МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«МОБІЛЬНИЙ ДОДАТОК ПРОГНОЗУ ПОГОДИ НА ОСНОВІ ІНТЕРПОЛЯЦІЇ ДАНИХ З OPENWEATHER»**

Виконав: студент 2 курсу, групи 8.1219

спеціальності 121 інженерія програмного забезпечення
(шифр і назва напряму підготовки)

освітньої програми інженерія програмного забезпечення
(назва освітньої програми)

О.А. Журомський
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, доцент, к.ф.-м.н. Горбенко В.І.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент завідувач кафедри фундаментальної математики, доцент, д.т.н. Гребенюк. С.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти магістр

Спеціальність 121 інженерія програмного забезпечення

Освітня програма інженерія програмного забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

« 20 » травня 2020 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ(СТУДЕНТЦІ)

Журомському Олексію Андрійовичу
(прізвище, ім'я та по-батькові)

1. Тема роботи (проекту) Мобільний додаток прогнозу погоди на основі інтерполяції даних з OpenWeather

керівник роботи (проекту) Горбенко Віталій Іванович, к.ф.-м.н., доцент
(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 20 » травня 2020 року № 576 - с

2. Строк подання студентом роботи 30.11.2020

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Реалізація програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Презентація

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 20.05.2020

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	01.09.2020	
2	Збір вихідних даних.	10.09.2020	
3.	Обробка методичних та теоретичних	08.10.2020	
	джерел.		
4.	Розробка першого розділу.	20.10.2020	
5.	Розробка другого розділу.	22.11.2020	
6.	Розробка третього розділу.	22.11.2020	
7.	Оформлення та нормоконтроль	30.11.2020	
	кваліфікаційної роботи		
8.	Захист кваліфікаційної роботи.	15.12.2020	

Студент _____ О.А. Журомський
(підпис) (ініціали та прізвище)

Керівник роботи _____ В.І. Горбенко
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.В. Кудін
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Мобільний додаток прогнозу погоди на основі інтерполяції даних з OpenWeather»: 49 с., 30 рис., табл., 7 джерел, 4 додатки.

ПРОГНОЗУВАННЯ ПОГОДИ, МЕТЕОДАНІ, API-КЛЮЧ, ІНТЕРПОЛЯЦІЯ ПОГОДНИХ ПОКАЗНИКІВ, МЕТОДИ ІНТЕРПОЛЯЦІЇ ПОГОДНИХ ДАНИХ, OPENWEATHER.

Об’єкт дослідження – методи прогнозування погоди та процес прогнозування метеоданих за API-ключем.

Мета роботи: дослідити сучасні мобільні додатки з прогнозування погоди та розглянути можливість інтерполяції метеоданих.

Метод дослідження – аналітичний, порівняльний.

У останні роки прогнозування погоди є досить популярним через використання нових технологій та зростання кількості метеостанцій, що дозволяє використовувати більший обсяг метеоданих при складання кінцевого прогнозу. Для можливості отримувати більш точний результат, в роботі розглянуто можливості API сервісу OpenWeather та його вбудована функція інтерполяції метеоданих, а також метод використання різних API- ключів. Також були представлений приклад створення власної метеостанції на основі мікрокомп'ютерів серії Raspberry на випадок відсутності метеостанції в області дослідження. Даний метод дозволив поліпшити якість точності обчислень погодних даних.

SUMMARY

Master's Qualification Thesis «Based on OpenWeather Data Interpolation Mobile Application for Weather Forecasting»: 49 pages, 30 figures, 1 table, 7 references, 4 supplements.

WEATHER FORECASTING, METODATA, API KEY, WEATHER INTERPOLATION, WEATHER DATA INTERPOLATION METHODS, OPENWEATHER.

The object of research – methods of weather forecasting and the process of forecasting meteorological data by API-key.

The aim of the study is to explore modern mobile weather forecasting applications and consider interpolating meteorological data.

The methods of research are analytical, comparative.

In recent years, weather forecasting has become quite popular due to the use of new technologies and the growing number of weather stations, which allows the use of more meteorological data in the final forecast. To be able to obtain a more accurate result, the paper discusses the capabilities of the API service OpenWeather and its built-in meteorological interpolation function, as well as the method of using different API-keys. An example of creating your own weather station based on Raspberry series microcomputers were presented in a case of no weather station in the study area. This method has improved the quality of the accuracy of weather data calculations.

ЗМІСТ

ВСТУП

Тема даної роботи присвячена прогнозуванню погоди і розробці нового додатку з прогнозування погоди . При аналізі результатів сучасних додатків, заснованих на спрощених алгоритмах, доволі часто виникають суттєві відмінності та відхилення в точності обчислення шести основних погодних показників – таких як хмарність, кількість атмосферних опадів, сила вітру, температура, вологість повітря і атмосферний тиск (в цілому шість основних критеріїв для прогнозу загального користування).

Розробниками даних додатків було зроблено багато спроб щодо поліпшення якості прогнозування погодних даних, але, на жаль, успіху в виконанні цього завдання не було досягнуто. В данній роботі запропоновано новий варіант для прогнозування погодних даних за допомогою інтерполяції результатів обчислення погодних показників, пізніше реалізований в створеному мобільному додатку.

Запропонований в роботі метод не є інноваційним − отримані дані будуть інтерпольовані та наближені більш до дійсних показників. Також розглянуто і можливість використання API-різних ключів для перевірки отримання метеоданих з різних API-сервісів.

Також в роботі розглянутий спосіб віддаленого отримання погодних даних з власної сконструйованої метеостанції як можливе розв'язання проблеми відсутності найближчих метеостанцій на розглянутій місцевості.

РОЗДІЛ 1 РОЗКРИТТЯ ПРОБЛЕМАТИКИ В ПРОГНОЗУВАННІ ПОГОДИ МОБІЛЬНИМИ ДОДАТКАМИ

1.1 Перші математичні підході до прогнозування погоди

Згідно з даними з Вікіпедії [1], “Прогнозування погоди” – науково обґрунтоване припущення про майбутній стан погоди в певному пункті або регіоні на певний період. Складається (розробляється) метеорологічними службами на основі методів метеорології. Логічно припустити що прогноз дається завчасно (як заведено– за тиждень) до потрібного терміну. Прогнози розподіляють на багато категорій, проте серед них найцікавіші з періоду прогнозування:

1. Наукастінг (прогноз поточної погоди) – від 0 до 12 годин;
2. Часові (ЧПП) – до 12 годин;
3. Короткострокові (КПП) – від 12 до 72 годин;
4. Середньострокові (СПП) – від 72 годин до 10 діб;
5. З розширеним терміном – від 10 до 30 діб;

Як можна помітити, з будь-якого додатку прогноз найчастіше дають не більше ніж на 10 діб − він є найпростішим в порівнянні з іншими, з більш об'ємними видами та періодами прогнозування через менше кількості необхідного часу і сил для обробки даних. Також прогноз погоди можна диференціювати і за метою їх призначення [2]:

1. Прогнози загального користування (публікуються в ЗМІ і на інтернет-сайтах, опублікованих на ТБ і Радіо) – містять коротку інформацію про хмарність, атмосферних опадах, атмосферні явища, вітри, температури, вологості повітря і атмосферний тиск;
2. Авіаційні прогнози також містять детальну характеристику вітру, видимості, атмосферних явищ, хмарності, температури повітря;

3. Морські та річкові прогнози містять детальну характеристику вітру, хвилювання, атмосферних явищ, температури повітря;

4. Сільськогосподарські (агрометеорологічні) прогнози містять детальну характеристику опадів і температури повітря;

Виходячи з цієї інформації, можна зробити висновок − прогнозування погоди є складним математичним процесом з використанням законів короткострокового і довгострокового прогнозування з досить великою кількістю динамічних даних у вигляді різноманітних показників атмосферного тиску, температури. Щоб виробляти найбільш точний прогноз варто враховувати не один десяток даних і використовувати досить складну систему з взаємопов'язаними відносинами різноманітних погодних показників.

1.2 Найбільш популярні мобільні додатки для прогнозування погоди та способи отримати прогноз погоди завдяки суперкомп’ютера

1) AccuWeather [2] присутня можливість відображати щотижневі, щоденні і щогодинні метеозведення. Для будьякого користувача цієї програми можна дізнатися про температуру повітря в градусах Цельсія або Фаренгейта, вологості, опадах, швидкості і напряму вітру, а також часу сходу і заходу сонця. Додаток повідомить про інтенсивність сонячного випромінювання, якщо користувач зібрався прийняти сонячні ванни, або про що насувається бурі. Крім того, AccuWeather має вбудовану карту, що дозволяє в реальному часі стежити за погодою в вашому регіоні, країні або в цілому континенті. Є підтримка годин з Wear OS і watchOS, а також важливі та корисні віджети. AccuWeather безкоштовний, але за відключення реклами користувачеві доведеться заплатити.

Також крім усього вішеперечесленного в додатку є і AccUcast − функція для руху хмарних масивів і зміна кліматичних умов в режимі реального часу. На даний момент у додатковій формі буде багато внутрішньої реклами, яка, у

свою чергу, відключається для платної підписки за 104,99 грн. Платна підписка дозволяє позбутися від постійної реклами, можливість отримати прогноз на наступні 48 годин та як гарантує додаток − точний прогноз на наступні 10 днів.Також в додаток додана функція для читання новин по всьому світу впливають погодні умови тієї чи іншої країни.

2) 1Weather [2] − досить популярний погодний додаток, яке надає всі необхідні інформацію про погоду в зручному і красивому виді. Він може показувати тижневий прогноз в реальному часі, анімації сходу і заходу сонця, а також фази місяця. Користувач може вибрати до 12 міст, в яких хочете відстежувати температуру повітря, швидкість і напрям вітру, вологість і опади − це стане в нагоді мандрівникам. 1Weather включає в себе погодну карту. Додаток підтримує годинник з Wear OS [2].

Доповнення до можливостей додатку −це факти про погоду в різних країнах і навчальні відео про погодні явища на англійській мові. Додаток доступний також і російською мовою.

3) The Weather Channel − один з найбільш точних у виведенні метеоданих погодних додатків. Він неймовірно популярий за кордоном, але прекрасно підходить саме користувачам з Росії або Україні. Надає прогнози погоди на кожну годину, на 15 днів і на вихідні. На панелі сповіщень можна розмістити інформацію про погоду в будь-яких точках світу. Додаток відмінно оптимізований і для смартфонів, і для планшетів. В The Weather Channel є зручні віджети з різними співвідношеннями сторін для будь-якого робочого столу [2].

Крім особливої точності прогнозу також ще мається погодинний та подобовий прогноз з точними даними по забрудненню повітря в даному регіоні. Додаток не тільки показує кількість і щільність шкідливих часток в повітрі, а й їх аналіз.

4) Weather Underground [2] − додаток однойменної метеорологічної служби. Має все такі ж можливості, що і інші погодні додатки зі списку, але має розширені можливості. Наприклад, він буде доповідати вам про якість повітря і рівні ультрафіолетового випромінювання в вашому місті, а також попереджати про спалахи грипу або загрози алергії. Weather Underground покладається на дані більш ніж 270 000 метеостанцій по всьому світу. Крім того, відомості про погоду роблять і самі користувачі програми: повідомлення відображаються на інтерактивній карті. Нарешті, за допомогою веб-камер додаток дозволяє подивитися що відбувається в тому чи іншому місті. Є підтримка Apple Watch.

Всі функції Weather Underground доступні в безкоштовній версії. Щоб позбутися від реклами, доведеться купити річну передплату.Даний продукт компанії IBM хоча і має цікаві функції − на кшталт моніторингу щоденного погодного графіка та навіть оцінку повітря, але неймовірно засмутило при поглибленні в його функції. Точність прогнозування додатку має більше ніж 65% точності − це свідчить про помилку всередині алгоритму. З приводу повітря навіть незрозуміло на рахунок яких даних грунтується такий аналіз.

Додаток не оновлює самостійно виведений прогноз, змушуючи користувачів самостійно згадувати про це, при цьому в додатку є свій віджет на робочому столі. З ранку також виводить нічну погоду.

5) Yahoo Погода − це, безумовно, один з найпопулярніших додатків зі списку. «Yahoo Погода» [2] постійно завантажує чудові фотографії з Flickr і автоматично вибирає фони, відповідно до розташування, часу доби та погодних умов. Виглядає дуже ефектним, але й функціональність не підводить. Додаток може показувати погоду одночасно в 20 містах. На найближчу добу і 10 днів відображаються докладні прогнози: дані про температуру та вологість повітря, опади, швидкість та напрям вітру, атмосферний тиск та навіть інтенсивність УФ випромінювання. Доповнюють звичну інформацію інтерактивна погодна карта, попередження про погану

погоду, красиві анімації сходу і запада. Віджети «Yahoo Погоди» прості і зручні. Є підтримка Apple Watch. Додаток безкоштовий, але в ньому є реклама.

На додаток також було є контроль за використання мобільних даних у цьому додатку, контроль за появою повідомлень від нього та як у одного з попередніх програм, мається інформація про поточну пандемії в цьому регіоні. Також показується максимальна і мінімальна температури за день, графік опадів та ймовірність їх випадання. Окреме слово варто сказати саме про точність даних видається даними додатком. В окремому випадку слід вказати точність прогнозування температури − при її уточнення і порівнянням даних температури з нашого термометра, який стояв в тіні, різниця була не так вже й велика − не більше 2 градусів за Цельсієм. За нинішніми результатами дослідження цей додаток зараз лідирує, як втім і по випаданню опадів − з ймовірністю в 60 + відсотків випадання опадів у вигляді дощу прогноз був абсолютно точним – навіть у часі їх випадання.

6) YoWindow − погодний додаток з простим інтерфейсом з цікавою особливістю − цифри і значки розташовані у верхній частині екрану, а все інше простір займає пейзаж який залежить від того, що відбувається за вікном. Дощ та сонце, схід та захід сонця, пори року які змінюють один одного − все як в реальності. До речі, пейзаж можна вибрати і встановити в якості робочого фону у телефоні. Ще в YoWindow є віджети і будильник зі звуками природи [2]. У даного мобільного застосування вельми специфічний дизайн − гарний анімаційний робоче меню програми повний купою різних показників − одночасно і погодинний прогноз, і окремі дані про видимість, про заходи і сходи сонця, дані про метеосупутник та інше.

Точність вкрай сумнівна − незалежно від використання великої кількості метеосупутників, прогнозований дощ з вірогідністю в 75 відсотків так і не відбувся. Температура і зовсім була спрогнозована з помилкою більше ніж в 5 градусів.

7) CARROT Weather − у додатку виділяється ШІ (штучний інтелект) з великою кількістю чорного гумору. Що б не творилося за вікном, він буде

говорити свої коментарі. Цинічні зауваження і жарти відпускаються інколи і на адресу користувача. Що правда − тільки англійською мовою[2]. До того ж в CARROT Weather повно сюрпризів. Наприклад, функція Time Travel дозволяє дізнатися, яка погода була в минулому. Це може знадобитись звичайному користувачу при обчисленні сприятливого періоду під час перевезення вантажів або в сільському господарстві. Сам додаток прогнозує досить точно − навіть приблизна різниця в градусах була не більше 1.

8) Weather Radar & Forecast − ефективний додаток що вміє підлаштовувати фон екрану блокування до подій за вікном. Крім того, тут маються різні віджети з відомостями про погоду в реальному часі, зображеннями вашого міста, годинником і календарем. Weather Radar Forecast відображає дані про температуру та кількість опадів в рядку стану Android, так що вони будуть перед очима, навіть коли відкриті інші додатки. Ще цей додаток вміє відправляти оповіщення про раптові ураганах, бурях і лавини [2]. Якщо протягом 10 хвилин почнеться дощ, він обов'язково виведе повідомлення про це.

Точність погоди в цілому наближена до ідеалу − як опади, так і зміни в температурі. Все це через використання повсякчасного поновлення динамічних погодних показників всередині програми.

9) Minimalist Weather − більшість погодних додатків для Android і iOS завалюють купою даних. У цьому додатку немає нічого зайвого. Тільки прогноз на найближчі п'ять днів з температурою, тиском та даними про опади. Втім, навіть цей мінімум можна скорегувати [2]. Дизайн Minimalist Weather простий для користування та виглядає досить елегантно. Анімації плавні, управління зручне. Якщо говорити про точність прогнозування, то додаток фактично не помиляється − у нього вкрай мала кількість даних і тому вони оновлюються при кожному підключенні до мережі.

1.3 Прогноз погоди своїми руками

WRF − це чисельна модель передбачення погоди, яка підходить як для прогнозування стану атмосфери, так і для наукових досліджень [3]. Розробляється співтовариством наукових організацій США, в тому числі Національним центром атмосфери і океану, Національним центром атмосферних досліджень. Являє собою систему модулів: модуль підготовки початкових і граничних даних (WRF Preprocessing System), власне вирішальне ядро (Advanced Research WRF), модуль постпроцесингу (WRF Postprocessing System). Велика частина моделі (власне вся математика) реалізована на мові fortran з використанням бібліотеки MPI.

На C написані модулі для роботи з даними. Модель доступна в початкових кодах. Природно модель споживає величезну кількість процесорного часу, і спроектована для запуску на суперкомп'ютерах, оскільки прогноз погоди − одна з найскладніших задач.

Скачується набір вхідних даних (поточний стан атмосфери). Потім запускаетмя модуль препроцессінга, які складається з трьох програм:

1) geogrid.exe − обробляє статичні дані (рельєф, типи грунту, гідрографія);
2) ungrib.exe − розпаковує початкові дані;
3) metgrid.exe – потрібен для інтерполяції отриманих вище даних.

Потім стартує ядро моделі, які за допомогою чисельних методів вирішує нелінійну систему диференціальних рівнянь в приватних похідних. Воно складається з двох розпаралеленого програм:

1) real.exe − виконує вертикальну інтерполяцію вхідних даних;
2) wrf.exe − власне дозволяє ядро.

Потім після завершення розрахунку отримуємо набір даних, тобто прогноз стану атмосфери на кілька діб вперед з дискретизацією в 1 годину. Тепер необхідно обробити ці дані та побудувати слайди, на які б ми нанесли приземну температуру, вітер, тиск та опади. Насправді wrf продрукує велику

кількість полів, але ми обмежимося тільки цими. Для цих цілей скористаємося програма wrfpost.exe з модуля постпроцесингу.

Якщо ж немає суперкомп'ютера, то процес обчислення займе досить великий час. У домашніх умовах можна використовувати Core i5 або i7, чим більше ядер − тим швидше буде вважати. На нашій машині знадобилося чотири ядра I3 − це зайняло майже два дня щоб це зробити. Крім копіляторов необхідно встановити бібліотеки MPI, наприклад, MPICH2. Єдина вимога − платформа повинна бути x86_64. Інакше препроцессінг не заведеться. Витрати у часі для області 3000 x 3000 км (прогноз на 2 доби) наведені в таблиці нижче.

Таблиця 1.1 – Необхідні дані для уточнення витрат у часі

Параметри / **Цикли обчислень**	**Кроки**	**Кількість точок**	**Час обчислень**
3	10	301 x 301	16
2	20	151 x 151	12
1	30	101 x 101	10

Всі настройки задаються в файлах namelist.input і namelist.wps.

Кроки по осях X і Y (параметри DX, DY) краще ставити однаковими. І крок повинен залежить від тих потужностей, якими ми володіємо. Погратися вистачить 10-50 км. Крок за часом DT рекомендується виставляти рівним 9 * DX (в км). DX, DY задаються в метрах, а DT в секундах.

У прикладах нижче використовується версія 3.2.1. Як мінімум, будуть потрібні:

1) WRF model V3.2.1,
2) WRF Preprocessing System V3.2.1,
3) WRF Postprocessing System V3.2,
4) low-resolution geographical data (10' resolution),
5) Початкові дані (масштабування 1°, розмір порядку 500МБ):

Щоб завантажити модель потрібно пройти процедуру простої реєстрації. Для того, щоб зробити модель з кроком менше 10 км, то потрібно буде завантажити більш детальний рельєф. Програму grads (для побудови слайдів) можна взяти з проекту OpenGrads.

РОЗДІЛ 2 АЛГОРИТМИ КОРОТКОСТРОКОВОГО ПРОГНОЗУ ПОГОДИ

Мета: Необхідно розглянути загальновідомі алгоритми короткострокового прогнозу погоди, перевірити їх актуальність і точність в обчисленні погодних показників, зрозуміти принципи їх реалізації в мобільних додатках.

2.1 Розробка пристрою для збору і обробки метеорологічних даних з будь-якого інтернет-джерела

В даний час розроблено велику кількість пристроїв, здійснюють прогнозування погоди, які включають в себе такі функції як: вимірювання температури в приміщенні тана вулиці, вологості, швидкості й напряму вітру. В якості додаткових можливостей − збереження температурних значень, годинник, календар звичайний і місячний, визначення температури точки роси, індексу нагрівання, вимір температури охолодження вітром. Сучасні портативні метеостанції не дозволяють отримувати інформацію про погоду віддалено. Показ поточної погодної обстановки здійснюється за допомогою дисплея, що не завжди зручно.

В даний час велика частина мобільних пристроїв підтримує вихід в Інтернет, тому впровадження в поширені метеостанції веб-інтерфейса або мобільного додатка збільшило б кількість покупців і зробило б роботу з пристроєм більш зручною і надійною. Дані показники досягаються тим, що на веб-сервері реалізовано зберігання інформації за певний період, що дозволяє будувати погодну статистику. Також в даній системі передбачені функції аутентифікації і авторизації користувачів. Виходячи з описаного вище, стало необхідно розробити побутову (портативну) метеостанцію за формуванням короткострокового прогнозу, яка має у собі: визначення температури,

вологості повітря, атмосферного тиску, освітленості, швидкості і напряму вітру, температура, ймовірность нічних заморозків і простий вебінтерфейс [4].

В якості апаратної основи були обрані одноплатний комп'ютер Raspberry Pi 2 і мікроконтролер Arduino Nano.Так як ARM процессор, встановлений на Raspberry Pi, не має у своєму складі АЦП, то виникла необхідність застосування стороннього перетворювача сигналів, що надходять з аналогових датчиків. В якості такого перетворювача в даному проекті використовується Arduino Nano. В якості сенсорів в даній системі застосовуються бюджетні моделі: DHT12, BMP180, GY-30 BH1750FVI, Vortex Anemometr, RTC DS1307. На рис 2.1 приведена структурна схема розробленого пристрою. Як барометра використовується датчик BMP180, як гігрометра − DHT12, люксметра − GY-30 BH1750FVI, термометра − середнє арифметичне показань DHT12 і BMP180, як анемометра − Vortex Anemometr, годин реального часу − RTC DS1307.

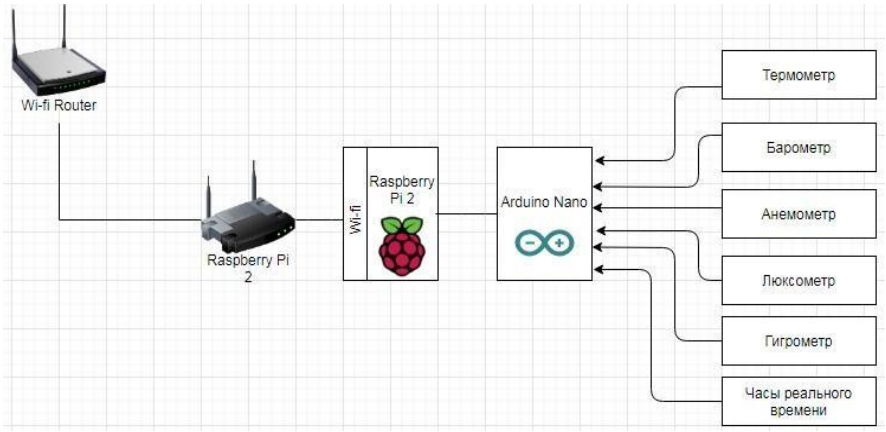


Рисунок 2.1 − Структурна схема розробленого пристрою

Дані від сенсорів надходять на мікроконтролер Arduino Nano, після чого відправляються на Raspberry Pi 2. Одноплатний комп'ютер, використовуючи Wi-Fi модуль, відправляє дані на роутер. Користувач за допомогою інтернет-браузера може ознайомитися з поточними погодними умовами. Варто відзначити, що Raspberry Pi 2 є більш повною платформою з точки зору

можливостей. Ідея спільного використання полягає в тому, щоб взяти найкраще з кожної платформи: використовувати Arduino Nano, щоб здійснити з'їм метеоінформації і передати на Raspberry Pi 2 для подальшої обробки і відображення за допомогою Wi-Fi технології. Обмін даними всередині пристрою здійснюється за допомогою інтерфейсу взаємодії I2C. При передачі даних від датчиків на Arduino Nano, мікроконтролер працює як master, датчики − як slave. Далі Arduino Nano перемикається з режиму master на slave і здійснює пересилку інформації на Raspberry Pi 2, який працює в режимі master. На рисунку 2.2 наведено фотографічне зображення розташування апаратних елементів пристрою.

Рисунок 2.2 − Фотографічне зображення розташування апаратних елементів пристрою

Для організації взаємодії між користувачем і метеостанцією передбачений веб-інтерфейс. За рахунок його реалізації вдалося досягти великої кількості платформ, які можливо підключити до розробленої системи. Так як пристрій функціонує в режимі «сервера» (воно працює незалежно від того, підключений до нього клієнт чи ні), то передача даних здійснюється за запитом від підключених «клієнтів». Адаптивна верстка дозволяє відображати

одні й ті ж дані на мобільному пристрої або на персональному комп'ютері, не погіршуючи при цьому читаність. Для того щоб доступ до веб-інтерфейсу був відкритий, необхідна наявність виділеного IP-адреси. Сама метеостанція при цьому повинна бути підключена безпосередньо в мережу або через маршрутизатор. Дані з Raspberry Pi 2 відправляються по каналу Wi-Fi на роутер, де вони комутуються і слідують на пристрій, запросила погоду. У тому випадку, якщо запит надійшов з локальної мережі, то дані передаються відправнику. При взаємодії з системою за вказаною схемою є можливість обміну з використанням глобальної мережі. У цьому випадку прогноз може бути завантажений на будь-який пристрій, що має доступ в інтернет. На рисунку 2.3 приведена схема можливої організації взаємодії системи.

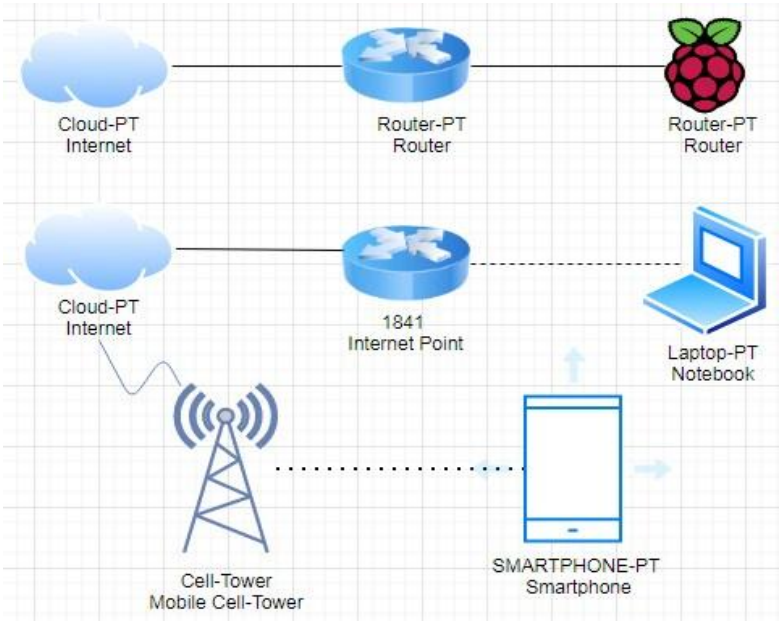


Рисунок 2.3 − Схема організації взаємодії системи

Слід зазначити, що система здатна організовувати роботу і використовуючи додаткове програмне забезпечення. Для цього передбачена система команд для взаємодії через POST-запити. При початковому налаштуванні станції необхідно встановити пароль для доступу. Для запобігання втрати доступу на системі є можливість скидання через службовий аккаунт. Він унікальний для кожної станції і надрукований в

документації до метеостанції. Програмна частина системи написана на мові C # в середовищі Microsoft Visual Studio. У розробленій системі для формування короткострокового прогнозу використовуються:

1) Метод професора П.І. Броуновим для визначення ймовірності нічних заморозків.

2) Метод визначення температури і вологості повітря на найближчу добу.

3) Програмна реалізація погодного калькулятора Zambretti.

Для формування прогнозу ймовірності виникнення нічних заморозків використовується інформація про температуру о дев'ятій годині вечора і значення різниці температур між годиною дня і дев'ятою годинами вечора. На рисунку 2.4 наведена залежність, виведена професором П.І. Броуновим. По осі ординат відкладені значення температури о дев'ятій годині вечора, по осі абсцис - значення різниці температур. Праворуч від графіка наведені ймовірності виникнення заморозків.

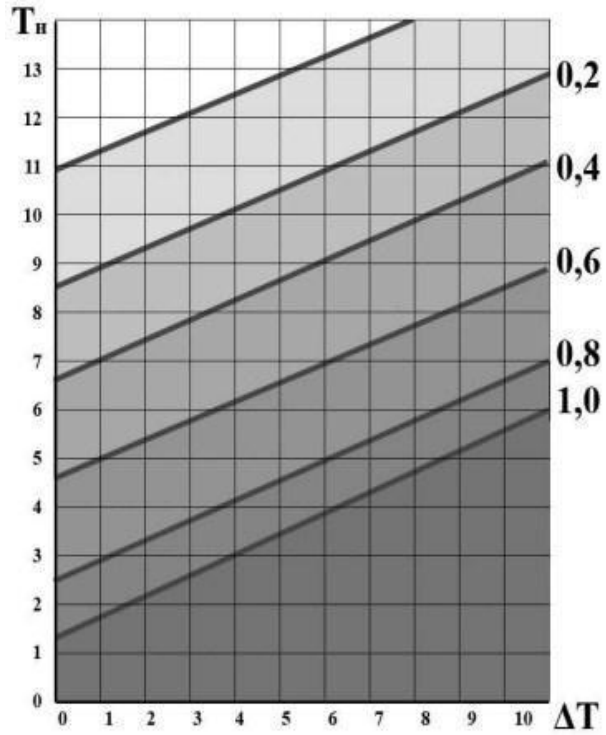


Рисунок 2.4 − Графік визначення ймовірності нічних заморозків

Метод, запропонований професором П.І. Броуновим, є найкращим, тому що він побудований по одному фактору - температурі. Температура і вологість повітря на найближчу добу визначаються зміною в повітряній масі, що переміщуються в пункт прогнозу з будь-якого регіону, і в той же час, змінюються під впливом термічних і вологих властивостей підстильної поверхні. Прогноз температури повітря до поверхні Землі на найближчу добу розраховується за формулою:

$$Tnp = Ta + \delta mpT + \delta c.\, x.T, \tag{2.1}$$

де Tnp − прогностичне значення температури повітря, Ta − значення температури повітря, звідки очікується переміщення повітряної частки, δmpT − трансформаційні зміни температури повітря, $\delta c.\, x.\, T$ − зміна температури повітря за рахунок добового ходу.

Прогноз вологості повітря на найближчу добу розраховується за формулою:

$$Td = Tda + mpTd + \delta c.\, x.\, Td, \tag{2.2}$$

де Td − прогностичне значення вологості повітря, Tda − значення вологості повітря, звідки очікується переміщення повітряної частки, δтрTd − трансформаційні зміни вологості повітря, δс.х.Т − зміна вологості повітря за рахунок добового ходу.

Зміна погоди може бути визначено за допомогою погодного калькулятора Zambretti, який складається з трьох дисків. Зовнішній великий диск враховує напрямок вітру. Середній диск − атмосферний тиск. Внутрішній диск − зміна тиску і сезон. Сезон в даному випадку поділяється на річний (квітень-вересень) і зимовий (жовтень-березень). Шляхом поєднання дисків

складається прогноз погоди на внутрішньому диску. Прогноз виводиться у вигляді букви латинського алфавіту A-Z, де A − це «Встановиться хороша погода», Z − це «Вітер з дощем».

Програмна реалізація розроблена на основі наступного алгоритму:

1. Визначають тенденцію зміни атмосферного тиску (росте, падає, незмінно протягом декількох годин).
2. У момент розрахунку визначають значення атмосферного тиску.
3. Розраховують показник Z за формулою внизу.
4. Проводять кореляцію отриманого значення Z за напрямком вітру, барической тенденції і по сезону.
5. За отриманого значення Z визначають прогноз погоди, розшифровуючи його.

Значення Z для різних тенденцій атмосферного тиску:

1) для позитивної тенденції − $Z = 130 - P/8{,}1$,
2) для негативної тенденції − $Z = 147 - 5P/37{,}6$,
3) для стабільного тиску − $Z = 179 - 2P\ /\ 12{,}9$,

де P − атмосферний тиск в момент розрахунку.

Рисунок 2.5 − Карта із зображенням використаних в науковій роботі метеорологічних станцій

2.2 Актуальність можливостей OpenWeather при прогнозуванні погоди

Ідея вільної та доступної інформації про погоду привела до того, що команда фахівців і розробників сервісу OpenWeatherMap надала можливість розробникам додатків використовувати безкоштовний API для відображення погодних даних, такими як:

1) Використання інтерактивної карти з точними і корисними погодними даними про поточну погоду;

2) Загальнодоступність прогнозу на весь тиждень в потрібному для користувача місті;

3) Можливість використання будь-яких історичних даних в 120 000 міст усього світу;

4) Використання даних з 40 000 метеостанцій, розкиданих по всьому світу в режимі онлайн з невеликою затримкою у часі;

5) Отримання необхідних даних з web-карт, наприклад з карти опадів, вітру та інших.

Робота даного сервісу заснована на наступною схемою:

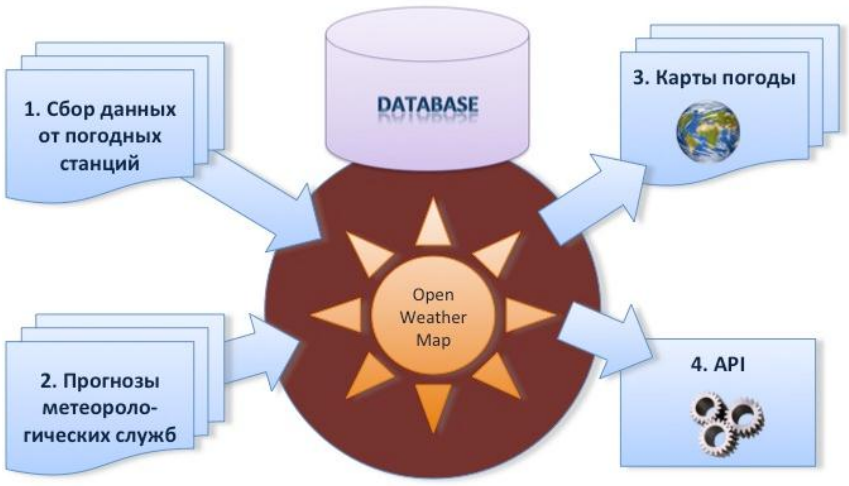


Рисунок 2.6 − Схема роботи вхідних та вихідних компонентів сервісу OpenWeatherMap

Спочатку відбувається отримання даних з погодних станцій разом з прогнозами різних метеослужб та інших джерел метеоданих, наприклад − наукових лабораторій, що роблять власний аналіз погодних показників.

Потім всі дані зберігаються в базі даних даного сервісу і після їх обробки вони стають уже інтерполяції даними про поточну погоду в будь-якій точці світу, а також вносяться в карти з погодними явищами. Після цього процесу, для всіх оброблених даних про погоду, включаючи також і можливі карти з різними погодними явищам, присвоюється унікальний API.

Вхідні дані системи OpenWeather:

1) Дані з метеостанцій − з 40 000 приватних станцій, а також з усіх непрофесійних;

2) Прогнози − оброблені дані метеослужб NOAA та Environment Canada;

3) Погодні дані для web-карт − використання моделі GFS для відображення глобальних, середніх та локальних метеоданих.

Вихідні дані в системі OpenWeather:

1) Об'ємні і точні інтерактивні карти про погоду і погодні явища у вигляді карт хмар, тиску, опадів, вітру та навіть дані радарів;

2) безкоштовний API у вигляді Tile / WMS для картографії або у вигляді JSON для отримання даних про нинішню погоду, прогнози та всіх можливих погодних карт.

Так виглядає загальна схема роботи OpenWeatherMap:

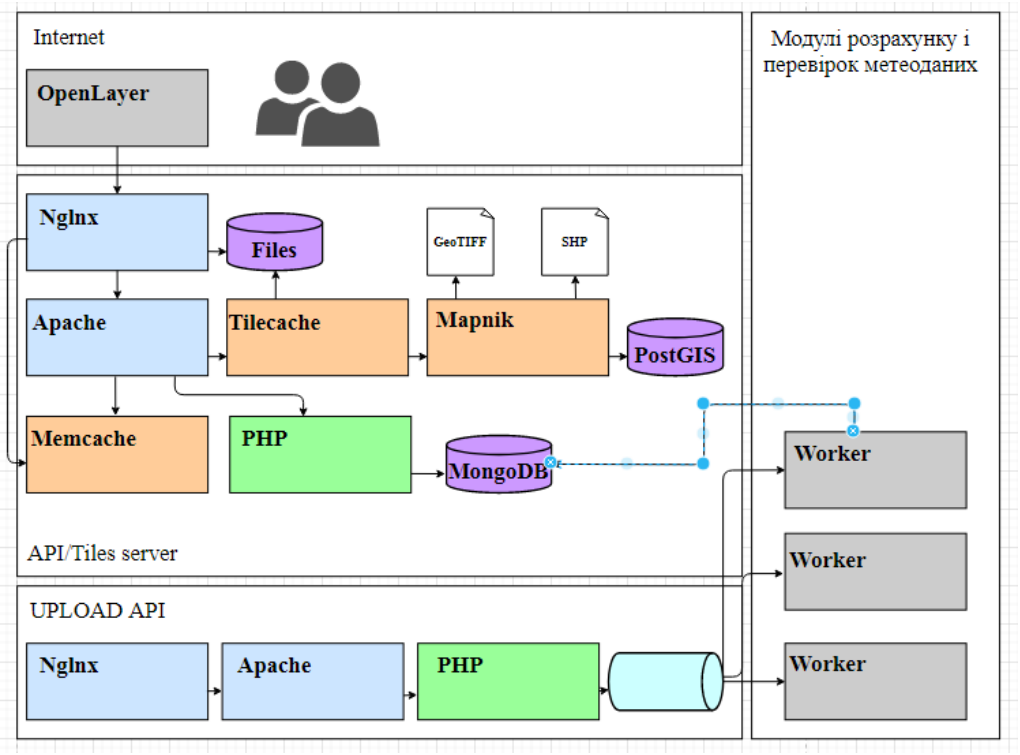


Рис 2.7 − Загальна схема роботи сервісу OpenWeatherMap

2.3 Можливість використання різних API ключів: в чому різниця і вигода для використання в додатках

Використання OpenWeather сервісу обумовлено, перш за все, простотою у використанні, безкоштовністю і стабільністю в отриманні даних [6].

Вивчимо основні розділи:

1) Відкриваємо сайт OpenWeather
2) Перейдемо до розділу API, обравши його в навігаційної панелі.
3) У секції Current weather data вибрати API Doc
4) Розбираємося в інформації, наданої в Current weather data API. Виклики API дозволяють розробникам запитувати інформацію для своїх додатків. Іншими словами, API нададуть дані для додатків, які розробляють розробники.

Перш ніж будемо заглиблюватись в API OpenWeatherMap, розглянемо API іншого сервісу погоди. На відміну від OpenWeatherMap API, Aeris Weather API більш надійний і великий.

Досліджуємо Aeris Weather API, виконавши такі дії:

1) Відкриваємо сайт Aeris
2) Переходимо в розділ документації, обравши розділ Documentation
3) Weather API, далі обрати Data Endpoints
4) Переходимо до Reference в бічному меню та обираємо Endpoints
5) в списку кінцевих точок виберемо observations

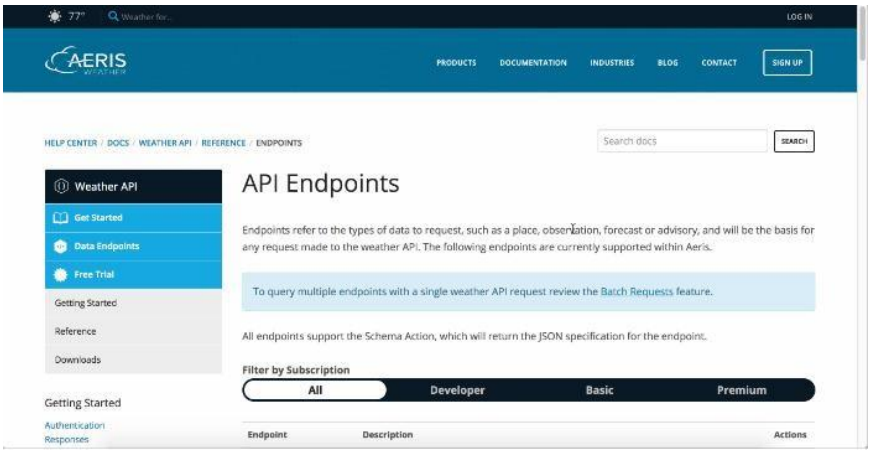


Рисунок 2.8 − Вкладка API Endpoints в API Aeris

Для нашого сценарію (зображення прогнозу погоди в мобільному додатку), можна використовувати декілько API-сервісів погоди.

JSON (JavaScript Object Notation) − це найбільш поширений спосіб повернення інформації API REST. Незважаючи на те, що деякі API повертають інформацію як в JSON, так і в XML, краще використовувати JSON для аналізу відповіді і відображення його на веб-сторінці. Це обумовлено тим, що JSON набагато краще вписується в існуючу технологію JavaScript + HTML + CSS, яка підтримує більшість веб-сторінок. За допомогою JavaScript або Java можливо легко аналізувати JSON і інтегрувати його в власний веб-контент.

Розгорнуту відповідь від кінцевої точки прогнозу погоди OpenWeatherMap буде виглядати приблизно так:


```
{
    "coord": {
        "lon": -121.96,
        "lat": 37.35
    },
    "weather": [
        {
            "id": 801,
            "main": "Clouds",
            "description": "few clouds",
            "icon": "02d"
        }
    ],
    "base": "stations",
    "main": {
        "temp": 70.14,
        "pressure": 1012,
        "humidity": 33,
        "temp_min": 62.6,
        "temp_max": 75.2
    },
    "visibility": 16093,
    "wind": {
        "speed": 14.99,
        "deg": 330
    },
    "clouds": {
        "all": 20
    },
    "dt": 1522619760,
    "sys": {
        "type": 1,
        "id": 479,
        "message": 0.0058,
        "country": "US",
        "sunrise": 1522590707,
        "sunset": 1522636288
    },
    "id": 420006397,
    "name": "Santa Clara",
    "cod": 200
}
```


Рисунок 2.9 − JSON відповідь від погодного сервісу OpenWeather

JSON був би не дуже корисний, якщо б завжди доводилося роздруковувати усю відповідь. Замість цього можна вибрати потрібну властивість, і отримати його через точкову запис. Крапка після response (ім'я корисної даної JSON, як вона визначена довільно в функції JQuery AJAX) визначає доступ до потрібних значень з об'єкта JSON.

Припустимо, ми хочемо отримати частину про швидкість вітру у відповіді JSON. Так виглядає точкова нотація, яку потрібно використовувати:

```
response.wind.speed
```

Для виведення елемента швидкості вітри з випуску JSON і розподіліть його в консолі JavaScript, додайте його в приклад коду (який ми створили в попередньому розділі) прямо під рядком console.log (відповідь):

```
console.log("wind speed: " + response.wind.speed);
```

Код буде виглядати так:

```
$.ajax(settings).done(function (response) {
console.log(response);
console.log("wind speed: " + response.wind.speed);
});
```

Після цього варто оновити браузер і вже можна дізнатися інформацію, що з'явилася в консолі:

```
wind speed: 13.87
```

РОЗДІЛ 3 ПРАКТИЧНА ЧАСТИНА НАПИСАННЯ ПРОГРАМИ І ПІДВЕДЕННЯ ПІДСУМКІВ РЕЗУЛЬТАТІВ РОБОТИ ПОДІБНОГО АЛГОРИТМУ

3.1 Використовувані програми і програмні компоненти при створенні описаного мобільного додатка

Eclipse ADT Bundle:

Дане середовище розробки модульних кросплатформених додатків потрібно було завантажити з основного сайту.

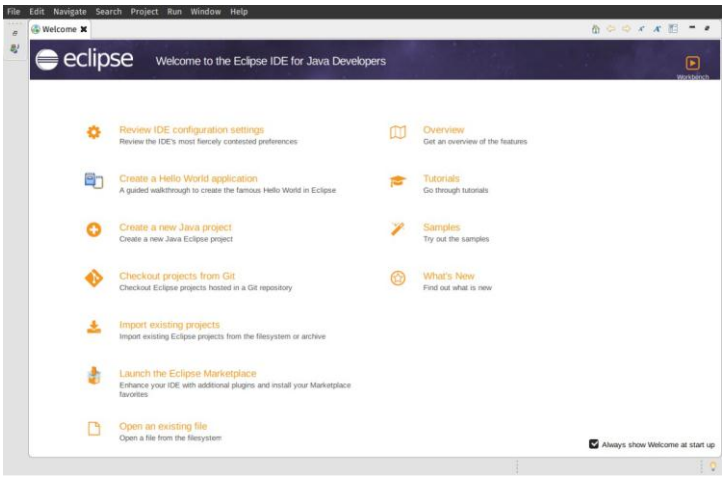


Рисунок 3.1 − Вигляд робочого середовища Eclipse

Ключ API OpenWeatherMap:

Необов'язково було б використовувати саме цей APIсервіс, але він є безкоштовним. Його можна отримати на сайті OpenWeatherMap.

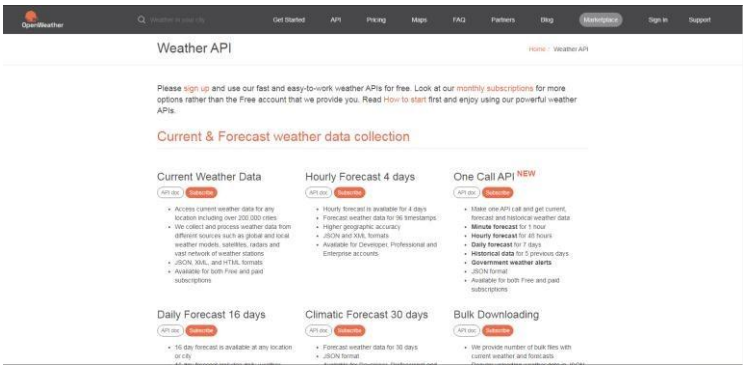


Рисунок 3.2 − Сайт сервісу OpenWeather для отримання особистого API-ключа

Іконки погодних умов:

При створенні був використаний шрифт Weather Icons Font, створений Еріком Флорсом [. Потрібно завантажити файл TTF, тому що саме його будемо використовувати в нашому додатку. Використовували цей шрифт щоб показувати різні значки в залежності від погодних умов.

3.2 Створення мобільного додатку для прогнозування погоди

Назва у цієї програми буде Simple Weather. Вводимо унікальне ім'я пакета, встановили мінімальний необхідний SDK на Android і встановили цільової SDK на останню актуальну версію. Можна залишити тему Holo Dark.

Ця програма буде мати тільки одне Activity, і воно буде засноване на шаблоні Blank Activity, як показано нижче.

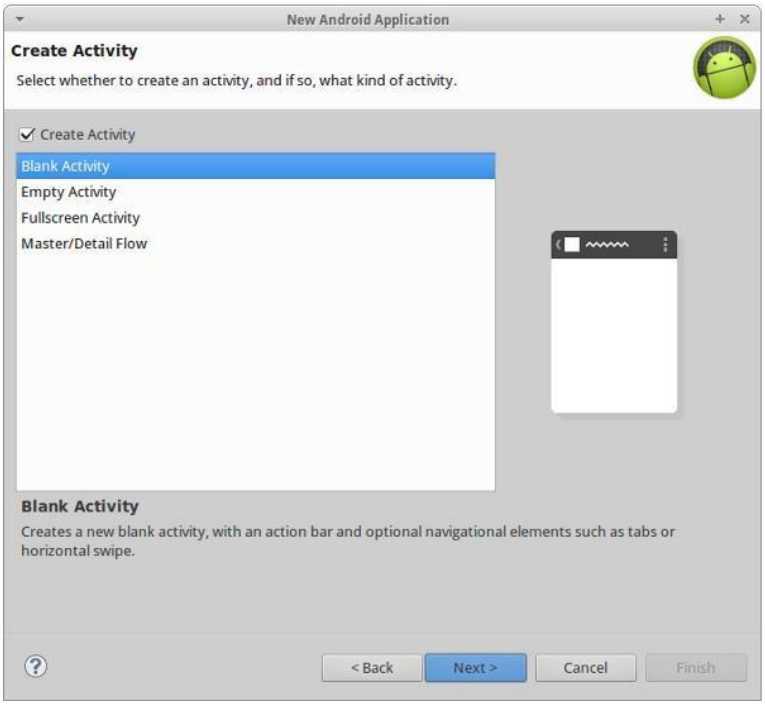


Рисунок 3.3 − Створення Blank Activity

Названо Activity − WeatherActivity. Будемо також використовувати Fragment всередині цього Activity. Макет, пов'язаний з Activity називається activity_weather.xml. Макет, пов'язаний з Fragment називається fragment _weather.xml.

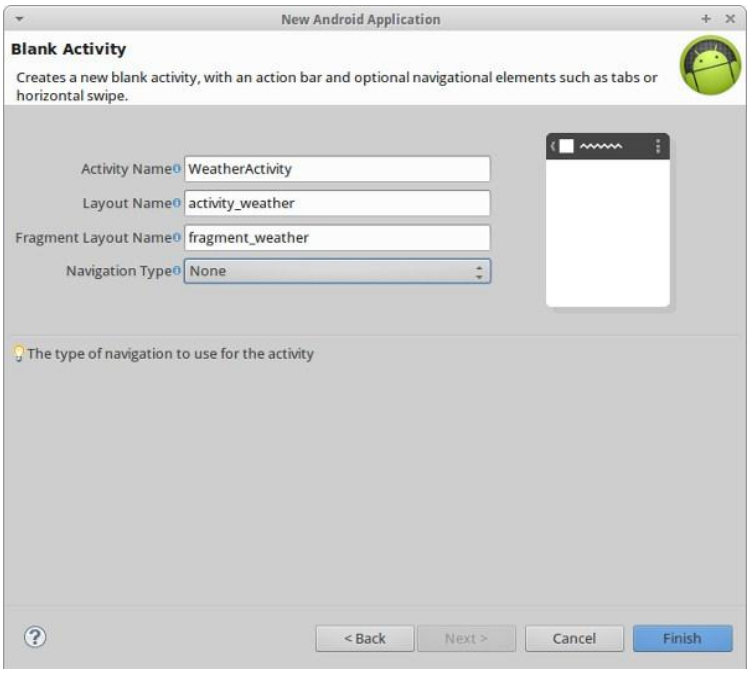


Рисунок 3.4 − Створення Layout та його Fragment

Копіюємо weathericons-regular-webfont.ttf в каталог assets / fonts нашого проекту та перейменовали в weather.ttf.

Єдине дозвіл, який потрібно в цьому додатку – це android.permission.INTERNET.

```
<uses-permission android:name="android.permission.INTERNET"/>
```

Рисунок 3.5 – Дозвіл додатку для доступу в інтернет

Вузол activity маніфесту повинен виглядати так:

```
<activity
    android:name="ah.hathi.simpleweather.WeatherActivity"
    android:label="@string/app_name"
    android:screenOrientation="portrait"
    >
    <intent-filter>
      <action android:name="android.intent.action.MAIN" />
      <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

Рисунок 3.6 – Вигляд вузла activity маніфесту

У activity_weather.xml змін було не так багато. Він повинен вже включати в себе FrameLayout. Використали додаткову властивість, щоб змінити колір фону на # FF0099CC.

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="ah.hathi.simpleweather.WeatherActivity"
    tools:ignore="MergeRootFrame"
    android:background="#FF0099CC" />
```

Рисунок 3.7 – Зміна кольору в activity_weather.xml

Змінили файл fragment_weather.xml, додавши п'ять тегів TextView, щоб відображати наступну інформацію:

1) місто і країна

2) поточна температура

3) значок іконки, що показує поточну погодну умову

4) відмітка часу, яка вказує користувачу, коли було отримано останнє оновлення інформація про погоду

5) більш детальна інформація про поточну погоду, наприклад, опис і вологість

Використовували RelativeLayout для розміщення текстових уявлень. У даній ділянці коду можна налаштувати розмір тексту textSize для різних пристроїв.

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="ah.hathi.simpleweather.WeatherActivity$PlaceholderFragment" >

    <TextView
        android:id="@+id/city_field"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:textAppearance="?android:attr/textAppearanceLarge" />

    <TextView
        android:id="@+id/updated_field"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/city_field"
        android:layout_centerHorizontal="true"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:textSize="13sp" />

    <TextView
        android:id="@+id/weather_icon"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:textSize="70sp"
        />

    <TextView
        android:id="@+id/current_temperature_field"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:textSize="40sp" />

    <TextView
        android:id="@+id/details_field"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/weather_icon"
        android:layout_centerHorizontal="true"
        android:textAppearance="?android:attr/textAppearanceMedium"
        />

</RelativeLayout>
```

Рисунок 3.8 − Зміни у файлі fragment_weather.xml

Даний файл містить рядки, що використовуються в нашому додатку, а також коди символів Unicode, які були використовувані для відображення значків погоди. Додаток може відображати дванадцять денних і дванадцять нічних різних типів погодних умов. Якщо ж потрібно більше, можна застосувати наступну хитрість. Добити наступні значення в файл values / strings.xml:

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">Simple Weather</string>
    <string name="change_city">Change city</string>

    <!-- Put your own APP ID here -->
    <string name="open_weather_maps_app_id">11111</string>

    <string name="weather_sunny">&#xf00d;</string>
    <string name="weather_clear_night">&#xf02e;</string>

    <string name="weather_foggy">&#xf014;</string>
    <string name="weather_cloudy">&#xf013;</string>
    <string name="weather_rainy">&#xf019;</string>
    <string name="weather_snowy">&#xf01b;</string>
    <string name="weather_thunder">&#xf01e;</string>
    <string name="weather_drizzle">&#xf01c;</string>

    <string name="place_not_found">Sorry, no weather data found.</string>

</resources>
```

Рисунок 3.9 − Внесення нових значень в strings.xml

Користувач повинен мати можливість вибирати місто, чию погоду побажають побачити. Змінюємо файл menu / weather.xml та додаємо елемент для цієї опції.

```xml
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    tools:context="ah.hathi.simpleweather.WeatherActivity" >

    <item
        android:id="@+id/change_city"
        android:orderInCategory="1"
        android:title="@string/change_city"
        app:showAsAction="never"/>

</menu>
```

Рисунок 3.10 − Можливість збереження міста в окремому меню додатку

Тепер, коли все XML-файли готові до використання, переходимо до запиту на API OpenWeatherMap для отримання даних про погоду.

Отримані поточні дані про погоду в будь-якому місті, в форматі JSON, за допомогою API OpenWeatherMap. У рядку запиту передаємо ім'я міста та систему вимірювання, в якій повинні бути за підсумком результати.

Наприклад, щоб отримати поточну інформацію про погоду в Канберрі, використовуючи метричну систему, відправляємо запит на http://api.openweathermap.org/data/2.5/weather?q=Canberra&un its=metric

Відповідь, яку отримуємо через API, виглядає так:


```
{
    "base": "cmc stations",
    "clouds": {
        "all": 90
    },
    "cod": 200,
    "coord": {
        "lat": -35.28,
        "lon": 149.13
    },
    "dt": 1404390600,
    "id": 2172517,
    "main": {
        "humidity": 100,
        "pressure": 1023,
        "temp": -1,
        "temp_max": -1,
        "temp_min": -1
    },
    "name": "Canberra",
    "sys": {
        "country": "AU",
        "message": 0.313,
        "sunrise": 1404335563,
        "sunset": 1404370965
    },
    "weather": [
        {
            "description": "overcast clouds",
            "icon": "04n",
            "id": 804,
            "main": "Clouds"
        }
    ],
    "wind": {
        "deg": 305.004,
        "speed": 1.07
    }
}
```


Рисунок 3.11 − Відповідь в форматі JSON з погодними показниками

Створюємо новий клас Java і називаємо його RemoteFetch.java. Цей клас відповідає за отримання даних про погоду через API OpenWeatherMap.

Використовували клас HttpURLConnection для виконання віддаленого запиту. API OpenWeatherMap очікує ключ API в HTTP-заголовку з ім'ям x-api-key. Це визначено в нашому запиті з використанням методу setRequestProperty.

Тепер використовуємо BufferedReader для читання відповіді API в StringBuffer. Коли отримали повну відповідь, перетворюємо його в об'єкт JSONObject. Як видно вище відповіді, дані JSON містять поле з ім'ям cod. Його значення дорівнює 200, якщо запит був успішним. Будемо використовувати це

значення, щоб перевірити, чи має відповідь JSON поточну інформацію про погоду чи ні.

Клас RemoteFetch.java виглядає наступним чином:

```
01  package ah.hathi.simpleweather;
02
03  import java.io.BufferedReader;
04  import java.io.InputStreamReader;
05  import java.net.HttpURLConnection;
06  import java.net.URL;
07
08  import org.json.JSONObject;
09
10  import android.content.Context;
11  import android.util.Log;
12
13  public class RemoteFetch {
14
15      private static final String OPEN_WEATHER_MAP_API =
16          "http://api.openweathermap.org/data/2.5/weather?q=%s&units=metric";
17
18      public static JSONObject getJSON(Context context, String city){
19          try {
20              URL url = new URL(String.format(OPEN_WEATHER_MAP_API, city));
21              HttpURLConnection connection =
22                      (HttpURLConnection)url.openConnection();
23
24              connection.addRequestProperty("x-api-key",
25                      context.getString(R.string.open_weather_maps_app_id));
26
27              BufferedReader reader = new BufferedReader(
28                      new InputStreamReader(connection.getInputStream()));
29
30              StringBuffer json = new StringBuffer(1024);
31              String tmp="";
32              while((tmp=reader.readLine())!=null)
33                  json.append(tmp).append("\n");
34              reader.close();
35
36              JSONObject data = new JSONObject(json.toString());
37
38              // This value will be 404 if the request was not
39              // successful
40              if(data.getInt("cod") != 200){
41                  return null;
42              }
43
44              return data;
45          }catch(Exception e){
46              return null;
47          }
48      }
49  }
```

Рисунок 3.12 − Зміни в класі RemoteFetch.java

Згідно із задумом користувач не повинен вказувати ім'я міста кожен раз, коли захоче використовувати додаток. Додаток має запам'ятати останнє місто, яке шукав користувач. Виконаємо це, використовуючи SharedPreferences. Однак замість прямого доступу до цих налаштувань з класу Activity, для цього було створити окремий клас.

Створили новий клас Java і назвали його CityPreference.java. Щоб зберегти і отримати ім'я міста, створили два методу setCity і getCity. Об'єкт SharedPreferences відтворюється в конструкторі. Клас CityPreference.java виглядає наступним чином:

```java
package ah.hathi.simpleweather;

import android.app.Activity;
import android.content.SharedPreferences;

public class CityPreference {

    SharedPreferences prefs;

    public CityPreference(Activity activity){
        prefs = activity.getPreferences(Activity.MODE_PRIVATE);
    }

    // If the user has not chosen a city yet, return
    // Sydney as the default city
    String getCity(){
        return prefs.getString("city", "Sydney, AU");
    }

    void setCity(String city){
        prefs.edit().putString("city", city).commit();
    }

}
```

Рисунок 3.13 − Зміни в класі CityPreference.java

Створили новий клас Java і назвали його WeatherFragment.java. У цьому фрагменті як макета використовується frag_weather.xml. Оголосили також п'ять об'єктів TextView і ініціалізували їх в методі onCreateView. Оголосили новий об'єкт Typeface з ім'ям weatherFont. Об'єкт TypeFace вказує на веб-шрифт, який був завантажений і збережений в папці assets / fonts.

Був використаний окремий Thread для асинхронного отримання даних через API OpenWeatherMap. Неможливо оновлювати призначений для користувача інтерфейс з подібного фонового потоку - тому нам потрібен об'єкт Handler, який ініціалізували в конструкторі класу WeatherFragment.

```java
public class WeatherFragment extends Fragment {
    Typeface weatherFont;

    TextView cityField;
    TextView updatedField;
    TextView detailsField;
    TextView currentTemperatureField;
    TextView weatherIcon;

    Handler handler;

    public WeatherFragment(){
        handler = new Handler();
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
            Bundle savedInstanceState) {
        View rootView = inflater.inflate(R.layout.fragment_weather, container, false);
        cityField = (TextView)rootView.findViewById(R.id.city_field);
        updatedField = (TextView)rootView.findViewById(R.id.updated_field);
        detailsField = (TextView)rootView.findViewById(R.id.details_field);
        currentTemperatureField = (TextView)rootView.findViewById(R.id.current_temperature_field);
        weatherIcon = (TextView)rootView.findViewById(R.id.weather_icon);

        weatherIcon.setTypeface(weatherFont);
        return rootView;
    }
}
```

Рисунок 3.14 – Створення класу WeatherFragment.java

Ініціалізували об'єкт weatherFont, викликаючи createFromAsset в класі Typeface. Після викликали метод updateWeatherData в onCreate.

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    weatherFont = Typeface.createFromAsset(getActivity().getAssets(), "fonts/weather.ttf");
    updateWeatherData(new CityPreference(getActivity()).getCity());
}
```

Рисунок 3.15 – Виклик createFromAsset в класі Typeface

У updateWeatherData запустили новий потік і викликали getJSON в класі RemoteFetch. Якщо значення, що повертається getJSON, так само null, виводиться повідомлення про помилку користувачеві. Якщо це не так, викликаємо метод renderWeather.

Тільки основний потік дозволяє оновлювати призначений для користувача інтерфейс програми для Android. Виклик Toast або renderWeather прямо з фонового потоку призведе до помилки виконання. Ось для чого були викликані ці методи з використанням методу post-обробника.

```
private void updateWeatherData(final String city){
    new Thread(){
        public void run(){
            final JSONObject json = RemoteFetch.getJSON(getActivity(), city);
            if(json == null){
                handler.post(new Runnable(){
                    public void run(){
                        Toast.makeText(getActivity(),
                                getActivity().getString(R.string.place_not_found),
                                Toast.LENGTH_LONG).show();
                    }
                });
            } else {
                handler.post(new Runnable(){
                    public void run(){
                        renderWeather(json);
                    }
                });
            }
        }
    }.start();
}
```

Рисунок 3.16 – Можливість оновлювати прогноз за допомогою updateWeatherData

Метод renderWeather використовує дані JSON для відновлення об'єктів TextView. Вузол weather відповіді JSON являє собою масив даних. Тут використовували перший елемент масиву метеорологічних даних.

```
private void renderWeather(JSONObject json){
    try {
        cityField.setText(json.getString("name").toUpperCase(Locale.US) +
                ", " +
                json.getJSONObject("sys").getString("country"));

        JSONObject details = json.getJSONArray("weather").getJSONObject(0);
        JSONObject main = json.getJSONObject("main");
        detailsField.setText(
                details.getString("description").toUpperCase(Locale.US) +
                "\n" + "Humidity: " + main.getString("humidity") + "%" +
                "\n" + "Pressure: " + main.getString("pressure") + " hPa");

        currentTemperatureField.setText(
                    String.format("%.2f", main.getDouble("temp"))+ " ℃");

        DateFormat df = DateFormat.getDateTimeInstance();
        String updatedOn = df.format(new Date(json.getLong("dt")*1000));
        updatedField.setText("Last update: " + updatedOn);

        setWeatherIcon(details.getInt("id"),
                json.getJSONObject("sys").getLong("sunrise") * 1000,
                json.getJSONObject("sys").getLong("sunset") * 1000);

    }catch(Exception e){
        Log.e("SimpleWeather", "One or more fields not found in the JSON data");
    }
}
```

Рисунок 3.17 – Використання методу renderWeather для поновлення тексту в додатку

В кінці методу renderWeather ми викликали setWeatherIcon з ідентифікатором поточної погоди, а також часом сходу і заходу сонця. Налаштування значка погоди трохи складна, тому що API OpenWeatherMap підтримує більше погодних умов, ніж потрібно б було відобразити за допомогою використовуваного веб-шрифту. Однак, ідентифікатори погоди слідують шаблоном, про який можна дізнатися більше на веб-сайті OpenWeatherMap.

За даним принципом (приблизно) зіставляли ідентифікатор погоди з іконкою:

1) коди погоди в діапазоні 200 позначають грозу, що означає, що могли б бути використати R.string.weather_thunder для них

2) коди погоди в діапазоні 300 пов'язані з мряки − використовуємо R.string.weather_drizzle

3) коди погоди в діапазоні 500 означають дощ, одже використовуємо R.string.weather_rain і так далі ...

Було використано час сходу і заходу сонця, щоб відобразити сонце або місяць, в залежності від поточного часу доби і тільки в тому випадку, якщо погода ясна.

```
01  private void setWeatherIcon(int actualId, long sunrise, long sunset){
02      int id = actualId / 100;
03      String icon = "";
04      if(actualId == 800){
05          long currentTime = new Date().getTime();
06          if(currentTime>=sunrise && currentTime<sunset) {
07              icon = getActivity().getString(R.string.weather_sunny);
08          } else {
09              icon = getActivity().getString(R.string.weather_clear_night);
10          }
11      } else {
12          switch(id) {
13          case 2 : icon = getActivity().getString(R.string.weather_thunder);
14                   break;
15          case 3 : icon = getActivity().getString(R.string.weather_drizzle);
16                   break;
17          case 7 : icon = getActivity().getString(R.string.weather_foggy);
18                   break;
19          case 8 : icon = getActivity().getString(R.string.weather_cloudy);
20                   break;
21          case 6 : icon = getActivity().getString(R.string.weather_snowy);
22                   break;
23          case 5 : icon = getActivity().getString(R.string.weather_rainy);
24                   break;
25          }
26      }
27      weatherIcon.setText(icon);
28  }
```

Рисунок 3.18 – Зміна іконки погодного показника за допомогою setWeatherIcon

Звичайно ж можна б обробляти більше погодних умов, відповідно змінюючи метод setWeatherIcon.

Нарешті, додаємо метод changeCity до фрагменту, щоб користувач міг оновити поточний місто. Метод changeCity буде викликатися тільки з основного класу Activity.

```
3   }
5       nbqafeWesfneDafa(city);
1   bnpjjc void cpguBscjfv(Strinu cjfv){
```

Рисунок 3.19 – Використання методу changeCity для зміни міста

Протягом настройки проекту, Eclipse заповнює WeatherActivity.java деяким шаблонним кодом. Замінюємо стандартну реалізацію методу onCreate на наведену нижче, в якій використовуємо WeatherFragment. Метод onCreate виглядає наступним чином:

```java
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_weather);

    if (savedInstanceState == null) {
        getSupportFragmentManager().beginTransaction()
                .add(R.id.container, new WeatherFragment())
                .commit();
    }
}
```

Рисунок 3.20 – Оновлення методу onCreate

Потім відредагували метод onOptionsItemSelected і обробили єдиний параметр меню, який мали. Все, що потрібно було зробити − це викликати метод showInputDialog.

У методі showInputDialog ми використовували AlertDialog.Builder для створення об'єкта Dialog, який пропонує користувачеві ввести ім'я міста. Ця інформація передається методу changeCity, який зберігає ім'я міста з використанням класу CityPreference і викликає метод changeCity у фрагменті.

```java
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    if(item.getItemId() == R.id.change_city){
        showInputDialog();
    }
    return false;
}

private void showInputDialog(){
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setTitle("Change city");
    final EditText input = new EditText(this);
    input.setInputType(InputType.TYPE_CLASS_TEXT);
    builder.setView(input);
    builder.setPositiveButton("Go", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            changeCity(input.getText().toString());
        }
    });
    builder.show();
}

public void changeCity(String city){
    WeatherFragment wf = (WeatherFragment)getSupportFragmentManager()
                                .findFragmentById(R.id.container);
    wf.changeCity(city);
    new CityPreference(this).setCity(city);
}
```

Рисунок 3.21 – Створення об'єкта Dialog у методі showInputDialog

ВИСНОВКИ

Результати проведеного дослідження дають підставу зробити такі висновки:

- у першому розділі описуються основні поняття, пов'язані з погодою і також з проблемою її прогнозування. У цьому ж розділі наведені мобільні додатки для отримання прогнозу - розглянуті їх можливості.
- у другому розділі розглянуто основні питання щодо можливостей API-сервісу OpenWeather та його корисність для розробників додатків, а також запропонований спосіб створення власної метеостанції. Також була розглянута методологія використання різних API ключів.
- у третьому розділі буде розглянутий принцип створення мобільного додатку за допомогою Eclipse і Java-мови на базі платформи Android.

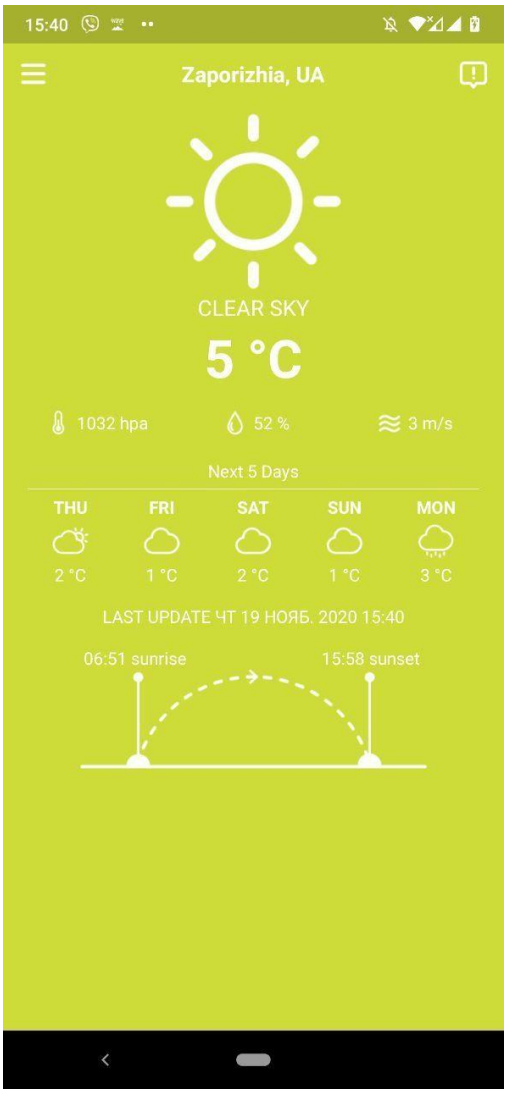
На основі виконаних завдань, рекомендується скласти учбовий матеріал, в якому описується основні відомості мобільного додатку, принцип його реалізації та деякі нюанси його створення бажаною мовою програмування.

ПЕРЕЛІК ПОСИЛАНЬ

1. Прогноз погоди – матеріал з Вікіпедії − вільної енциклопедії. URL:https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D0%BD%D0%BE%D0%B7_%D0%BF%D0%BE%D0%B3%D0%BE%D0%B4%D0%B8 (дата звернення : 20.10.2020)
2. 10 красивых приложений с точными прогнозами погоды. URL: https://lifehacker.ru/tochnye-prognozy-pogody-android-ios/ (дата звернення : 22.10.2020)
3. Прогноз погоды своими руками. URL: https://habr.com/ru/post/117140/ (дата звернення : 22.10.2020)
4. Алгоритмы прогнозирования погодных условий в системах сбора и обработки метеорологических данных (Н.Н. Гринченко, В.Ю. Потапова, А.С. Тарасов). URL: http://masters.donntu.org/2019/fknt/kravchenko/library/article6.htm (дата звернення : 19.11.2020)
5. OpenWeatherMap – как энтузиасты делают погоду. URL: https://habr.com/ru/post/164045/ (дата звернення : 19.11.2020)
6. Сценарий использования API прогнозных сервисов. URL: https://starkovden.github.io/using-api-scenario.html (дата звернення : 21.11.2020)
7. Иконки погодных условий Эрика Флорса. URL: https://github.com/erikflowers/weather-icons (дата звернення : 21.11.2020)

Додаток А

Розділ головного меню в додатку



Додаток В

Розділ обраних користувачем міст

Додаток С

Можливість вибору міста (де знаходиться доступна метеостанція)

Додаток D

Зміна кольору фона при зміні погоди на ясну