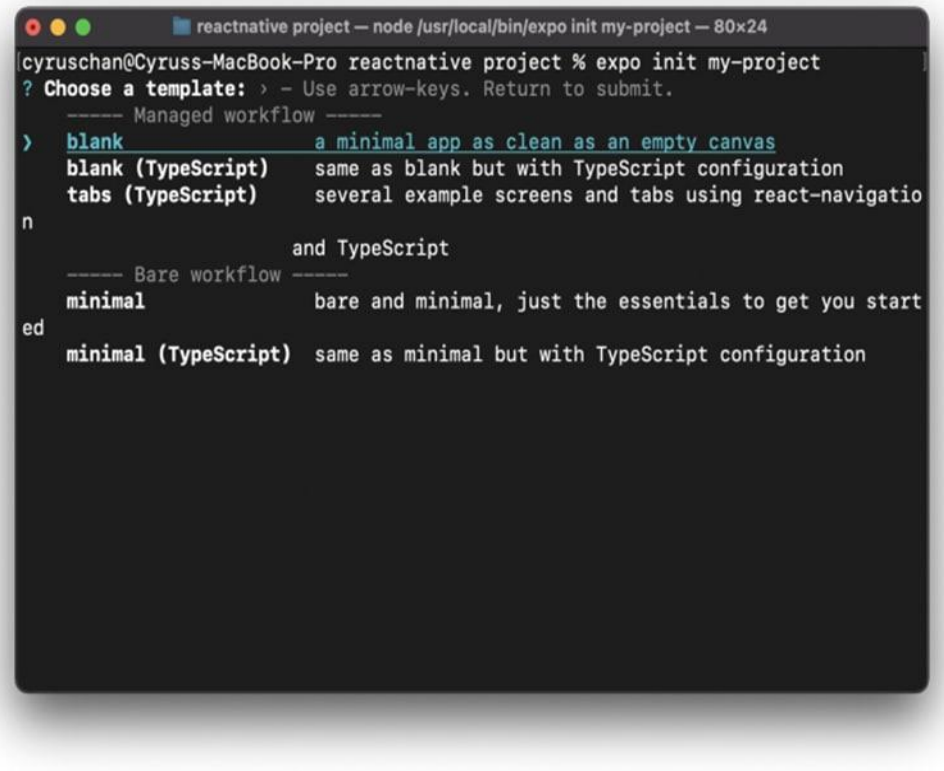


Рисунок 3.1.4 – Вибір варіантів попереднього налаштування проекту

Expo пропонує на вибір 5 варіантів попереднього налаштування:

– blank – це налаштований та повністю готовий для розробки проект.

– blank (Type Script) - це налаштований проект який буде розроблятися на Type Script.

– tabs (Type Script) – це налаштований проект для Type Script з додатковим встановленням навігації.

– minimal - це мінімально налаштований проект.

– minimal () - це мінімально налаштований проект який буде розроблятися на Type Script

Тут я обрав перший варіант, за для зменшення затрат часу на налаштування проекту.

3.1.2. Встановлення необхідних бібліотек

Наступним етапом після ініціалізації проекту являється – встановлення всіх необхідних для розробки бібліотек (ми обирали їх в 2 розділі).

Для встановлення бібліотек необхідно зайти в командну строку та прописати код який складається з двох частин де перша це «npm install» яка відповідає за безпосереднє встановлення бібліотеки, другою частиною коду є «package_name» і вона вже являється безпосередньо назвою необхідного нам пакету.

По завершенню встановлення бібліотек ми можемо перевірить їх версії та залежності в файлі packaje.json рис 3.1.4.

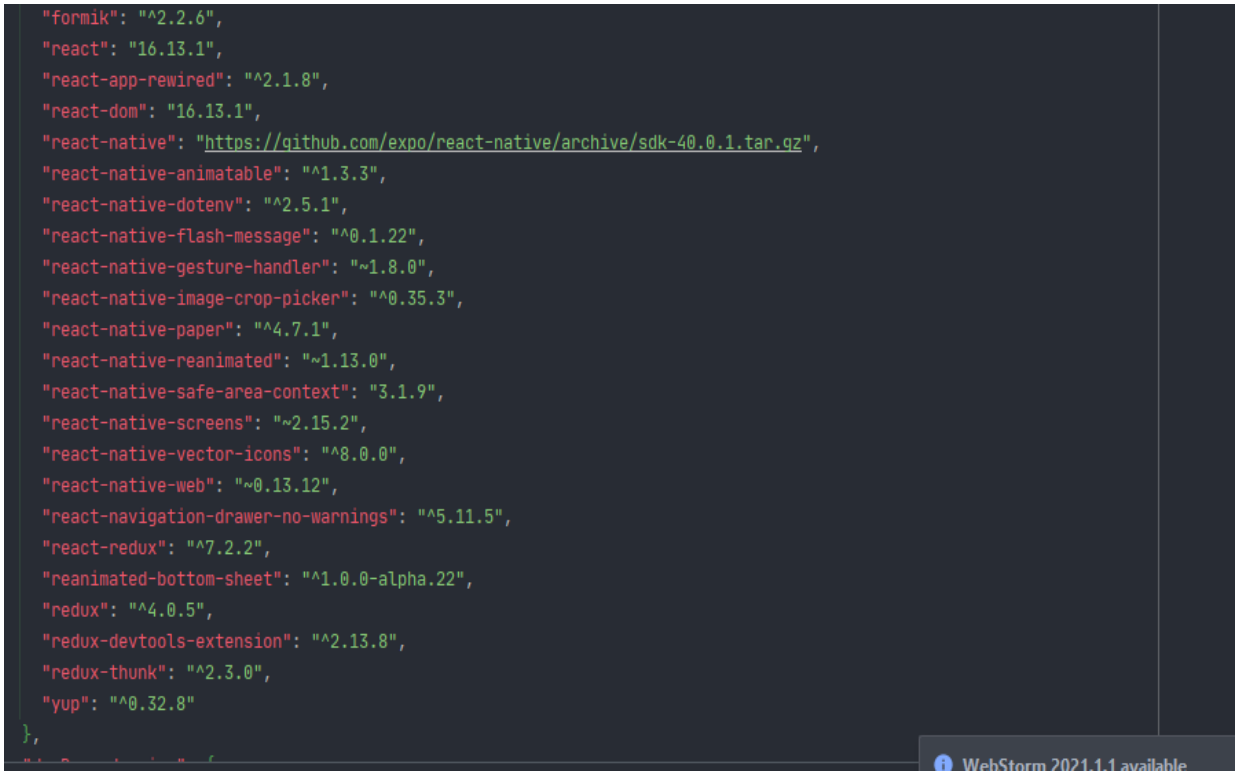


Рисунок 3.1.4 – Файл package.json

3.1.3. Розгортання проекту.

Тепер коли налаштування проекту завершене та всі необхідні бібліотеки встановлені ми можемо приступити до розгортання проекту.

Для запуску потрібно в консолі прописати «expo start», після чого почнеться процес старту проекту.

Коли збірка проекту закінчиться і проект запуститься, то автоматично відприється нове вікно у браузері з Expo Developer Tools рис 3.1.5. В цьому вікні в нас є панель навігації та, щро саме головне – консоль через яку й відбувається відловлювання помилок по всьому застосунку. Також в ньому відбуваютсья тестові виведення інформації та сповіщення по попередженнях.

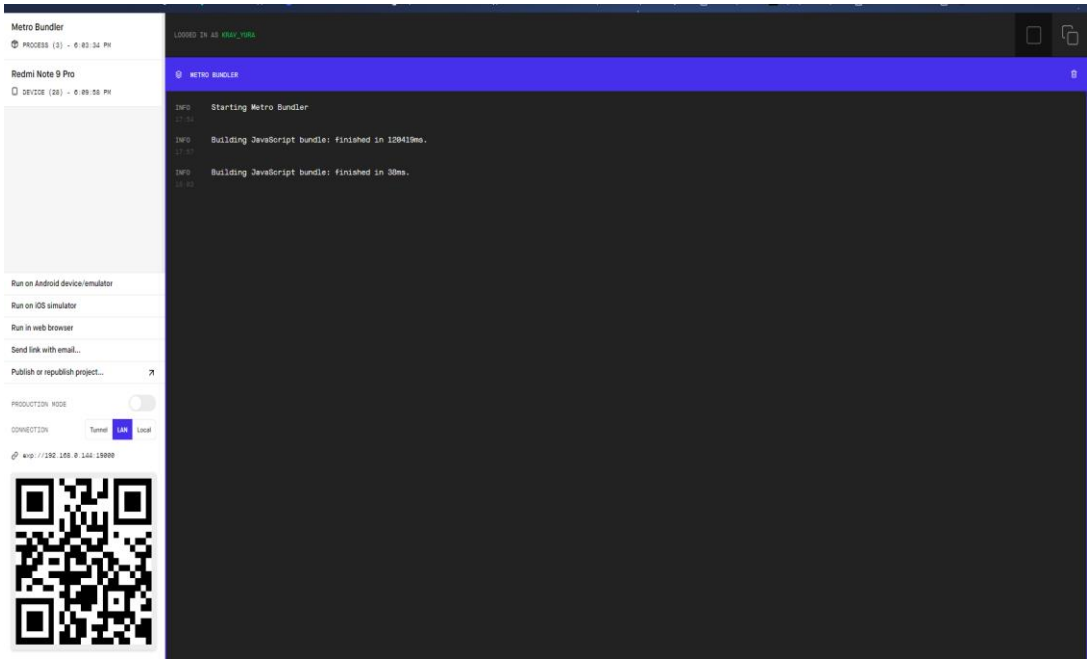


Рисунок 3.1.5 – Expo Developer Tools

На панелі навігації для подальшої роботи ми обераємо один з чотирьох варіантів відкриття проекту:

– Відкрити на емуляторі накшталт Android studio

– Відкрити на ios девайсі

– Відкрити в браужері (деякі з бібліотек для нативної розробки не будуть працювати, тому це не кращий вибір)

– Останнім варіантом являється QR-код який потрібно відсканувати з встановленого на телефон додатку від розробників.

Так застосунок буде розроблятися для операційної системи android, то я оберу останній варіатн та буду використовувати свій android девайс, тому що для одночасного запуску Android syudio та проекту потрібен досить потужний комп’ютер.

Для подальшої роботи потрібно встановити мобільний додаток Expo Go рис 3.1.6 від розробників Expo CLI, на який і буде виводитись результат розробки.

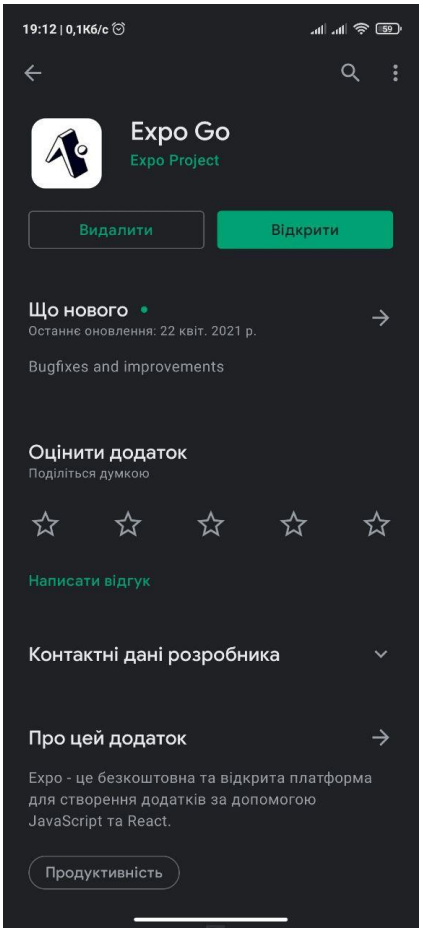


Рисунок 3.1.6 – Expo Go – мобільний застосунок від розробників Expo CLI

Після встановлення додатку нам необхідно зайти в нього обрати пункк – відстанувати QR-код та навести камеру на лівий нижній кут сторінки Expo Developer Tools рис 3.1.5. після чого відбудеться завантаження необхідних для старту застосунку файлів і по завершенню додаток відкриється.. На цьому розгортання проекту завершується і ми можемо приступити до розробки.

3.2. Діаграма використання

За для побудови хорошої архітектури та навігації проекту потрібно зробити діаграму варіантів використання застосунку. В таблиці табл 3.2.1 розписаний весь функціонал який буде доступний кінечному користувачеві.

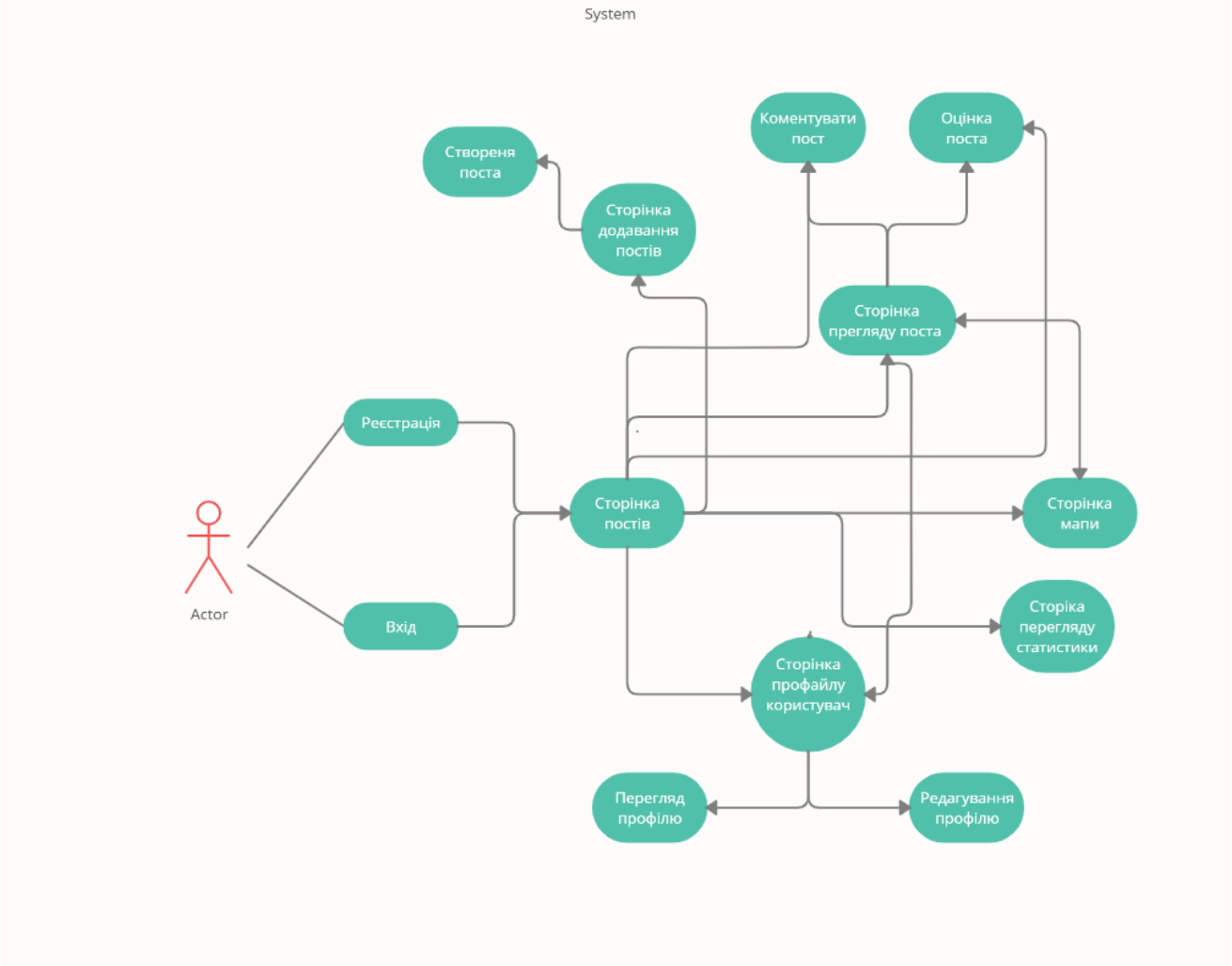


Рисунок 3.2.1 – Діаграма використання

3.3.Взаємодія з базою даних

В якості бекенд частини для розроби застосутку я буду використовувати Firebase.

3.3.1. Firebase

Firebase – Backend-as-a-Service – BaaS, який розпочинався як стартап YC11 і виріс в платформу розробки застосунків нового покоління на пратформі Google Cloud Platform[13].

Firebase дозволяє програмістам більш зосередитись на розробці клієнтської частини додатку, що дуже позитивно впливає на якість коду та враження кінцевого користувача. Вам не потрібно налаштовувати та керувати серверами і що саме головне, не потрібно писати API, томуще все це за вас виконує Firebase

Основні можливості:

– Робота в режимі реального часу (Realtime) – являється хмарною базою даних. Дані зберігаються в форматі JSON та синхронізуютсья в реальному часі з кожним підключенням користувача. Всі користувачі при підключенні діляться одним екземпляром бази даних Realtime і автоматично отримують оновлення з найновішими даними

– Доступ до Realtime Доступ можна отримати безпосередньо з мобільного пристоя або з веб-браузера, немає необхідності в написанні серверної частини окремо для кожного з випадків. Безпека та перевірка вхідних даних доступні в Realtime, дуже гнучнка мова правил на основі даних які виконуються для читання та запису.

– Firebase Cloud Messaging (FCM) – це кросплатформерне рішення для обміну повідомленнями за домомогою якого можна відправляти сповіщення на клієнтські пристрої[15] рис 3.3.1.

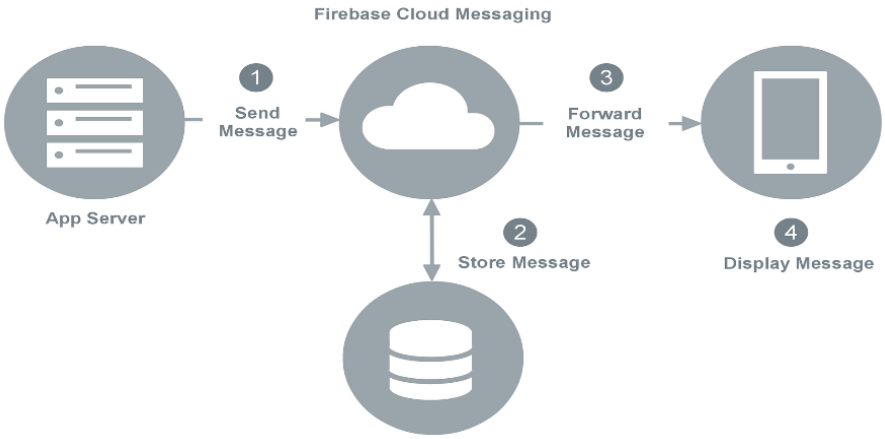


Рисунок 3.3.1 – Принцип роботи FCM

3.3.2. Підключення Firebase до проекту

Для початку роботи з Firebase потрібно зареєструватися на стайті платформи, перейти в консоль розробки та створити новий проект

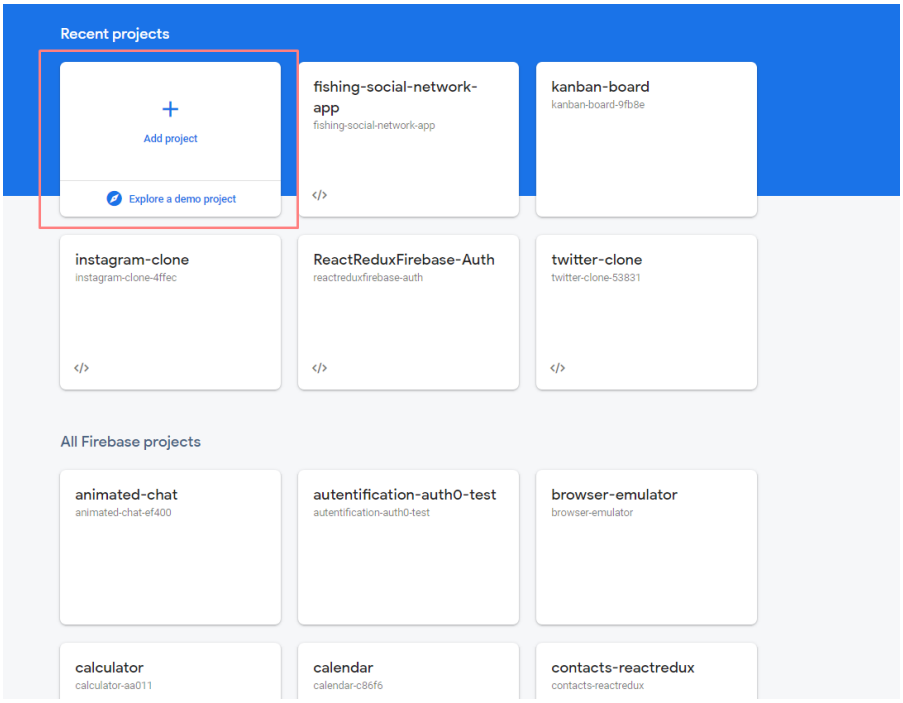


Рисунок 3.3.2 – Сторіка створення нового проекту в Firebase

Після створення проекту нам необхідно перейти в налаштування та скопіювати конфіг рис 3.3.2 з ключами для доступу.

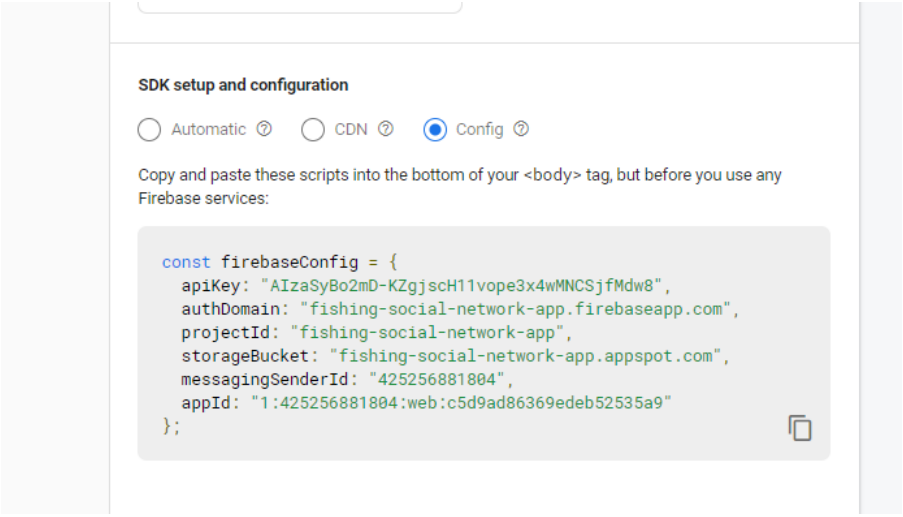


Рисунок 3.3.2 – Кофіг для отримання доступу до Firebae з клієнтської частини

Далі ми створюємо файл firebase-config рис 3.3.2 в якому й будуть зберігатися всі ключі доступу та налаштування Firebase

```
    REACT_APP_FIREBASE_STORAGE_BUCKET,
    REACT_APP_FIREBASE_MESSAGING_SENDER_ID,
    REACT_APP_FIREBASE_APP_ID
} from "@env"

import firebase from 'firebase/app';
import 'firebase/auth';
import 'firebase/storage';
import 'firebase/firestore';
  Krav, 05.02.2021 9:27 • add redux
let firebaseConfig = {
    apiKey: REACT_APP_FIREBASE_API_KEY,
    authDomain: REACT_APP_FIREBASE_AUTH_DOMAIN,
    projectId: REACT_APP_FIREBASE_PROJECT_ID,
    storageBucket: REACT_APP_FIREBASE_STORAGE_BUCKET,
    messagingSenderId: REACT_APP_FIREBASE_MESSAGING_SENDER_ID,
    appId: REACT_APP_FIREBASE_APP_ID
};

firebase.initializeApp(firebaseConfig);

export const projectStorage = firebase.storage();
export const projectAuth = firebase.auth();
export const projectFirestore = firebase.firestore()
export const timestamp = firebase.firestore.FieldValue.serverTimestamp();
export {firebase}
```

Рисунок 3.3.3 – Файл firebase-config

На рис 3.3.3 видно, що поля конфігу імпортується з файлу @env. Це розблено для безпеки даних. Env файл рис 3.3.4 буде доступний тільки для вас, а при використовуванні відкритих репозиторіїв на git hub цей файл додається в ігноруючі, топбно навіть в цьому випадку ніхто не отримає доступу до нього.

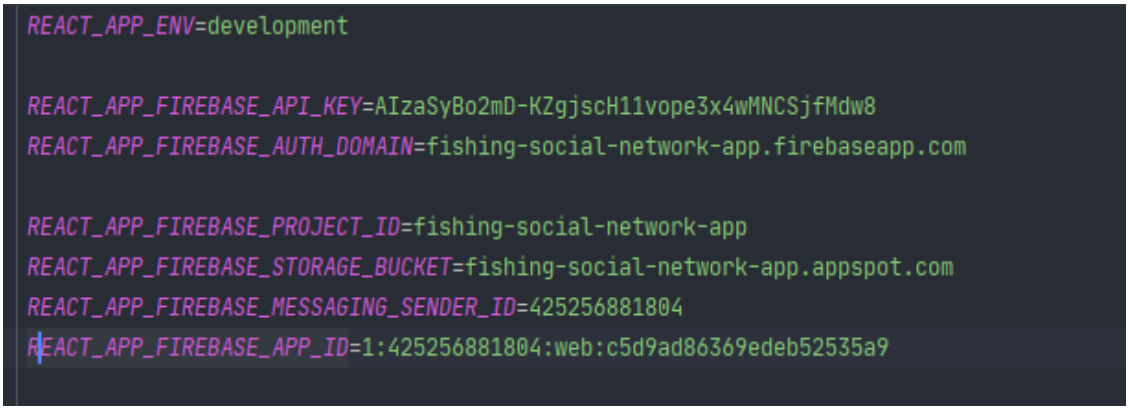


Рисунок 3.3.4 – Файл env.

3.3.3. Firebase Authentication

Firebase Authentication надає базові функції, прості у використанні SDK та готові бібліотеки користувацького інтерфейсу для аутентифікації користувачів у вашому застосунку[16]. Firebase Authentication підтрмує аутентифікаці з використанням:

– Електронної пошти та паролю
– Телефонного номену
– За домогою аккаунта Google
– Через аккаунт Facebook
– Через аккаунт Titter

Для використання Firebase Authentication в своєму застосунку потрібно перейти на сторіку створеного проекту в Firebase та там підключити блок аутентифікації.

3.3.4. Cloud Firestore

Cloud Firestore – це гнучка, легко маштабована хмарна база даних від Firebase та Google для веба, мобільних застосунків та серверних. Вона синхронізує ваші дані меж клієнтськими частинами застосунку за допомогою прослуховвування в реальному часі, а також надає підтримку офлайн режиму для мобільних платформ та веба[17]

Основні особливості Cloud Firestore:

– Гнучкість - модель данних Cloud Firestore підтримує гнучкі. Ієрархічні структури даних. Зберігає данів в документах які в свою чергу зберігаються в колекціях. Документи можуть мати вкладені обь'єкити та підколекції.

– Виразні запити – ви можете використовувати запити для отримання одного документу, певного документу або цілої колекції документів, які відповідають параметрам вашого запиту. Запити можуть містити параметри які складаються з одного чи декількох фільтрів. Також всі колекція за замовчуванням індексуються через це швидкодія запиту прямопропорційна швидкодії результату. Розроблений для масштабування – він використовує інфраструктуру Google Cloud Platform: автоматичну мультирегіональну розклдаку для збереження даних, надійні гарантіх цілісності, підтримку

Для початку роботи з Firestore потрібно перейти в потрібно перейти в секції Database та натиснути на Get Started для Cloud Firesotre, потім обрати начальний режим для правел безпеки.

За для подальшої роботи нам потрібно зробити схематичну структуру бази даних рис 3.3.5

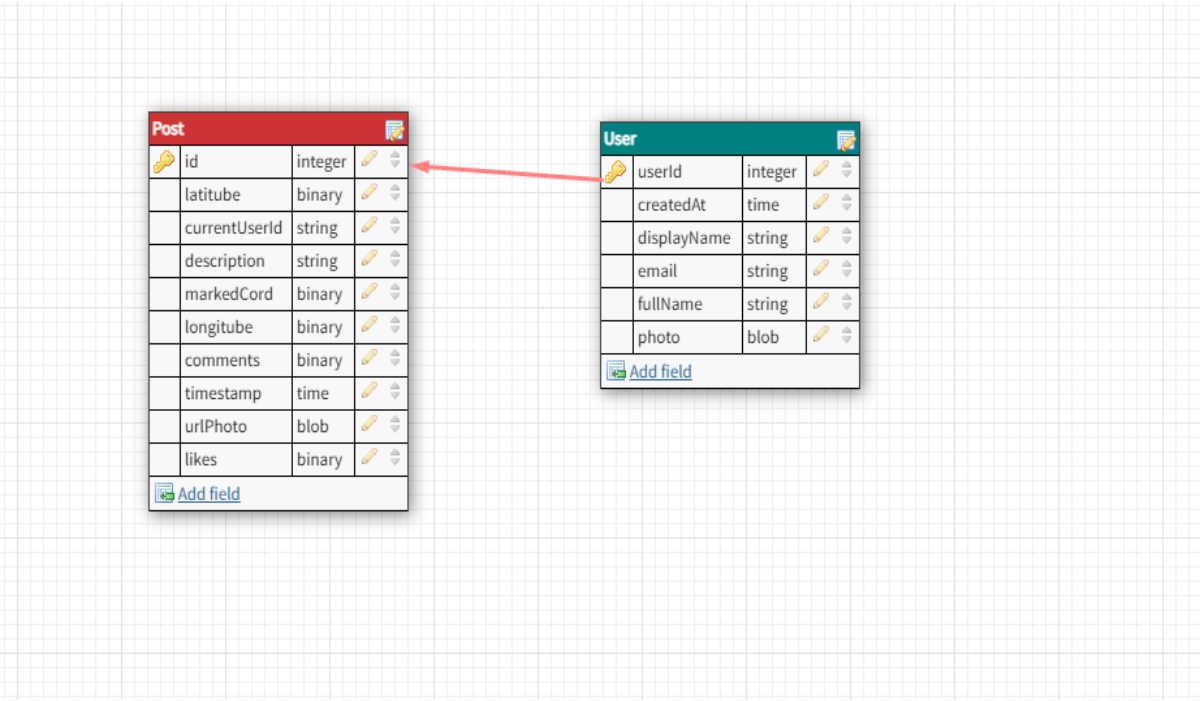


Рисунок 3.3.5 – Структура бази даних Firesore

3.4.Навігація в застосунку

Навігація в застосунку буде реалізована в двох варіантах – навігаційна панель внизу екрану (footer) та за допомогою бокового меню (sidebar).

Для кращого розуміння структури проекту потрбно скласти мапу навігіції по проекту.

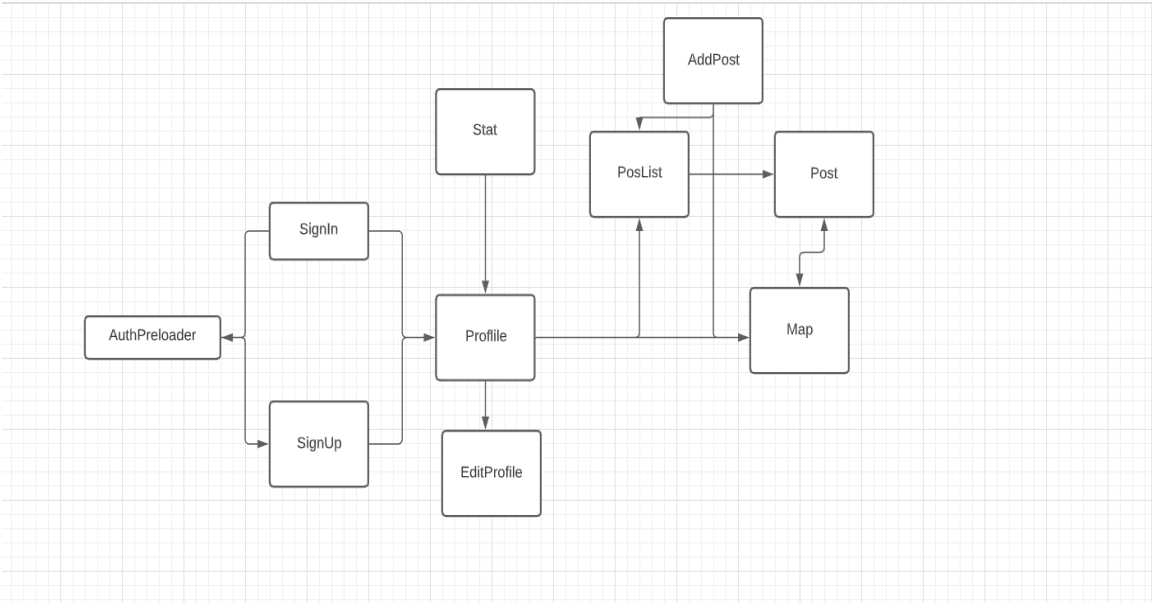


Рисунок 3.4.1 – Мапа навігації проекту

3.5.Опис компонентів

Так як застосунки на React розробляються компонентним підходом[18], то нам потрібно розбити весь функціонал по компонентах, це можна зробити виходячи з діаграми використання рис 3.2.1

– Header стр 3.5.1 – фукнціонал скдається з кнопки для відкриття богового меню(SideBar), назви сторінки на якій знаходимось, та кнопки back для переходу на головну сторінку.

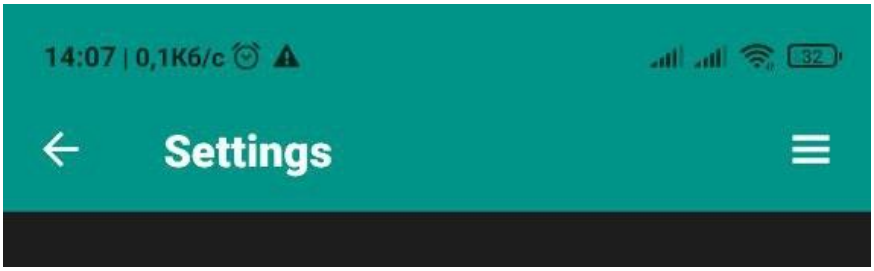


Рисунок 3.5.1 – Комопнент header

– Footer рис 3.5.2– цей компонент виступає в ролі панелі навігації по основних сторінках застосунку.

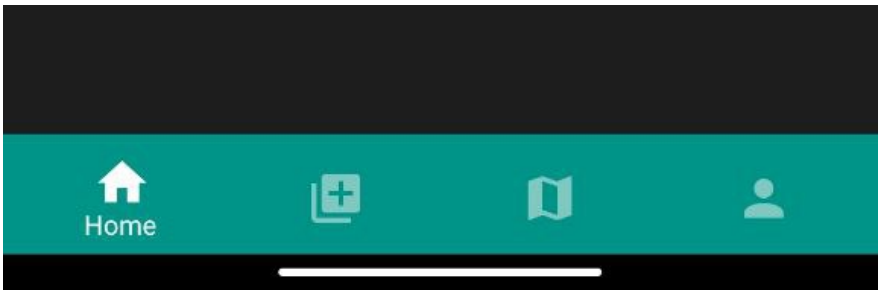


Рисунок 3.5.2 – Комопнент footer

– Sidebar рис 3.5.3 - компонент навігаційного меню. За допомогою нього і відбувається основна навігація по застосунку навігація Також він відповідає за вихід користувача з облікового запису

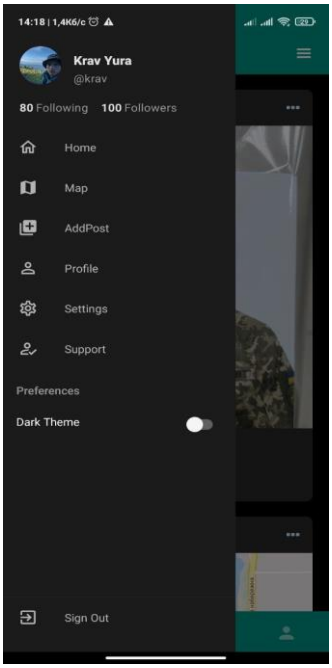


Рисунок 3.5.3 – Компонент Sidebar

– Map рис 3.5.3 - компонент відповідає за відображення позицій місць лову риболовів які створили пости, та для показу заборонених участків для будь якого виду лову.

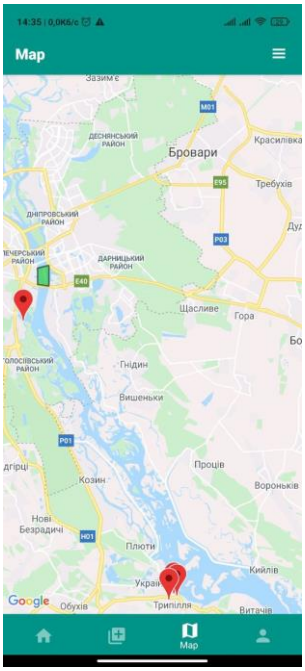


Рисунок 3.5.4 – Компонент Map

– Setting рис 3.5.5 - комопнент для зміни інформації про обліковий запис користувача

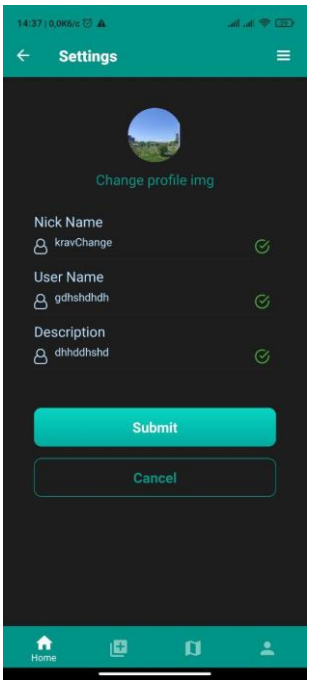


Рисунок 3.5.5 – Комопнент Settings

– SignIn рис 3.4.6 - компонент для авторизації користувача. Підтримуючись варіант входу – вхід за допомогою електронної адреси та паролю.

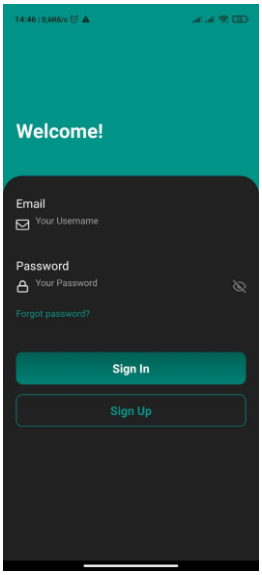


Рисунок 3.5.6 – Комопнент SignIn

– SignUp рис 3.5.7 - компонент для створення облікового запису користувача. За для реєстрації потрібно ввести нікнейм, електронну адресу та пароль. Також є поле підтвердження паролю, бо як показує практика користувач часто робить помилки при вводі паролю і після цього не може увійти в свій обліковий запис.

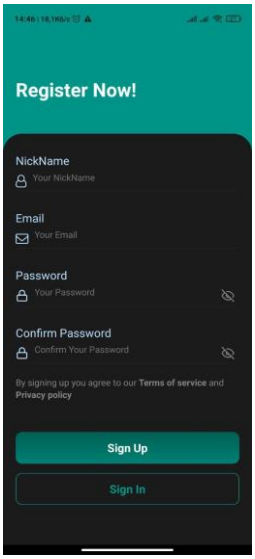


Рисунок 3.5.7 – Комопнент SignUp

– Profile рис 3.5.7 – компонент в якому відображається основна інформація про користувача та є список створених ним посів.

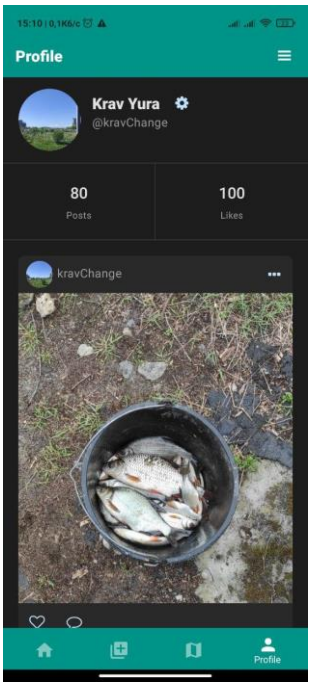


Рисунок 3.5.8 – Компонент Profile

– AddPost рис 3.5.9 – компонент відповідає за створення нового поста. Він включає в себе карту поле опису та поле для завантаження фотографії

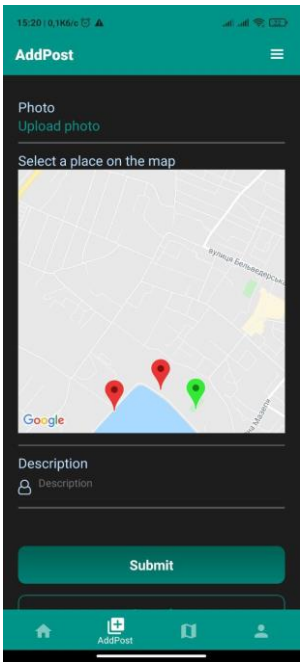


Рисунок 3.5.9 – Компонент AddPost

3.6. Висновки до розділу

Даний розділ був присвячений розробці застосунку. Був ініціалізований та налаштований проєкт, підключені всі необхідні бібліотеки, побудована діаграма використання та схема бази даних.

ВИСНОВКИ

В результаті виконання даної роботи був розроблений мобільний застосунок В результаті виконання даної роботи був розроблений мобільний застосунок

– Проведено аналіз існуючих аналогів та виявлені їх недоліки та переваги. На основі аналізу зроблено висновок, що розробка даного застосунку є ціклом оправдана та має сенс

– Був обраний оптимальний стек технологій для максимальної швидкодії застосунку.

– Проведено ініціалізацію та повне налаштування проекту для повноцінної роботи. Обрано та встановлено всі необхідні бібліотеки. Підключено хмарну базу даних Firestore та авторизацію за допомогою сервісу Frebase Autentication. Також було розроблено схему використання застосунку, мапу навігації та структуру бази даних.

Перспективи проєкту. Надалі планується додавання нового та покращення старого функціоналу розробленого застосунку. Ще згодом почнеться розробка застосунку під операційну системи IOS

СПИСОК ВИКОРАСТИНОЇ ЛІТЕРАТУРИ

1) What is Adaptive design [Електронний ресурс] – Режим доступу: https://www.interaction-design.org/literature/topics/adaptive-design (дата звернення 18.04.2021). – Назва з екрана.

2) What is progressive design [Електронний ресурс] – Режим доступу: https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps (дата звернення 22.04.2021). – Назва з екрана.

3) User Experience [Електронний ресурс] – Режим доступу: https://www.interaction-design.org/literature/topics/ux-design (дата звернення 29.04.2021). – Назва з екрана.

4) 5 levels design [Електронний ресурс] – Режим доступу: https://www.isixsigma.com/topic/full-factorial-design-with-2-factors-and-5-levels/ (дата звернення 22.04.2021). – Назва з екрана.

5) React documentation [Електронний ресурс] – Режим доступу: https://reactnative.dev/ (дата звернення 25.04.2021). – Назва з екрана.

6) Flutter documentation [Електронний ресурс] – Режим доступу: https://flutter.dev/ (дата звернення 26.04.2021). – Назва з екрана.

7) Xamarin [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/en-us/xamarin/android/ (дата звернення 29.04.2021). – Назва з екрана.

8) PhoneGap documentation [Електронний ресурс] – Режим доступу: http://docs.phonegap.com/ (дата звернення 26.04.2021). – Назва з екрана.

9) React Native CLI [Електронний ресурс] – Режим доступу: https://mohitaunni.medium.com/react-native-cli-explained-for-beginners-4725a271c30d (дата звернення 27.04.2021). – Назва з екрана.

10) Expo CLI [Електронний ресурс] – Режим доступу: https://docs.expo.io/workflow/expo-cli/ (дата звернення 27.04.2021). – Назва з екрана.

11) Flux documentation[Електронний ресурс] – Режим доступу: https://medium.com/@marina.kovalyova/flux-the-react-js-application-architecture-773f515d068d(дата звернення 2.04.2021). – Назва з екрана.

12) Redux documentation[Електронний ресурс] – Режим доступу: https://redux.js.org/faq/general#when-should-i-learn-redux (дата звернення 29.04.2021). – Назва з екрана.

13) Redux state tree [Електронний ресурс] – Режим доступу: https://redux.js.org/understanding/thinking-in-redux/three-principles (дата звернення 29.04.2021). – Назва з екрана.

14) Firebase documentation[Електронний ресурс] – Режим доступу: https://firebase.google.com/docs/guides (дата звернення 28.04.2021). – Назва з екрана.

15) Firebase Authentication [Електронний ресурс] – Режим доступу: https://firebase.google.com/docs/cloud-messaging (дата звернення 29.04.2021). – Назва з екрана.

16) Firebase Cloud messengin [Електронний ресурс] – Режим доступу: https://firebase.google.com/docs/auth?authuser=0 (дата звернення 29.04.2021). – Назва з екрана.

17) Cloud firestore [Електронний ресурс] – Режим доступу: https://firebase.google.com/docs/firestore (дата звернення 29.04.2021). – Назва з екрана.

18) React components principles [Електронний ресурс] – Режим доступу: https://ru.reactjs.org/docs/design-principles.html (дата звернення 29.04.2021). – Назва з екрана.

ДОДАТОК А

Дипломна робота

"СТВОРЕННЯ МОБІЛЬНОГО ЗАСТОСУНКУ ДЛЯ ВЕДЕННЯ СТАТИСТИКИ ПО ЛОВЛІ РИБИ НА ФРЕЙМВОРЦІ REACT NATIVE"

Виконав студент:
Кравченко Ю.В.
Керівник:
Гаманюк І.М

Київ 2021

Основні характеристики роботи

- Об'єкт дослідження – процес аналізу продуктивності риболова
- Предмет дослідження - аналіз продуктивності риболова за допомогою мобільного застосунку
- Мета роботи - створення мобільний застосунку для риболовів за допомогою фреймворку React Native.
- Методи дослідження - методи проектування архітектури проекту, методи проектування та розробки мобільних додатків, методи проектування та розробки користувацьких інтерфейсів.

Таблиця порівнять з аналогами

Найменування	Мій застосунок	Fishing Times Free	Блокнот риболова
Українська локалізація	+		
Взаємодія з картами	+	+	
Календар риболова		+	+
Статистика по ловлі риби	+	+	
Можливість поділитися інформацією по риболовлі з друзями	+		
Інформація що до зимувальних ям, та заборонених місць лову	+		

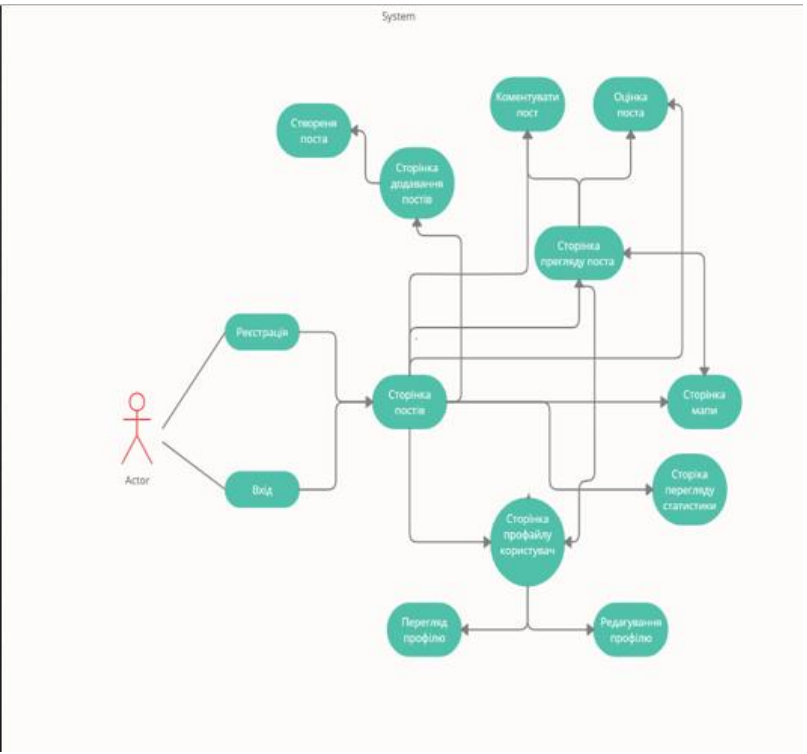
Таблиця порівнянь фреймворків для кросплатформерної розробки

Найменування	React Native	Flutter	Lonic	Xamarin	PhoneGap
Підтримка функції Hot Reloading	+	+	-	-	-
Мова програмування	JavaScript + React	Dart	JavaScript, HTML, CSS + Angular, React, Vue	C# + .NET	JavaScript, HTML, CSS
Спільнота розробників	Дуже велика	Не велика	Не велика	Не велика	Не велика
Відкритий код фрейворку	Так	Так	Так + платні пакети	Так + платні пакети	Так
Швидкодія	Висока, наближена до нативної	Висока, наближена до нативної	Середня	Не велика	Середня
Вбудовані компоненти	Так	Ні	Ні	Так	Ні

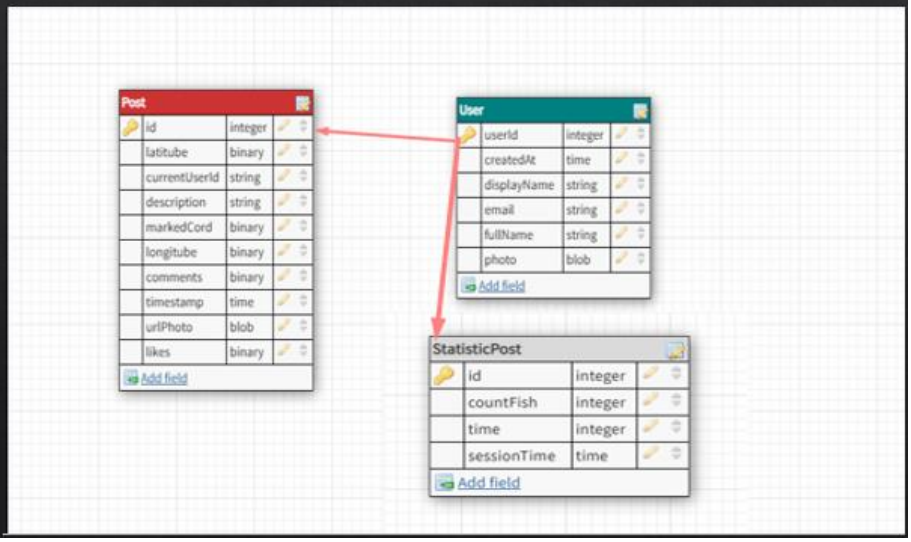
Таблиця порівнянь CLI

Найменування	Expo CLI	React-Native CLI
Можливість налаштувати проект за декілька хвилин	+	-
Можливість поділитися проектом за допомогою силки або QR-коду.	+	-
Вага тільки ініціалізованого застосунку	25-мб	7-мб
Одразу встановлені JS API, такі як : Push-сповіщення, Asset Manager	+	-
Є підтриика Bluetooth	-	+
Можливість використоувати властні модулі написані на Java/Objective-C.	-	+

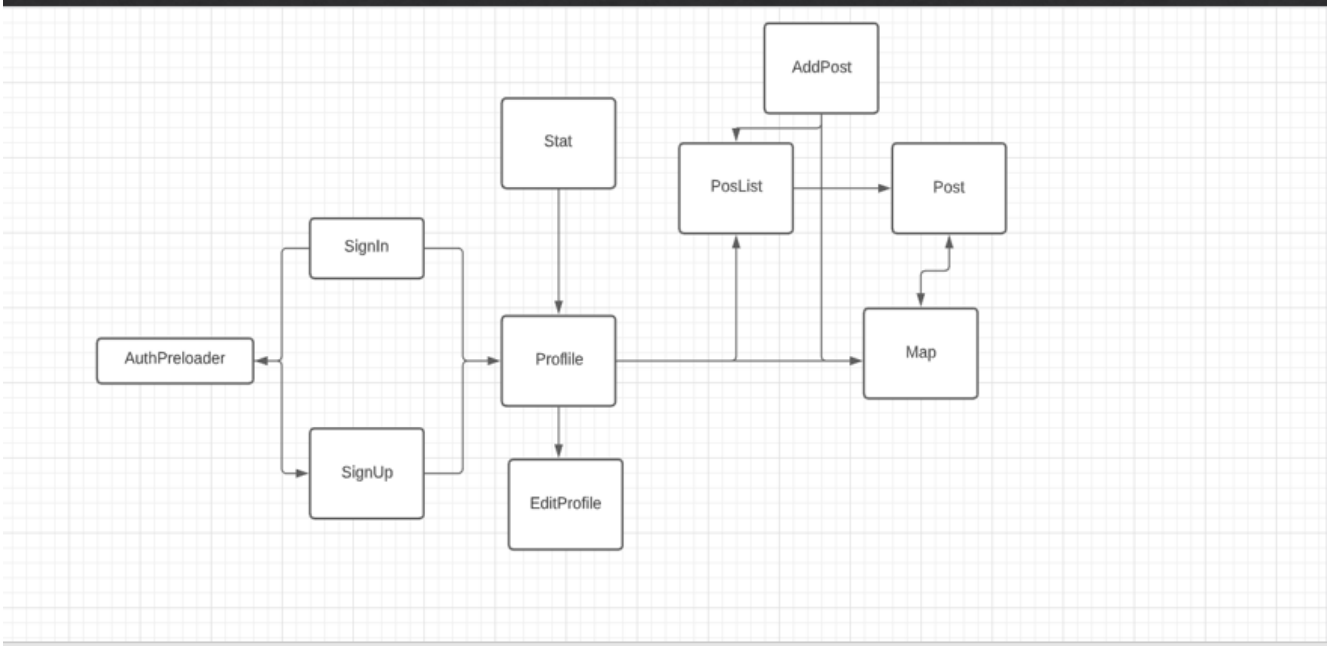
Діаграма використання



Структура бази даних



Мапа навігації



Висновки

- В результаті виконання даної роботи був розроблений мобільний застосунок за допомогою фреймворку React Native. В ході виконання були вершині такі задачі:
- Проведено аналіз існуючих аналогів та виявлені їх недоліки та переваги. На основі аналізу зроблено висновок, що розробка даного застосунку є циклом оправдана та має сенс.
- Був обраний оптимальний стек технологій для максимальної швидкодії застосунку.
- Проведено ініціалізацію та повне налащутвання проекту для повноцінної роботи. Обрано та встановлено всі необхідні бібліотеки. Підключено хмарну базу даних Firestore та авторизацію за допомогою сервісу Frebase Autentication. Також було розроблено схему використання застосунку, мапу навігації та структуру бази даних.
- **Перспективи проєкту.** Надалі планується додавання нового та покращення старого функціоналу розробленого щастосунку. Ще згодом планується розробка застосунку для операційної системи IOS.