**Пояснювальна записка  
до курсової роботи**

на тему: Мобільний застосунок «Метеорологічна мапа»

КПІ.ІТ-0115.045440.02.81

Київ – 2022

Зміст

[Перелік умовних позначень 4](#_Toc124366020)

[Вступ 5](#_Toc124366021)

[1 АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 6](#_Toc124366022)

[1.1 Загальні положення 6](#_Toc124366023)

[1.2 Змістовний опис і аналіз предметної області 8](#_Toc124366024)

[1.3 Аналіз існуючих технологій та успішних IT-проєктів 10](#_Toc124366025)

[1.3.1 Аналіз відомих алгоритмічних та технічних рішень 10](#_Toc124366026)

[1.3.2 Аналіз допоміжних програмних засобів та засобів розробки 11](#_Toc124366027)

[1.3.3 Аналіз відомих програмних продуктів 12](#_Toc124366028)

[1.4 Аналіз вимог до програмного забезпечення 15](#_Toc124366029)

[1.4.1 Розроблення функціональних вимог 19](#_Toc124366030)

[1.4.2 Розроблення нефункціональних вимог 22](#_Toc124366031)

[1.5 Постановка задачі 22](#_Toc124366032)

[Висновки до розділу 23](#_Toc124366033)

[2 МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 24](#_Toc124366034)

[2.1 Моделювання та аналіз програмного забезпечення 24](#_Toc124366035)

[2.2 Архітектура програмного забезпечення 25](#_Toc124366036)

[2.3 Конструювання програмного забезпечення 27](#_Toc124366037)

[Висновки до розділу 30](#_Toc124366038)

[3 АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 31](#_Toc124366039)

[3.1 Аналіз якості ПЗ 31](#_Toc124366040)

[3.2 Опис процесів тестування 31](#_Toc124366041)

[3.3 Опис контрольного прикладу 32](#_Toc124366042)

[Висновки до розділу 32](#_Toc124366043)

[4 ВПРОВАДЖЕННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 33](#_Toc124366044)

[4.1 Розгортання програмного забезпечення 33](#_Toc124366045)

[4.2 Підтримка програмного забезпечення 34](#_Toc124366046)

[Висновки до розділу 36](#_Toc124366047)

[ВИСНОВКИ 37](#_Toc124366048)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 38](#_Toc124366049)

Перелік умовних позначень

|  |  |  |
| --- | --- | --- |
| IDE | – | Integrated Development Environment – інтегроване середовище розробки. |
| API | – | Application programming interface, прикладний програмний Інтерфейс |
| SDK | – | Software development kit |
| IT | – | Інформаційні технології |
| ER | – | Entity-Relation diagram |
| OC | – | Операційна система. |
| БД | – | База даних. |

Вступ

Погодні умови є фактором, які достатньо сильно впливають на наше життя і тому є значна потреба в системах, що надають інформацію різного типу про погоду. Людство розробило безліч веб сайтів та мобільних застосунків метою яких є відображення різних аспектів погоди користувачам. Для звичайних подорожей, планування засівів, транспортних перевезень полів чи можливо навіть планування військових операцій виникла необхідність бачити погодну обстановку на карті в реальному часі.

Тому були розроблені системи, що відображають погоду на карті в реальному часі (наприклад: «Метео радар»), що показують дуже детальну інформацію про погоду на веб-сайтах та мобільних додатках.

На жаль, існуючі системи не надають інформацію про погоду за відсутності з`єднання Інтернет, що може стати серйозною проблемою якщо користувач є мандрівником і знаходиться десь в лісі й не має з’єднання або ж фермером в полі з відсутнім стабільним з’єднанням і не може оцінити погодну обстановку. В зв’язку з цим було вирішено створити мобільний застосунок «Метеорологічна мапа», що буде відображати дані про погоду та зберігати в сховище мобільного телефону користувача дані про погоду на 3-4 дні вперед. Таким чином, одного разу зайшовши в додаток зі з’єднанням Інтернет та проглянувши необхідні ділянки мапи, користувач ще декілька днів зможе використовувати додаток не маючи стабільного підключення. Для оновлення даних системі необхідно з’єднання Інтернет, бо очевидно, що дані через 4 дні після завантаження дані стануть неактуальними.

Дане покращення дає змогу користувачам отримувати необхідну інформацію без наявного підключення до мережі, що в свою чергу дозволяє використовувати додаток в подорожах чи виконанні професійних завдань в незаселених місцях без необхідної інфраструктури.

# АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Загальні положення

За дослідженнями Zippia, використання Мобільних телефонів зростає з кожним роком (Рис 1.1.1). За даними дослідження, в 2020 році 86% населення планети використовує мобільний телефон, а мобільний трафік вже перевищив трафік що генерують персональні комп’ютери [1].

Chart, bar chart

Description automatically generated

Рисунок 1.1.1 – Зростання використання смартфонів

Бум мобільних пристроїв пояснюється відносною дешевизною як мобільного інтернету, так і появою дешевих смартфонів, що є значно більш доступними ніж персональні комп’ютери. Не мало важливим фактором є можливість використання мобільних пристроїв фактично будь-де: в метро в дорозі на роботу, в пробках, вдома що загалом і пояснює неймовірну кількість згенерованого трафіку.

У зв’язку з популяризацією смартфонів мобільні застосунки стають все більш важливими факторами для розвитку бізнесу. З розвитком соціальних мереж та збільшенням мобільного трафіку зростає також і кількість рекламної інформації, що користувачі переглядають саме зі смартфонів, тому перше знайомство з брендом часто відбувається з мобільного телефону.

За прогнозами експертів Cisco, до 2023 року глобально буде завантажено майже 300 мільярдів застосунків [2], при чому найбільш популярними стануть соціальні мережі, мобільні ігри та бізнес застосунки, що свідчить про те що технології стрімко розвиваються і потребують якісної розробки нових застосунків та адаптації старих інформаційних систем для зручного користування їх на смартфонах.

Тому на разі ми можемо спостерігати активну розробку мобільних застосунків для інтернет магазинів, роботи з документами, мобільними іграми та іншими бізнес додатками. Навіть український уряд запросив розробку мобільного додатку «Дія», який зараз є одним з найбільш завантажуваним в Україні й дозволяє переглядати особисті документи й робити запити до різних державних установ.

У свою чергу бізнеси намагаються використовувати всі можливості смартфонів та компонентів, що вони містять: камеру, акселерометр, GPS модуль та інші. Великі маркети впроваджують системи “Scan and go”, що дозволяє користувачу сканувати товари в магазині та оплачувати покупки онлайн не відвідуючи касу. Схожа ситуація в індустрії мобільних ігор: очевидно що використання в грі акселерометру як наприклад руля для віртуальних автомобілів приваблює користувачів та дає більше трафіку, що для бізнесу означає більше прибутку.

При цьому непотрібно забувати про те що мобільні пристрої в більшості своїй не є надпотужними, тому розробка додатків вимагає обережного користування ресурсами та якісної оптимізації. То ж індустрія диктує свої вимоги до лаконічного та простого відображення інформації, а виконання складних ресурсо затратних процесів зазвичай переносять на сервери, що віддають результат користувачам через мережу Інтернет.

Не виключенням стали й інформаційні системи, що надають інформацію про погоду. Все більше ресурсів створюють мобільних додатків, щоб надавати користувачам погодні прогнози саме через смартфони, що й диктує індустрія. Різні компанії що надають інформацію про погоду створюють погодні застосунки з дизайнерськими інтерфейсами, покращують користувацькі інтерфейси для підвищення конкурентно здатності свої застосунків. Багато застосунків впроваджують використання GPS модулів для того, щоб користувач міг швидко отримувати погодну інформацію для свого місцезнаходження виконуючи при цьому мінімальну кількість дій.

Одним з останніх цікавих напрямків розвитку погодних застосунків стали метеорологічні мапи, що показують напрям вітру, температуру, інші погодні явища на мапах в режимі реального часу для вузько направленого користування як-то корегування курсу кораблів в обхід штормів та для загального користування для, наприклад планування подорожей.

Очевидно, що звичайним користувачам не дуже зручно і можливо навіть складно користуватися детальними метеорологічними мапами, тому створюються мобільні застосунки, що відображають спрощені варіанти метеорологічних мап на смартфонах.

## Змістовний опис і аналіз предметної області

Очевидно, що неможливо зробити погодний застосунок, що не використовує мережу Інтернет, адже в смартфонах не передбачено датчиків чи інших технічних засобів, що можуть прогнозувати погоду. Отже на разі всі погодні застосунки є клієнт серверними додатками, що отримують дані з певного серверу та відображають інформацію користувачу. Тобто для розробки Метеорологічного застосунку необхідно знайти API, або розробити самостійно, що буде надавати погодну інформацію.

Застосунки під ОС iOS можуть бути розроблені нативно, або ж кросплатформенно, але через особливості ОС та специфіки вимог до дизайну при достатньому бюджеті на розробку додатку віддається перевага розробки нативних застосунків.

Також для розробки необхідно розробити дизайн користувацького інтерфейсу, що буде якісно відображати необхідну інформацію та розробити систему аналітики для нього. Системи аналітики в мобільних застосунках дозволяють бачити які частини застосунку привертають більше уваги користувачів, а які навпаки відштовхують, що в результаті дає змогу покращувати малоефективні частини додатку та акцентуватися на сильних сторонах. При певній кількості доданих функціональностей всі наступні варто додавати використовуючи A/B тести, що засновані на вже існуючій системі аналітики, що також дозволить покращити враження користувача від додатку.

Очевидним недоліком клієнт серверних застосунків і погодних додатків зокрема є те, що вони не працюють без підключення до мережі Інтернет, адже без з’єднання неможливо отримати дані від API. При поганому підключенні, клієнт серверні застосунки працюють повільно, що значно знижує задоволеність користувачів від застосунку та відштовхує їх. Виправити це можна кешуванням інформації або ж збереженням деяких погодних даних наперед. Тобто якщо користувач при наявному інтернет з’єднанні завантажить дані про прогноз погоди на 5 діб, то застосунок зможе відображати інформацію про погоду ще 5 днів використовуючи тільки завантажену інформацію.

Для розробки мобільного застосунку «Метеорологічна мапа» було вирішено використовувати нативні для ОС iOS технології. У якості API було вибрано відкрите API погоди “Open Weather API”, що дозволяє отримати прогноз погоди на 5 днів кожні 3 години для різних локацій. Інформацію з цього API застосунок буде відображати користувачу, а також зберігати в локальному сховищі смартфону користувача, що дозволить якийсь час використовувати застосунок без підключення до мережі Інтернет.

## Аналіз існуючих технологій та успішних IT-проєктів

Проаналізуємо відоме на сьогодні алгоритмічне забезпечення у даній області та технічні рішення, що допоможуть у реалізації мобільного застосунку «Метеорологічна мапа». Далі будуть розглянуті допоміжні програмні засоби, засоби розробки та готові програмні рішення.

### Аналіз відомих алгоритмічних та технічних рішень

У застосунку не передбачається використання спеціалізованих алгоритмічних рішень.

Для ОС iOS застосунки можна розробляти нативними засобами або ж використовувати кросфплатформені рішення. Оскільки додаток передбачається лише для ОС iOS, то переваги кросфплатформених рішень, а саме швидкість розробки для різних ОС нівелюються. Тому, було вирішено розробляти мобільний застосунок використовуючи нативні технології: мову програмування Swift та бібліотеки побудови інтерфейсу, що надає корпорація Apple.

Застосунок має 2 екранні форми: «Метрологічна мапа», тому додаток буде поділено на 2 відповідних модулі, що мають реалізовувати певний патерн проектування та взаємодіяти між собою. З існуючих варіантів найпопулярніших патернів проектування мобільних застосунків під ОС iOS MVC, MVVM, MVP, VIPER найбільше підходить модифікований патерн MVVM, а саме: MVVM+R. VIPER точно не підходить, адже є занадто складним для додатку «Метеорологічна мапа». При цьому MVC теж не підійде в цьому випадку, бо компонент «Controller» в модулі «Метрологічна мапа» доведеться виконувати дії з роботою з виглядом самої карти й необхідно його максимально розвантажити. Точка входу даного цього застосунку є саме компонент «Controller», адже він відобразить карту і має сповістити інші компоненти застосунку про локацію відображену на карті, тому було обрано MVVM. Модифікація MVVM+R дозволяє перенести відповідальність за перехід на інші модулі на компонент «Router», а оскільки в даному додатку 2 модулі, то така модифікація є необхідною.

### Аналіз допоміжних програмних засобів та засобів розробки

Важливив є визначення SDK відображення мапи. На сьогодні для розробки застосунків з мапою для ОС iOS існує 2 найпопулярніших рішення: Google maps та Apple MapKit. Оскільки при нативній розробці для ОС iOS використання Apple MapKit не потребує завантаження додаткових залежностей, то йому й була віддана перевага.

Зазвичай в розробці застосунків під ОС iOS MVVM передбачає використання реактивної парадигми програмування, тому для цього було обрано найпопулярніше відкрите рішення – RxSwift SDK. Як таких конкурентів в цієї бібліотеки немає, тому і обирати між чим теж немає.

Для реалізації запитів на API було вирішено не використовувати нативну функціональність URLSession, а віддати перевагу бібліотеку Alamofire, що є абстракцією над URLSession і надає простіший інтерфейс використання та загалом зменшує кількість рядків коду написаного для кожного запиту.

Для збереження даних в локальному сховищі смартфону користувача необхідно обирати між двома «best-practice» рішеннями: нативним CoreData SDK та Realm SDK. Оскільки в додатку не передбачається вивантаження даних в iCloud, то все ж ми можемо розглянути Realm SDK як варіант для збереження даних. У багатьох порівняннях, і в «CoreData чи Realm» від Барта Якобса [3] зокрема, зазначається, що Realm SDK працює швидше ніж CoreData SDK за проведеними дослідженнями, тому обрано було саме Realm SDK.

Для розробки користувацького інтерфейсу було порівняно старіший фреймворк UIKit та новіший SwiftUI. За технічним завданням застосунок має підтримувати iOS 13, що в свою чергу озчанає, що для цієї ОС SwiftUI SDK буде доступний в найпершій версії та матиме урізаний функціонал в порівнянні з UIKit [4], тому обрано було саме UIKit.

Для розробки як IDE використовуватиметься ХСode, адже єдиний аналог AppCode, не підтримує функціональність «Storyboards», що унеможливить розробку.

### Аналіз відомих програмних продуктів

На всіх нових смартфонах iPhone встановлено нативний застосунок «Погода», що має функціональність метеорологічної мапи. Він відображає температуру, можна переглянути напрямок вітру на мапі, якість повітря (Рис. 1.3.1). Окрім цього в застосунку передбачено детальніший прогноз погоди для певних міст. Без з’єднання з мережею Інтернет застосунок не надає інформацію (Рис. 1.3.2)

Graphical user interface, application, map

Description automatically generated

Рисунок 1.3.1 – Вигляд мапи нативного застосунку «Погода»

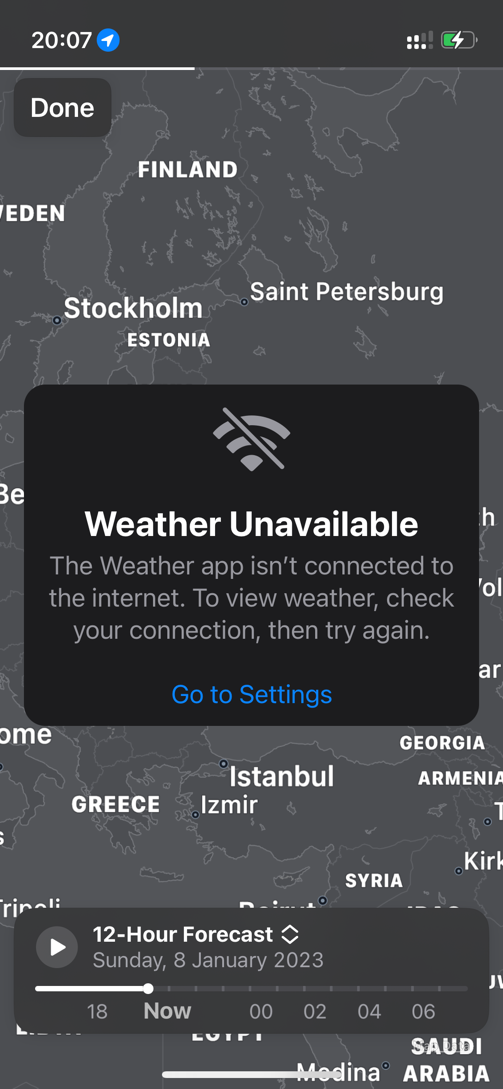


Рисунок 1.3.2 – Вигляд повідомлення про відсутність Інтернет з’єдання в нативному застосунку «Погода»

Вводячи в пошуково запиті App Store «Weather map» - перший застосунок, що запропонує платформа – “Weather Radar”. Застосунок надає детальну інформацію на мапі про різні аспекти погоди: рівень хмарності, температуру, напрям вітру (Рис. 1.3.3). При цьому оком видно піксельність карти, отже вона не нативна і виглядає не якісно. Взагалі можна зробити припущення, що застосунок є кросфплатформеним через нижчу якість зображень в ньому, ніж в тому ж нативному «Погода». Як і попередній додаток цей застосунок не працює взагалі без під’єднання до мережі Інтернет (Рис. 1.3.4).

Map

Description automatically generated

Рисунок 1.3.3 – Вигляд мапи застосунку «Weather Radar»

Logo, company name

Description automatically generated

Рисунок 1.3.4 – Вигляд повідомлення про відсутність Інтернет з’єдання в застосунку «Weather Radar»

В таблиці 1.3.1 порівняно функціональні можливості застосунків «Погода», «Weather Radar» та «Weather map».

Таблиця 1.3.1 – Порівняння з аналогом

|  |  |  |  |
| --- | --- | --- | --- |
| Функціонал | Weather map | Погода | Weather Radar |
| Відображення карти | Відображає якісну нативну карту | Відображає якісну нативну карту | Відображає неякісне кросплатформенне рішення |
| Відображення погодної інформації для міст | Відображає температуру та стан погоди для міст з населенням більше 50000 осіб. Не відображає температуру кольором для певної місцевості | Відображає погоду тільки для найбільших міст, але має змогу відображати температуру кольором на мапі | Відображає погоду тільки для міст що видно на карті, має змогу відображати температуру кольором на мапі |
| Прогноз погоди та детальна інформація | Відображає погоду для певного міста, надає прогноз погоди на 5 днів | Відображає погоду для певного міста, надає прогноз погоди на 10 днів | Відображає погоду для певного міста, надає прогноз погоди на 13 днів |
| Відображення інформації без підключення до Інтернет | Після останнього підключення до мережі має змогу відображати дані ще 3-4 дні | Відображає лише деталі погоди для місця локації користувача деякий проміжок часу | Не працює зовсім |

## Аналіз вимог до програмного забезпечення

Головною функцією програмного забезпечення є відображення метеорологічної мапи, більше функцій можна побачити на рисунку 1.4.1.

Diagram

Description automatically generated

Рисунок 1.4.1 – Діаграма варіантів використання

В таблицях 1.4.1 – 1.4.6 наведені варіанти використання програмного забезпечення.

Таблиця 1.4.1 – Варіант використання UC-1

|  |  |
| --- | --- |
| Use case name | Перегляд мапи |
| Use case ID | UC-01 |
| Goals | Отримання погодної інформації на певній місцевості |
| Actors | Користувач |
| Trigger | Користувач бажає отримати інформацію про погоду на місцевості |
| Pre-conditions | В додатку вимкнений режим технічного обслуговування. |
| Flow of Events | Користувач відкриває додаток та бачить мапу. Користувач скролить мапу для пошуку необхідної місцевості та переглядає погодну інформацію. |
| Extension | В випадку увімкнення режиму технічного обслуговування, користувач буде бачити повідомлення про це та не матиме змоги користуватися застосунком. |
| Post-Condition | Відкриття мапи, знаходження необхідної території. |

Таблиця 1.4.2 – Варіант використання UC-2

|  |  |
| --- | --- |
| Use case name | Перегляд прогнозу погоди |
| Use case ID | UC-02 |
| Goals | Отримання детальної інформації та прогнозу погоди для певного міста |
| Actors | Користувач |
| Trigger | Користувач бажає отримати інформацію про прогноз погоди для певного міста. |
| Pre-conditions | В додатку вимкнений режим технічного обслуговування. |
| Flow of Events | Користувач натискає на мітку на мапі, що відповідає обраному місту та має змогу побачити деталі про погоду та прогноз на декілька наступних днів. |
| Extension | В випадку увімкнення режиму технічного обслуговування, користувач буде бачити повідомлення про це та не матиме змоги користуватися застосунком. |
| Post-Condition | Відкриття екранної форми з прогнозом погоди. |

Таблиця 1.4.3 – Варіант використання UC-3

|  |  |
| --- | --- |
| Use case name | Відкриття веб сайту фонду «Повернись живим» |
| Use case ID | UC-03 |
| Goals | Переадресувати користувача на веб сайт фонду «Повернись живим» |
| Actors | Користувач |
| Trigger | Користувач бажає відкрити веб сайту фонду «Повернись живим». |
| Pre-conditions | В додатку вимкнений режим технічного обслуговування. |
| Flow of Events | Користувач відкриває прогноз погоди для певного міста. Користувач відкриває веб сайту фонду «Повернись живим» |
| Extension | В випадку увімкнення режиму технічного обслуговування, користувач буде бачити повідомлення про це та не матиме змоги користуватися застосунком. |
| Post-Condition | Переадресування користувача в браузер і відкриття там веб сайту фонду «Повернись живим». |

Таблиця 1.4.4 – Варіант використання UC-4

|  |  |
| --- | --- |
| Use case name | Увімкнення режиму технічного обслуговування |
| Use case ID | UC-04 |
| Goals | Увімкнути режим технічного обслуговування |
| Actors | Адміністратор |
| Trigger | Адміністратор вирішує увімкнути режим технічного обслуговування. |
| Pre-conditions | - |
| Flow of Events | Адміністратор в консолі Firebase в розділі «Remote config» вмикає режим технічного обслуговування. |
| Extension |  |
| Post-Condition | Користувачі не мають змоги користуватися застосунком та бачать повідомлення про увімкнений режим технічного обслуговування |

Таблиця 1.4.5 - Варіант використання UC-5

|  |  |
| --- | --- |
| Use case name | Вимкнення режиму технічного обслуговування |
| Use case ID | UC-05 |
| Goals | Вимкнути режим технічного обслуговування |
| Actors | Адміністратор |
| Trigger | Адміністратор вирішує вимкнути режим технічного обслуговування. |
| Pre-conditions | - |
| Flow of Events | Адміністратор в консолі Firebase в розділі «Remote config» вимикає режим технічного обслуговування. |
| Extension |  |
| Post-Condition | Користувачі мають змогу користуватися застосунком та не бачать повідомлення про увімкнений режим технічного обслуговування |

Таблиця 1.4.6 - Варіант використання UC-6

|  |  |
| --- | --- |
| Use case name | Перегляд аналітики використання застосунку |
| Use case ID | UC-06 |
| Goals | Перегляд аналітики використання застосунку |
| Actors | Адміністратор |
| Trigger | Адміністратор вирішує переглянути аналітику використання застосунку в консолі Firebase. |
| Pre-conditions | - |
| Flow of Events | Адміністратор в консолі Firebase в розділі «Analytics» переглядає кількість активних сесій, кількість надісланих повідомлень про різні дії користувача. |
| Extension |  |
| Post-Condition | Адміністратор отримав інформацію про те як користувачі використовують застосунок. |

### Розроблення функціональних вимог

Програмне забезпечення розділене на два модулі. Кожен модуль має свій певний набір функцій. У таблиці 1.4.6 наведено загальну модель вимог, а в таблицях 1.4.7 – 1.4.16 наведений опис функціональних вимог до програмного забезпечення. Матрицю трасування вимог можна побачити на рисунку 1.4.2.

Таблиця 1.4.6 – Модель вимог у загальному вигляді

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер | Опис | Код | Пріоритет | Ризик |
| 1 | Екранна форма відображення мапи | FR-1 | 1 | Високий |
| 1.1 | Відображення місцезнаходження користувача | FR-2 |  |  |
| 1.2 | Під час переміщення карти мають підвантажуватися дані про нові видимі міста та зберігатися в сховище смартфону | FR-3 |  |  |
| 2 | Екранна форма відображення деталей погоди | FR-4 | 4 | Низький |

Продовження таблиці 1.4.6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2.1 | Відображення банеру фонду «Повернись живим» | FR-5 |  |  |
| 3 | Відображення повідомлення про технічне обслуговування | FR-6 | 3 | Середній |
| 4 | Можливість увімкнення та вимкнення режиму технічного обслуговування в консолі Firebase | FR-7 | 2 | Низький |
| 5 | Можливість перегляду аналітики в консолі Firebase | FR-8 | 5 | Низький |
| 5.1 | Можливість перегляду кількості унікальних сесій в консолі Firebase | FR-9 |  |  |
| 5.1 | Можливість перегляду кількості запитів на погодне API, на API для міст, кількість відкриттів Екранної форми «Деталі погоди» в консолі Firebase | FR-10 |  |  |

Таблиця 1.4.7 – Функціональна вимога FR-1

|  |  |
| --- | --- |
| Назва | Екранна форма відображення мапи |
| Опис | Система повинна відображати мапу з метеорологічною інформацією. |

Таблиця 1.4.8 – Функціональна вимога FR-2

|  |  |
| --- | --- |
| Назва | Відображення місцезнаходження користувача |
| Опис | Система повинна відображати місцезнаходження користувача на мапі. |

Таблиця 1.4.9 – Функціональна вимога FR-3

|  |  |
| --- | --- |
| Назва | Під час переміщення карти мають підвантажуватися дані про нові видимі міста та зберігатися в сховище смартфону |

Таблиця 1.4.10 – Функціональна вимога FR-4

|  |  |
| --- | --- |
| Назва | Екранна форма відображення деталей погоди |
| Опис | Система повинна відображати деталі та прогноз погоди для певного міста. |

Таблиця 1.4.11 – Функціональна вимога FR-5

|  |  |
| --- | --- |
| Назва | Відображення банеру фонду «Повернись живим» |
| Опис | Система повинна відображати банер фонду «Повернись живим» та переадресовувати користувача на веб сайт фонду при натисненні на банер. |

Таблиця 1.4.12 – Функціональна вимога FR-6

|  |  |
| --- | --- |
| Назва | Відображення повідомлення про технічне обслуговування |

Таблиця 1.4.13 – Функціональна вимога FR-7

|  |  |
| --- | --- |
| Назва | Можливість увімкнення та вимкнення режиму технічного обслуговування в консолі Firebase |
| Опис | При увімкненні режиму технічного обслуговування застосунок не повинен робити будь-які API запити. |

Таблиця 1.4.14 – Функціональна вимога FR-8

|  |  |
| --- | --- |
| Назва | Можливість перегляду аналітики в консолі Firebase |

Таблиця 1.4.15 – Функціональна вимога FR-9

|  |  |
| --- | --- |
| Назва | Можливість перегляду кількості унікальних сесій в консолі Firebase |

Таблиця 1.4.16 – Функціональна вимога FR-10

|  |  |
| --- | --- |
| Назва | Можливість перегляду кількості запитів на погодне API, на API для міст, кількість відкриттів екрану «Деталі погоди» в консолі Firebase |

**A picture containing text, shoji, crossword puzzle

Description automatically generated**

Рисунок 1.4.2 – Матриця трасування вимог

### Розроблення нефункціональних вимог

У таблиці 1.4.17 наведено опис не функціональної вимоги до програмного забезпечення.

Таблиця 1.4.17

|  |  |
| --- | --- |
| Назва | Можливість бачити інформацію про погоду на мапі без підключення до мережі Інтернет. |
| Опис | Після перегляду даних на мапі зі стабільним підключенням до мережі Інтернет завантажені дані мають бути доступними ще 3 доби. |

## Постановка задачі

Враховуючи всі описані вимоги, необхідно розробити мобільний застосунок загального призначення для ОС iOS для перегляду погоди на мапі, з основною метою – планування маршруту подорожей чи місця відпочинку в залежності від погоди. Застосунок повинен відображати карту погоди, яка надає інформацію про стан погоди та температурний режим в містах з населенням більше 50 тисяч осіб. При виборі певного міста, користувач має отримати інформацію про температуру, швидкість вітру, вологість та прогноз погоди на майбутнє. Окрім цього, необхідно додати банер благодійного фонду «Повернись живим», натискання на який буде переадресовувати користувача в браузер та відкривати веб сайт фонду.

Важливою складовою є реалізація збереження даних в локальному сховищі пристрою користувача. При відсутності стабільного підключення до мережі Інтернет користувач має бачити збережену в локальному сховищі інформацію. З API застосунок матиме змогу отримувати інформацію про погодний прогноз з 5 днів вперед, тому через навіть 4 дні після останнього використання зі стабільним підключенням до мережі Інтернет застосунок матиме актуальні дані в сховищі для надання користувачу інформацію про погоду.

Для визначення ефективних векторів розвитку застосунку необхідно розробити систему аналітики на основі Firebase Analytics, що дозволить бачити які частини застосунку найбільше зацікавлюють користувача, а які навпаки відштовхують. Окрім того завдяки Firebase Crashlytics будуть відслідковуватися дефекти в програмному забезпеченні.

## Висновки до розділу

Отже, після аналізу предметної області та після аналізу застосунків, що є потенційними конкурентами було сформовано функціональні вимоги, які роблять застосунок “Weather map” конкурентоспроможним на ринку мобільних застосунків для ОС iOS. Одна з головних відмінностей: можливість роботи без підключення до мережі Інтернет 3 доби після останнього підключення.

Також було проаналізовано та обрано допоміжні програмні застосунки та бібліотеки що є найбільш ефективними для розробки саме цього застосунку.

# МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Моделювання та аналіз програмного забезпечення

Для опису бізнес процесу програмного забезпечення використовується BPMN модель (рисунок 2.1).

Diagram

Description automatically generated

Рисунок 2.1 – BPMN модель

Опис послідовності перегляду метеорологічної мапи:

* користувач відкриває застосунок;
* користувач скролить мапу в пошуках необхідної області;
* за потреби користувач обирає місто
* користувач переходить на екранну форму деталей та прогнозу погоди;
* користувач переглядає всю необхідну інформацію;
* користувач закриває екранну форму деталей та прогнозу погоди;
* користувач переглядає мапу;

## Архітектура програмного забезпечення

В попередніх пунктах було визначено, що додаток «Weather map» є клієнт серверним застосунком, що взаємодіє з двома зовнішніми API.

Для проектування самого застосунку було обрано патерн проектування MVVM+R, що містить такі компоненти:

* M – компонент «Model», містить бізнес логіку, роботу з API та внутрішнім сховищем смартфону користувача,
* V – компонент «View», що відповідає за відображення інформації користувачу, реакцію на дії користувача на вигляді застосунку,
* VM – компонент «ViewModel», що відповідає за підготовку інформації з API для компоненту «View», а також вирішує яку інформацію надати йому при різних діях користувача,
* R – компонент «Router», що відповідає за перехід на інші модулі застосунку.

Застосунок поділено на 2 модулі: один для відображення мапи, другий для відображення прогнозу погоди та детальної метеорологічної інформації.

Стек MVVM+R зберігається в контролері, тобто в компоненті «View», але посилання на «Model» сховане всередині компоненту «ViewModel».

Дані передаються між компонентами методами реактивного програмування, але екранна форма «Детальна інформація про погоду» є статичною і в модулі що за нею відповідає методи реактивного програмування не використовуються.

Для «спілкування» компонентів «ViewModel» та «View» використовується всього два «Observers»: один сповіщає про зміну центру видимої області на карті, інший про необхідність додавання на мапу нової інформації про погодні умови міста.

Компонент «ViewModel» містить посилання, отримує та аналізує інформацію компоненту «Model», що в свою чергу розділений на менеджер режиму технічного обслуговування та 2 сервіси:

* Сервіс отримання про міста для певної локації, що отримує дані від локального сховища та від API,
* Сервіс для отримання інформації про погоду, що отримує дані про погоду так само з локального сховища або зовнішнього API.

На рисунку 2.2 зображено детально архітектуру модулю «Метеорологічна мапа».

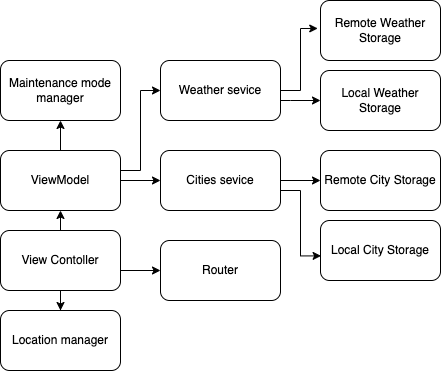


Рисунок 2.2 – Архітектура модулю «Метеорологічна мапа»

При відкритті модулю «Детальна інформація про погоду в місті» компонент «ViewModel» одразу надає інформацію компоненту «View» та більше вони не взаємодіють, оскільки екранна форма є статичною. Діаграму архітектури модулю «Детальна інформація про погоду в місті» відображена на рисунку 2.3.

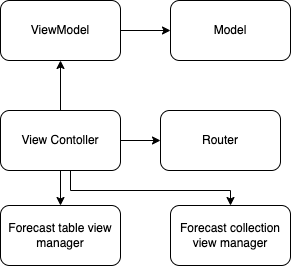


Рисунок 2.3 – Архітектура модулю «Детальна інформація про погоду в місті»

У компоненті «View» було також виділено окремі менеджери для роботи з таблицями, що відображають прогноз погоди з метою розвантаження самого контролеру вигляду.

## Конструювання програмного забезпечення

Спеціалізованих алгоритмів застосунок не містить.

Для реалізації реактивної парадигми програмування було використано RxSwift, що забезпечує потоко-безпечну роботу в цій парадигмі.

Для реалізації запитів до API використаний фреймворк Alamofire, що є абстракцією над нативним способом виконання запитів. Цей фреймворк надає функціонал що значно зменшує кількість написаного коду для кожного запиту. Для зручної роботи з JSON файлами, що застосунок отримує як відповідь від API було використано бібліотеку SwiftyJSON, що значно спрощує процес парсингу JSON відповідей.

Локальне сховище даних було реалізовано за допомогою бібліотеки RealmSwift, що використовує в якості системи управління базами даних MongoDB та надає зручний інтерфейс для роботи з таблицями і фільтрацією даних. В таблицях бази даних зберігається інформація про міста, про погоду та локації для яких ці дані були завантажені.

Опис таблиць бази даних наведено у таблицях 2.1 - 2.3. Модель бази даних наведена на рисунку 2.4.

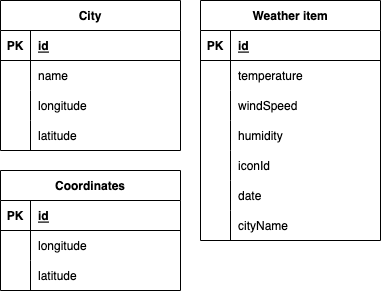


Рисунок 2.4 – Інформація про розгортання клієнту

Таблиця 2.1 – Опис таблиці Сity

|  |  |  |  |
| --- | --- | --- | --- |
| Таблиця | Назва поля | Тип даних | Опис |
| City | id | UUID | ідентифікаційний номер міста |
| name | String | назва міста |
| longitude | Double | довгота локації міста |
| latitude | Double | широта локації міста |

Таблиця 2.2 – Опис таблиці Coordinates

|  |  |  |  |
| --- | --- | --- | --- |
| Таблиця | Назва поля | Тип даних | Опис |
| Coordinates | id | UUID | ідентифікаційний номер міста |
| longitude | Double | довгота |
| latitude | Double | широта |

Таблиця 2.3 – Опис таблиці Weather item

|  |  |  |  |
| --- | --- | --- | --- |
| Таблиця | Назва поля | Тип даних | Опис |
| Weather item | id | UUID | ідентифікаційний номер міста |
| cityName | String | назва міста |
| temperature | Int | температура повітря |
| windSpeed | Double | швидкість вітру |
| humidity | Int | вологість повітря |
| iconId | Int | ідентифікаційний номер зображення що відповідає стану погоди |
| date | Double | дата за яку відображається погода в часовому інтервалі від 1970 року |

Опис утиліт, бібліотек та іншого стороннього програмного забезпечення, що використовується у розробці наведено в таблиці 2.5.

Таблиця 2.5 – Опис утиліт

|  |  |  |
| --- | --- | --- |
| №  п/п | Назва утиліти | Опис застосування |
| 1 | XCode | Головне середовище розробки програмного забезпечення курсової роботи. |
| 2 | Realm Studio | Програмне забезпечення яке надає легкий графічний інтерфейс для доступу до бази даних. |

## Висновки до розділу

Отже, в розділі було змодельовано архітектуру застосунку на основі патерну MVVM+R, описано необхідні допоміжні фреймворки для роботи з запитами до API, роботи з базою даних та реалізації реактивної парадигми програмування.

Окрім того було сконструйовано базу на даних для локального зберігання даних про погоду

Було обрано та описано утиліти для розробки: XCode як головна IDE та Realm studio для полегшення роботи з базою даних.

# АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Аналіз якості ПЗ

Програмне забезпечення мобільного застосунку «Weather map» було проаналізовано за допомогою сервісу Codacy [5]. Якість програмного забезпечення проаналізовано за трьома метриками (Рис. 3.1):

* відсоток проблем відносно очікуваного базового рівня;
* відсоток складних файлів у сховищі;
* відсоток дубльованих файлів у сховищі.

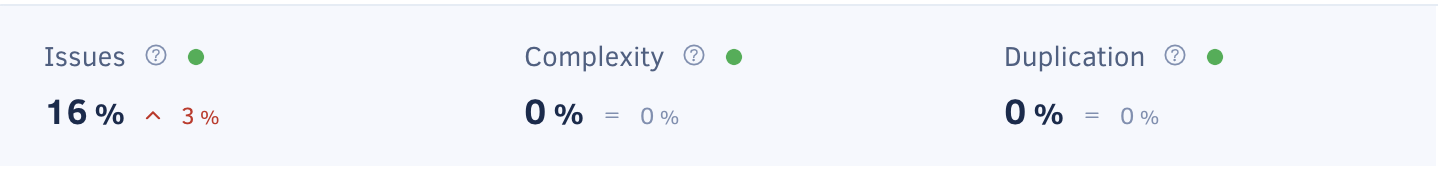


Рисунок 3.1 – Результати аналізу якості ПЗ.

У результаті аналізу було знайдено 76 проблеми в коді, з яких тільки 3 критичних з точки зору якості коду, але не для його дієздатності (Рис. 3.2).

Application

Description automatically generated with medium confidence

Рисунок 3.2 – Проблеми вихідного коду ПЗ.

## Опис процесів тестування

Було виконане мануальне тестування програмного забезпечення, опис відповідних тестів наведено у таблицях 3.1 – 3.7.

Таблиця 3.1 – Тест 1.1

|  |  |
| --- | --- |
| Тест | Відображення місця знаходження користувача |
| Модуль | Метеорологічна мапа |
| Номер тесту | 1.1 |
| Початковий стан системи | Користувач відкрив щойно встановлений застосунок. |
| Опис проведення тесту | При першому запуску застосунку має з’явитися запит до користувача про дозвіл на отримання місцезнаходження. Користувач дає дозвіл. |
| Очікуваний результат | Видима область мапи переміщується в місцезнаходження користувача, на карті місцезнаходження відображається. |
| Фактичний результат | Видима область мапи переміщується в місцезнаходження користувача, на карті місцезнаходження відображається. |

Таблиця 3.2 – Тест 1.2

|  |  |
| --- | --- |
| Тест | Приховання місця знаходження користувача |
| Модуль | Метеорологічна мапа |
| Номер тесту | 1.2 |
| Початковий стан системи | Користувач відкрив щойно встановлений застосунок. |
| Опис проведення тесту | При першому запуску застосунку має з’явитися запит до користувача про дозвіл на отримання місцезнаходження. Користувач не дає дозвіл. |
| Очікуваний результат | Видима область мапи не переміщується, на карті місцезнаходження не відображається. |
| Фактичний результат | Видима область мапи не переміщується, на карті місцезнаходження не відображається. |

Таблиця 3.3 – Тест 2.1

|  |  |
| --- | --- |
| Тест | Перегляд метеорологічної мапи |
| Модуль | Метеорологічна мапа |
| Номер тесту | 2.1 |
| Початковий стан системи | Користувач відкрив застосунок зі стабільним Інтернет з’єднанням. |
| Опис проведення тесту | Карта переміщується скролом користувача. |
| Очікуваний результат | На карті в місці великих міст з’являють відмітки з погодним станом та температурою в місті. |
| Фактичний результат | На карті в місці великих міст з’являють відмітки з погодним станом та температурою в місті. |

Таблиця 3.3 – Тест 2.2

|  |  |
| --- | --- |
| Тест | Перегляд метеорологічної мапи без з’єднання |
| Модуль | Метеорологічна мапа |
| Номер тесту | 2.1 |
| Початковий стан системи | Користувач відкрив застосунок без Інтернет з’єднанням після відкриття його зі стабільним Інтернет з’єднанням. |
| Опис проведення тесту | Карта переміщується скролом користувача. |
| Очікуваний результат | На карті в місцях великих міст, що попередньо переглянуті користувачем, з’являють відмітки з погодним станом та температурою в місті. |
| Фактичний результат | На карті в місцях великих міст, що попередньо переглянуті користувачем, з’являють відмітки з погодним станом та температурою в місті. |

Таблиця 3.4 – Тест 3.1

|  |  |
| --- | --- |
| Тест | Перегляд деталей погоди |
| Модуль | Деталі погоди |
| Номер тесту | 3.1 |
| Початковий стан системи | Користувач відкрив застосунок зі стабільним Інтернет з’єднанням. |
| Опис проведення тесту | Користувач натискає на будь-яку помітку погоди на мапі. |
| Очікуваний результат | Відкривається екранна форма з деталями погоди, на якій коректно вказано назву міста, відображається температура, вологість повітря, швидкість вітру, прогноз погоди 4 рази на кожні 3 години вперед та прогноз для 5 днів вперед. |
| Фактичний результат | Відкривається екранна форма з деталями погоди, на якій коректно вказано назву міста, відображається температура, вологість повітря, швидкість вітру, прогноз погоди 4 рази на кожні 3 години вперед та прогноз для 5 днів вперед. |

Таблиця 3.5 – Тест 3.2

|  |  |
| --- | --- |
| Тест | Перегляд деталей погоди без Інтернет з’єднання |
| Модуль | Деталі погоди |
| Номер тесту | 3.2 |
| Початковий стан системи | Користувач відкрив застосунок без Інтернет з’єднанням після відкриття його зі стабільним Інтернет з’єднанням. |
| Опис проведення тесту | Користувач натискає на будь-яку помітку погоди на мапі. |
| Очікуваний результат | Відкривається екранна форма з деталями погоди, на якій коректно вказано назву міста, відображається температура, вологість повітря, швидкість вітру, прогноз погоди 4 рази на кожні 3 години вперед та прогноз для 5 днів вперед. |
| Фактичний результат | Відкривається екранна форма з деталями погоди, на якій коректно вказано назву міста, відображається температура, вологість повітря, швидкість вітру, прогноз погоди 4 рази на кожні 3 години вперед та прогноз для 5 днів вперед. |

Таблиця 3.6 – Тест 4.1

|  |  |
| --- | --- |
| Тест | Перегляд деталей погоди без Інтернет з’єднання через 4 години |
| Модуль | Деталі погоди |
| Номер тесту | 4.1 |
| Початковий стан системи | Користувач відкрив застосунок без Інтернет з’єднанням через 4 години відкриття його зі стабільним Інтернет з’єднанням. |
| Опис проведення тесту | Користувач натискає на будь-яку помітку погоди на мапі. |
| Очікуваний результат | Відкривається екранна форма з деталями погоди, на якій коректно вказано назву міста, відображається температура, вологість повітря, швидкість вітру, прогноз погоди 4 рази на кожні 3 години вперед та прогноз для менше ніж 5 днів вперед. |
| Фактичний результат | Відкривається екранна форма з деталями погоди, на якій коректно вказано назву міста, відображається температура, вологість повітря, швидкість вітру, прогноз погоди 4 рази на кожні 3 години вперед та прогноз для менше ніж 5 днів вперед. |

Таблиця 3.7 – Тест 4.2

|  |  |
| --- | --- |
| Тест | Перегляд деталей погоди з Інтернет з’єднання через 4 години |
| Модуль | Деталі погоди |
| Номер тесту | 4.2 |
| Початковий стан системи | Користувач відкрив застосунок без Інтернет з’єднанням через 4 години відкриття його зі стабільним Інтернет з’єднанням. |
| Опис проведення тесту | Користувач натискає на будь-яку помітку погоди на мапі. |
| Очікуваний результат | Відкривається екранна форма з деталями погоди, на якій коректно вказано назву міста, відображається температура, вологість повітря, швидкість вітру, прогноз погоди 4 рази на кожні 3 години вперед та прогноз для менше ніж 5 днів вперед. |
| Фактичний результат | Відкривається екранна форма з деталями погоди, на якій коректно вказано назву міста, відображається температура, вологість повітря, швидкість вітру, прогноз погоди 4 рази на кожні 3 години вперед та прогноз для 5 днів вперед. |

## Висновки до розділу

Отже, у розділі було проаналізовано якість програмного забезпечення та не знайдено проблем чи помилок, що могли б критично вплинути на роботу застосунку.

Також було розроблено список тестів мануального тестування застосунку для того, щоб переконатися у відповідності програмного забезпечення

# ВПРОВАДЖЕННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Розгортання програмного забезпечення

На разі єдина можливість доставити застосунок до юзерів в ОС iOS є App Store.

Для експорту застосунку в AppStore необхідно експортувати його спочатку в Apple Store Connect. Застосунок «Weather Map» не використовує жодних критичних ресурсів чи сервісів смартфону, тому не потребує спеціальних сертифікатів від Apple для експорту в AppStore. Через це застосунок скомпільовану версію застосунку було експортовано в App Store Connect та підписано на звичайні «Signing» сертифікати автоматично засобами архівування застосунків в IDE XCode.

Після експорту скомпільовану застосунку Apple вимагає заповнити певну інформацію про нього:

* контактна інформація розробника,
* контакти служби підтримки,
* підтвердження, що в застосунку немає сцен насилля чи сексуального характеру для визначення мінімального віку користувачів, що можуть користуватися додатком,
* інформацію про використання персональних даних, де було вказано, що застосунок не зберігає жодних персональних даних користувача,
* скріншоти застосунку, що будуть відображатися на його сторінці в AppStore.

Додавши всю необхідну інформацію застосунок було відправлено на перевірку Apple, яку додаток пройшов підтвердивши цим, що не порушує правил майданчику AppStore.

Для можливості проведення тестування зовнішньою командою розробників також було погоджено експорт додатку на платформу TestFlight.

На разі застосунок можна знайти в AppStore за посиланням: <https://apps.apple.com/app/oleski-weather-map/id1663626101>. Вигляд сторінки продажу додатку на AppStore зображено на рисунку 4.1.

Graphical user interface, application

Description automatically generated

Рисунок 4.1 – Інформація про розгортання клієнту

Для тестування застосунку через TestFlight необхідно перейти за посиланням <https://testflight.apple.com/join/hmkdRN7P> та підтвердити згоду на приєднання до зовнішньої команди тестування додатку «Weather Map».

## Підтримка програмного забезпечення

Кожна нова версія застосунку буде так само експортуватися в AppStore й користувач зможе оновити застосунок самостійно вручну. Якщо ж у користувача на смартфоні увімкнено налаштування «Автоматичне оновлення», то застосунок оновиться до нової версії автоматично.

Для загальної підтримки реалізовано систему аналітики та крашлітики, яку можна переглянути в консолі Firebase, що відображена на рисунках 4.2 та 4.3 відповідно.

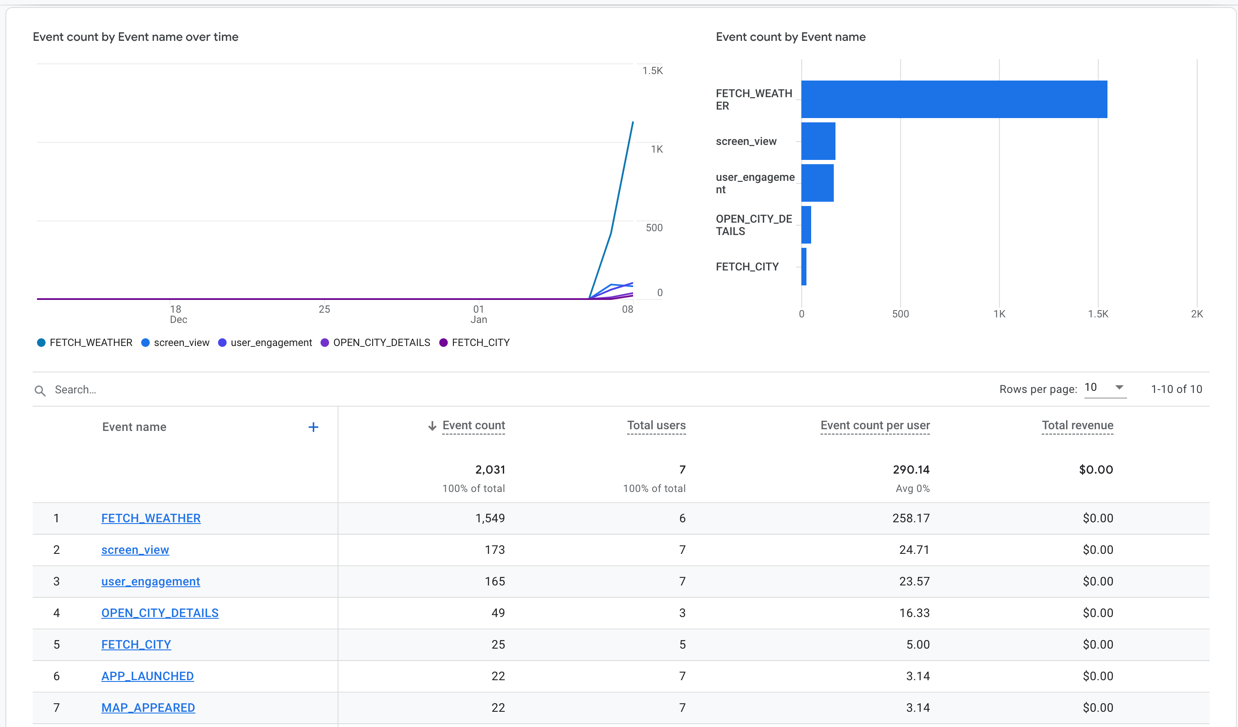


Рисунок 4.2 – Аналітика на консолі Firebase.

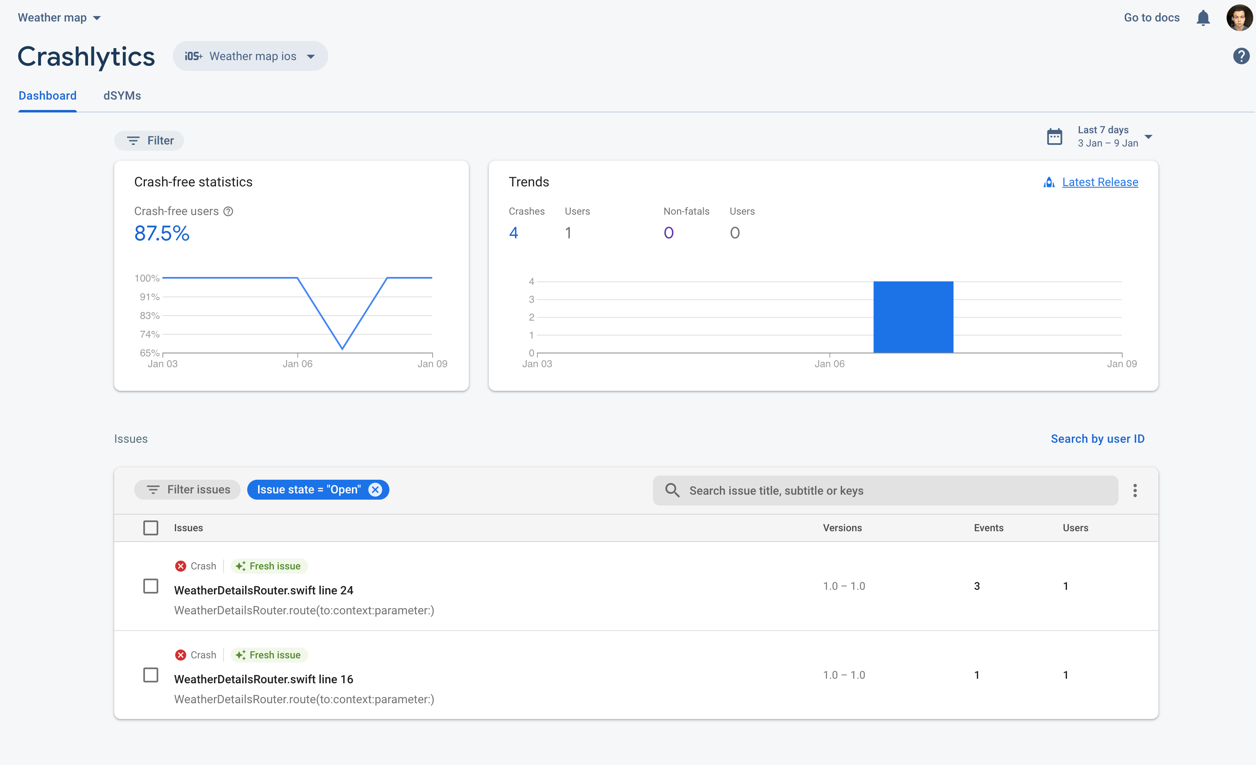


Рисунок 4.3 – Крашлітика на консолі Firebase.

На рисунку 4.3 відображається, що 12.5% користувачів зіткнулися з крашем застосунку, але насправді це тестові демонстраційні краші.

У разі виникнення непередбачуваних ситуацій застосунок має можливість перейти в режим технічного обслуговування та унеможливить запити на API, що можна зробити на консолі Firebase в розділі «Remote Config», що відображено на рисунку 4.4.

Graphical user interface, application

Description automatically generated

Рисунок 4.4 – Розділ «Remote Config» на консолі Firebase.

## Висновки до розділу

Отже, застосунок було розгорнуто на стандартному майданчику AppStore, та на платформі для тестування застосунків від Apple – TestFlight. Для цього було заповнено певну юридично необхідну інформацію за правилами маркетплейсу Apple.

Також для підтримки застосунку було організовано систему аналітики, за якою можна буде спостерігати стабільність роботи застосунку.

Доступ до застосунку в AppStore можна отримати за посиланням: <https://apps.apple.com/app/oleski-weather-map/id1663626101>.

Доступ до застосунку в Testflight можна отримати за посиланням: <https://testflight.apple.com/join/hmkdRN7P>.

# ВИСНОВКИ

Отже, було розроблено мобільний застосунок під OC iOS «Weather map», який надає користувачу мапу з відображенням стану погоди та температури в великих містах, надає можливість перегляду прогнозу погоди на декілька днів та інших деталей погоди: швидкість вітру, вологість, температура. Особливістю застосунку є те, що додаток зберігає погодні дані про переглянуті міста. Інформацію про них можна буде переглянути пізніше без стабільного підключення до мережі Інтернет.

У результаті виконання курсової роботи було спроєктовано мобільний застосунок на основі архітектурного патерну MVVM+R.

В якості середовища розробки обрано XCode.

У якості БД використано MongoDB, для полегшення роботи з якою використовувався фреймворк RealmSwift.

Після реалізації застосунку, він був протестований на смартфонах iPhone з різними версіями ОС iOS, з різними розмірами екранів, щоб переконатися, що додаток коректно відображається на різних пристроях.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Флін Д. Статистика використання смартфонів в США в 2020 [Електронний ресурс] / Джек Флін – Режим доступу до ресурсу: <https://www.zippia.com/advice/smartphone-usage-statistics/>
2. Річний звіт Cisco про Інтернет [Електронний ресурс] – Режим доступу до ресурсу: <https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html>.
3. Барт Я. CoreData чи Realm [Електронний ресурс] / Якобс Барт – Режим доступу до ресурсу: <https://cocoacasts.com/core-data-or-realm>.
4. Баддер К. SwiftUI в 2021 [Електронний ресурс] / Кріс Баддер – Режим доступу до ресурсу: <https://betterprogramming.pub/swiftui-2021-the-good-the-bad-and-the-ugly-458c6ee768f9>.
5. Сервіс Codacy [Електронний ресурс] – Режим доступу до ресурсу: <https://app.codacy.com/app>.