НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ

Завідувач кафедри

Комп'ютерних наук

_____ Голуб Б.Л.

"_____" _____ 20 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему

«Програмне забезпечення для інформаційної системи аналізу моніторингу якості повітря»

Спеціальність 121 – «Інженерія програмного забезпечення»

Гарант освітньої програми

К.т.н., доцент _____ Вайганг Г.О.

(науковий ступінь та вчене звання) (підпис) (ПІБ)

Керівник бакалаврської кваліфікаційної роботи

К.ф.-м.н., доцент _____ Британ А. В.

(науковий ступінь та вчене звання) (підпис) (ПІБ)

Виконав _____ Фіяло А. В.

(підпис) (ПІБ студента)

КИЇВ – 2025

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри
комп'ютерних наук
(назва кафедри)
_____ Голуб Б.Л.
(підпис) (ПІБ)

"16" грудня 2024 р.

З А В Д А Н Н Я
на виконання бакалаврської кваліфікаційної роботи студенту
Фіялу Анатолію Васильовичу
(прізвище, ім'я, по батькові)

Спеціальність 121 – «Інженерія програмного забезпечення»

ОПП «Інженерія програмного забезпечення»

Тема бакалаврської кваліфікаційної роботи: Програмне забезпечення системи обліку пацієнтів в районній поліклініці
затверджена наказом ректора НУБіП України від "16" грудня 2024 р.
№ 2249"С"
Термін подання завершеної роботи на кафедру 2025.05.25
(рік, місяць, число)
Вихідні дані до бакалаврської кваліфікаційної роботи: опис організації та змісту роботи амбулаторно-поліклінічних закладів
Перелік питань, які потрібно розробити:
1. Ознайомлення з методиками розрахунку індексів якості повітря
2. Проєктування інформаційного забезпечення
3. Розробка програмного забезпечення
4. Рекомендації щодо впровадження та експлуатації системи

Дата видачі завдання "16" грудня 2024 р.

Керівник бакалаврської кваліфікаційної роботи

К.ф.-м.н., доцент _____ Британ А.В.
(науковий ступінь та вчене звання) (підпис) (ПІБ)

Завдання прийняв до виконання _____ Фіяло А.В.
(підпис) (ПІБ студента)

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – база даних

ІЯП – індекс якості повітря

НФ – нормальна форма

ПЗ – програмне забезпечення

СУБД – система управління базами даних

API – програмний інтерфейс

AQI – індекс якості повітря

CAQHI – національний індекс якості повітря та здоров’я

CN-AQI – китайський стандарт індексу повітря

CSRF - Cross-Site Request Forgery, тип атаки, через яку користувач несвідомо запускає шкідливий запит

CSS – каскадні таблиці зі стилями

DDoS - розподілена атака системи з подальшими наслідками перевантаження серверу

EPA – агенство для охорони навколишнього середовища США

HTML – мова для розмітки гіпертексту

HTTP – основний протокол передачі гіпертексту в мережах

HTTPS – теж саме що й HTTP, тільки захищений

JSON –JavaScript Object Notation, текстовий формат даних для обробки машиною

MVC – модель – подання – контролер, архітектурний шаблон побудови програмного забезпечення

NAQI – національний індекс для якості повітря Японії

UK-DAQI – щоденний індекс для відображення якості повітря

UML – універсальна мова моделювання

XSS - Cross-Site Scripting, тип атаки, який вбудовує шкідливий код у вебсторінку

ВСТУП

Забруднення повітря, без перебільшення, є однією з найгостріших проблем сучасності у сфері глобальної охорони здоров'я, щорічно спричиняючи до мільйонів випадків передчасної загибелі. Всесвітня організація охорони здоров'я (ВООЗ) підкреслює, що переважна більшість населення планети дихає повітрям, рівень забруднення якого перевищує офіційні, встановлені нею, безпечні межі. Вплив шкідливих речовин, таких як тверді частинки, оксиди азоту, озон, оксиди сірки та монооксид вуглецю, може спричинити серйозні проблеми зі здоров'ям, починаючи від респіраторних та серцево-судинних захворювань і закінчуючи розвитком онкологій різного типу та скороченням тривалості життя. Навіть нетривалий вплив забрудненого повітря може загострити існуючі захворювання, зокрема бронхіальної астми або ХОЗЛ (хронічного обструктивного захворювання легень), тоді як довготривалий вплив значно збільшує ймовірність виникнення і розвитку хронічних патологій. [1]

Окрім негативного шкідливого впливу на здоров’я людини, забруднення повітря також завдає значної шкоди для навколишнього середовища, впливаючи негативно на екосистеми, скорочуючи біорізноманіття, стаючи причиною кислотних дощів, руйнуючи аграрні культури та ліси, а також підсилюючи зміну клімату [2].

На жаль, Україна стикається зі значними проблемами у сфері атмосферного забруднення та визнана однією з найбільш постраждалих країн Європи. Київ, столиця держави, доволі часто страждає від високого рівня забруднення повітря, періодично входячи до переліку міст з найбільш забрудненим повітрям у світі. Головними джерелами забруднення повітря в Києві є викиди транспорту, діяльність промислового сектору (включаючи теплоелектростанції та виробничі підприємства), будівельні роботи та часті пожежі (зокрема, лісові та торф'яні). Крім того, сезонні температурні коливання також можуть суттєво ускладнювати ситуацію з якістю повітря. Війна, що триває в країні, безумовно, ще більше

поглибила проблему через руйнування промислових об’єктів і виникнення додаткових джерел забруднення, внаслідок бойових дій [3].

Аналіз значних масивів інформації, що надходить з різноманітних джерел моніторингу якості повітря, як правило, потребує спеціалізованих інструментів для результативного виявлення закономірностей, тенденцій та можливих аномалій. Складність структури й обсяг зібраних даних щодо стану атмосфери зумовлюють потребу у надійних засобах аналітики. Створення спеціалізованого програмного забезпечення, здатного об'єднувати дані з різнотипних джерел (державних, громадських, супутникових) в єдину платформу, безсумнівно, забезпечить повніше та точніше розуміння якості повітря в Києві та навколишніх районах. Своєчасне й точне опрацювання даних про якість повітря має ключове значення для поінформованості населення про стан повітря, розробки та впровадження заходів контролю забруднення, оцінки ефективності екологічної політики, підтримки наукових досліджень та розробок у сфері управління якістю повітря. Враховуючи поточні проблеми в функціонуванні системи моніторингу якості повітря в Україні а також значні наслідки забруднення повітря для здоров'я людей і довкілля, розробка ефективного та надійного програмного забезпечення для обробки даних є не лише вкрай актуальною, а й необхідною для вирішення цієї системної критично важливої проблеми [4].

Метою розробки такого програмного забезпечення є створення інтуїтивно зручної користувацької платформи, яка значною мірою полегшить процес отримання, обробки та аналізу та структурування інформації щодо стану якості повітря отриманої з різних джерел, тим самим скорочуючи витрат часу й зусиль, які зазвичай потрібні для таких операцій. Дане програмне забезпечення міститиме інструменти які дадуть змогу користувачам швидко ідентифікувати основні тенденції та закономірності у зміні рівня забрудненості повітря. Аналітичні можливості програми дозволятимуть отримати індекси якості повітря (ІЯП) розраховані за різноманітними методами та стандартами.

Веб-орієнтований характер програми гарантуватиме оперативний доступ зацікавленим сторонам до найновіших даних та звітів аналізу якості повітря з

будь-якого місця, де доступний Інтернету. Система буде налаштована спеціально для обробки даних з різноманітних систем моніторингу в режимі, максимально наближеному до реального часу, надаючи актуальні показники забруднення повітря. Механізми перевірки та контролю якості даних, а також система повідомлень та застережень, будуть впроваджені для забезпечення якості та точності інформації, що надається користувачам.

Таким чином, надаючи детальну та структуровану інформацію про якість повітря, програмне забезпечення сприятиме прийняттю державними установами обґрунтованих рішенньщодо регуляції екологічних норм та стратегій контролю забруднення. Дані, отримані за допомогою програмного забезпечення, можуть бути використані для реалізації профілактичних ініціатив у сфері охорони здоров'я, таких як видача попереджень у періоди підвищеного забруднення та надання рекомендацій щодо захисту населення. Крім цього, платформа може слугувати корисном інструментом для дослідників, які аналізують джерела, масштаби та наслідки забруднення, та сприяють розробці та запровадженню нових стратегій зменшення шкідливих наслідків. [5]

Розробка веб-застосунку відкриє широкий доступ до платформи для користувачів з різних пристроїв (настільних комп'ютерів, ноутбуків, планшетів) без необхідності встановлення спеціального програмного забезпечення. Окрім того, веб-інтерфейс забезпечить централізоване керування даними та оновленнями, спрощуючи супровід і впровадження нових функцій. Такий формат роботи сприятиме взаємодії та обміну даними між усіма учасниками процесу моніторингу якості повітря.

Python є широко використовуваною мовою програмування, вирізняється своїм потужним інструментарієм: численними бібліотеками та фреймворками для аналізу даних, наукових обчислень та веб-розробки, що робить її ідеальним вибором для реалізації проєкту. Її простота синтаксису та зрозуміла логіка полегшать роботу команди розробників та потенційно навіть науковців, які працюють з даними. Потужна підтримка спільноти Python забезпечує доступ до широкої бази знань, документації та готових рішень, що суттєво прискорює

розробку й спрощує усунення можливих труднощів.

Flask - це сучасний, високопродуктивний Python веб-фреймворком, спеціально розробленим для оперативної та ефективної побудови API (інтерфейсів прикладного програмування). Він підтримує такі функції, як автоматична перевірка даних, серіалізація та генерація деякої документації до API, що певною мірою оптимізує процес розробки та підвищує надійність системи.

PostgreSQL є надійною реляційною системою керування базами даних з відкритим вихідним кодом, відомою за своєю надійністю, стабільністю та масштабованістю, роблячи її оптимальним рішенням для зберігання великих обсягів інформації про якість повітря. Вона підтримує розширені типи даних та функції, зокрема обробку геоданих, що є особливо цінним для просторового аналізу забруднення повітря. До того ж, PostgreSQL має активну спільноту та велику кількість документації, що гарантує довготривалу підтримку та легкий доступ до ресурсів.

Апробація програмного додатку відбувалася шляхом участі в науково-практичній конференції «Теоретичні та прикладні аспекти розробки комп’ютерних систем», що відбулася на базі факультету ІТ НУБіП, де було представлено доповідь на тему «ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ АНАЛІЗУ МОНІТОРИНГУ ЯКОСТІ ПОВІТРЯ».

Записка містить в собі 73 сторінки, список з 45 джерел даних, та додатки А, Б, В, Д, Е, Ж, К.

Перелік умовних позначень являє собою список скорочень абревіатур та спеціальних термінів, що зустрічаються у роботі.

Вступ має опис актуальності теми, мети, завдання, об'єкту і предмету дослідження, методів та інструментів, що використовуються, та деталей щодо записки.

Розділ Дослідження предметної сфери для розробки програмного забезпечення системи моніторингу та аналізу якості повітря містить в собі

основні поняття предметної області якості повітря, опис та деталізацію індексу якості повітря, а також містить список функціональних та нефункціональних вимог до ПЗ. Має в собі моделювання предметної області проекту з визначенням структури предметної області, основних сутностей, UML діаграм проєкту, огляд подібних систем та джерел даних, і в кінці сформульовано конкретні завдання для розробки системи на основі проаналізованих в розділі даних.

Розділ Проектування інформаційного та програмного забезпечення містить в собі опис логічної моделі даних, діаграми класів та діаграми компонентів

Розділ Розробка інформаційного та програмного забезпечення містить в собі обґрунтування вибору системи керування базами даних, розробку структури БД визначення інструментарію для реалізації програмного забезпечення проекту та опис покрокового процесу реалізації серверної і клієнтської частини системи

Розділ 4 Методичні вказівки для впровадження та технічної експлуатації програмного проекту містить в собі опис тестування та сценарії тестування, сформовані апаратні та програмні вимоги системи та описано структуру інсталяційного набору.

У висновках підсумовано результат роботи та зроблено огляд досягнутих цілей.

Список використаних джерел містить в собі перелік різних джерел та посилань на них, що використані в написанні роботи.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ СФЕРИ ДЛЯ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ МОНІТОРИНГУ ТА АНАЛІЗУ ЯКОСТІ ПОВІТРЯ

1.1 Аналіз специфіки сфери якості повітря

1.1.1 Огляд поняття якості для атмосферного повітря

Якість атмосферного повітря — є комплексною характеристикою для стану повітряного середовища, яка визначається певним вмістом у ньому фізичних, хімічних і біологічних речовин, що можуть прямо чи побічно мати великий вплив на здоров'я людей, стан навколишнього середовища (довкілля) та стабільність загальних кліматичних процесів. Саме цей показник є результатом певної взаємодії між природними і антропогенними чинниками та зазвичай оцінюється через концентрації різних забрудників, таких як дрібнодисперсні частинки (PM10, PM2.5), діоксид азоту (NO_2), озон (O_3), чадний газ (CO), сірчистий ангідрид (SO_2), а також леткі органічні сполуки [6].

1.1.2 Характеристика індексу якості повітря

Індекс якості повітря (ІЯП, або ж англійською скорочено AQI) — це стандартизований числовий показник, який відображає певний визначений рівень забруднення атмосферного повітря та безпосередньо пов'язаний із ним рівень ризику для здоров'я людини, та який обчислюється маючи за основу концентрації ключових забрудників (наприклад таких як озон, дрібнодисперсні частинки, чадний газ, діоксид сірки, діоксид азоту) і результат обчислень може бути співставлений та визначений за категоріями від «добре» до «надзвичайно небезпечно» та використовувати визначену кольорову індикацію для зручності

інтерпретації інформації громадськістю [7].

При цьому і основне завдання індексу якості повітря (далі – ІЯП) заключається в агрегації даних про концентрації кількох визначених основних забруднювачів повітря в єдиний індекс-показник, який зазвичай легко сприймається громадськістю. Саме для цього в системах з використанням ІЯП зазвичай використовуються так звані «брейкпойнти» – тобто граничні значення концентрацій для кожного індивідуального забруднювача, які відповідають певним встановленим рівням якості повітря, наприклад таким як "добрий", "помірний" або ж "нездоровий". Використання саме певних граничних значень забезпечує стандартизоване незалежно тлумачення якості повітря залежне від конкретних концентрацій забруднювачів, що є над важливим для розробки рекомендацій у області охорони здоров'я. Важливо також зазначити, що різні країни та регіони зазвичай часто розробляють особисті системи ІЯП, адаптовані під специфіку пов’язану з внутрішніми екологічними умовами, переліком основних забруднювачів та регуляторних вимог. Саме ці відмінності відображають усе різноманіття спектру проблем пов’язаних з забрудненням повітря та пріоритетів у різних частинах світу, підкреслюючи цим необхідність в локалізованих підходах до оцінки якості повітря. Також це неодмінно вказує на важливість порівняння один з одним, різних індексів якості повітря задля встановлення максимально точного, або ж іншими словами «істинного» впливу на здоров’я людини. [8]

1.1.3 Основні параметри якості повітря

Далі надано зручний перелік з основних забруднювачів та деяких допоміжних показників, що зазвичай використовуються з метою оцінити стан повітря.

Основні забруднювачі:

1. Дрібнодисперсні частинки PM2,5 – певні речовини, а точніше тверді або рідкі аерозолі з діаметром ≤2,5 мкм, які мають здатність проникати в

альвеоли і при цьому спричиняти системне запалення й призводити до серцево-судинних захворювань [22, с. 74].

2. Дрібнодисперсні частинки PM10 – також речовини, аерозолі діаметром уже ≤ 10 мкм, що осідають у верхніх дихальних шляхах викликаючи цим подразнення тих же дихальних шляхів та створюючи ризики для виникнення бронхіту [22, с. 89].

3. Озон (O_3) в районі земної поверхні – вторинний забруднювач, що створюється в результаті реакцій оксидів азоту та деяких летких органічних сполук через дію на них сонячного світла, і викликає оксидативний стрес для тканин дихальних шляхів [22, с. 97].

4. Діоксид азоту (NO_2) – це в нас газ із групи NO_X, який утворюється під час згорання палива, та спричиняє характерні запальні процеси у легенях що таким чином підвищує ризик астми [22, с. 111].

5. Діоксид сірки (SO_2) – має утворення внаслідок спалювання та згоряння вугілля та нафтопродуктів. Своєю дією подразнює слизові оболонки та може трансформуватися в опади кислоти [22, с. 125].

6. Чадний газ (CO) – є безбарвний, без запаху, продукт утворений за рахунок неповного згоряння вуглеводнів. Він з’єднується з гемоглобіном, знижуючи можливість транспортування кисню в організмі [22, с. 130].

Розширення (доповнення до основних) забруднювачів:

7. Вуглекислий газ (CO_2) –парниковий газ, який стосується лише закритих приміщень; високі концентрації можуть явно вказувати на недостатню вентиляцію в приміщеннях [23].

8. Леткі органічні сполуки (VOC) – включають формальдегід, толуол, бензол тощо і можуть довготривало накопичуватися й спричиняти складні канцерогенні ефекти [23].

Допоміжні показники:

9. Температура повітря – зазвичай значно впливає на хімічні перетворення забруднювачів, і також, дифузію в атмосфері. Має вимірювання в градусах Цельсія °C або ж градусах Фаренгейта °F.

10. Відносна вологість – визначає кількість пари (водяної) в повітрі; впливає на властивості гідратації частинок, і спектр аерозолів.

11. Атмосферний тиск – відображає масу повітря яке скупчене над певною точкою (наприклад датчиком) і його коливання можуть сприяти застою стану забруднювачів.

12. Швидкість та напрямок вітру – вони визначають собою перенесення і розсіювання забруднювачів та вимірюються в м/с та градусах, відповідно.

13. Опади (дощ, сніг) – сприяють осіданню забруднюючих частинок із атмосфери. Варто враховувати що їх облік необхідний щоб аналізувати і прогнозувати короткострокові зміни у якості повітря.

1.1.4 Стандартизація значень та способи розрахунку індексів якості повітря в різних країнах світу

Якщо розглянути способи розрахунку ІЯП для різних країн, можна зауважити, що вони по своїй суті переважно будуються саме на стандартній формулі для так званого суб-індексу, тобто обчислення виконується для кожного забрудника окремо. Важливо, що при розрахунку враховуються конкретні межі (або як їх ще називають - границі) для певних забруднюючих речовин, так звані «пороги», які визначені стандартами певної країни. Ці пороги зазвичай змінюються в залежності від певного списку умов, серед яких є і середньозважений для регіону стан забруднення повітря. Тобто для країн котрі мають більші за інші країни проблеми з забрудненням атмосферного повітря, зазвичай показники порогових значень на порядок вище.

Отже, розглянемо саму формулу для обрахунку суб-індексу для кожного конкретного забрудника [9, с. 13]:

$$I = \frac{I_{high} - I_{low}}{C_{high} - C_{low}} \left(C - C_{low}\right) + I_{low}, \quad (1.1)$$

де:

I – сам суб-індекс забрудника

C – виміряна його концентрація

C_{high} і C_{low} – найближчі верхня та нижня межі значення концентрації для відповідного забрудника

I_{high} і I_{low} – відповідно, межі індексації (верхня та нижня межі самого AQI)

У США для алгоритму обчислення ІЯП беруться до уваги концентрації шести головних забруднювачів: приземного озону, твердих частинок (PM2.5 та PM10), чадного газу, діоксиду сірки та діоксиду азоту. Методика розрахунку передбачає собою обчислення індивідуального індексу для кожного забруднювача на основі спеціальних брейкпойнтів, встановлених для різних рівнів концентрації. Загальний ІЯП визначається як найвищий з індивідуальних індексів забруднювачів. Для швидкого та точного відображення якості повітря в реальному часі, особливо щодо PM2.5 та озону, використовується алгоритм NowCast, який надає більшу пріоритетність крайнім даним спостережень. Різні проміжки значень ІЯП відповідають певним рівням якості повітря та супроводжуються інформацією щодо можливого впливу на здоров'я. Таблицю діапазонів граничних меж конценрацій для індексу цієї країни можна переглянути в додатку А [8]

Вирахування національного індексу якості повітря (NAQI) Індії, як відомо, включає вісім основних забруднювачів: PM2.5, PM10, діоксид азоту, діоксид сірки, чадний газ, озон, аміак та свинець. Для кожного з них розраховується субіндекс, а загальний NAQI визначається за найгіршим значенням субіндексу. Різні категорії NAQI супроводжуються конкретними рекомендаціями для населення щодо захисту здоров'я залежно від рівня забруднення. Таблицю діапазонів граничних меж конценрацій для індексу цієї країни можна переглянути в додатку В. [10]

Китайська система ІЯП в свою чергу здійснює моніторинг концентрацій PM2.5, PM10, діоксиду сірки, діоксиду азоту, чадного газу та озону. Розрахунок, як відомо, включає визначення індивідуального індексу якості повітря (IAQI) для кожного забруднювача. Загальний ІЯП визначається як максимальне значення серед усіх IAQI. Різні рівні ІЯП в Китаї мають відповідні конкретні рекомендації для населення щодо поведінки при різних рівнях забруднення.

Таблицю діапазонів граничних меж конценрацій для індексу цієї країни можна переглянути в додатку Б. [11]

Канадський індекс якості повітря та здоров'я (AQHI) фокусується на оцінці ризиків для здоров'я, які в свою чергу пов'язані зі з’єднанням трьох основних забруднювачів: озону, PM2.5 та діоксиду азоту. AQHI має певну шкалу від 1 до 10+, що відповідає різним рівням ризику для здоров'я мешканців: низькому, помірному, високому та дуже високому. У деяких провінціях Канади можуть враховуватися й інші забруднювачі, що відображає регіональні особливості забруднення повітря. Таблицю діапазонів граничних меж конценрацій для індексу цієї країни можна переглянути в додатку Д. [12]

Щоденний індекс якості повітря (DAQI) у Великій Британії, зокрема, використовує шкалу від 1 до 10, яка базується на концентраціях ключових забруднювачів: озону, діоксиду азоту, діоксиду сірки, PM2.5 та PM10. Індекс розраховується на основі найвищої концентрації будь-якого з цих забруднювачів відносно встановлених для них діапазонів. Для різних рівнів DAQI надаються конкретні поради щодо здоров'я для різних груп населення. Таблицю діапазонів граничних меж конценрацій для індексу цієї країни можна переглянути в додатку Е [13]

1.1.5 Огляд доступу до інформації про якість повітря в Україні

На сьогоднішній день в Україні діє дві рівноправні системи моніторингу: державна, яка керується Міністерством захисту довкілля та природних ресурсів і здійснюється через мережу стаціонарних постів Гідрометцентру (ЦГО ім. Срезневського), та громадська, що представлена незалежними мережами сенсорів й онлайн-платформами.

Нагляд у держаній структурі ведеться на 129 стаціонарних постах, де проби зразків атмосферного повітря відбираються 3–4 рази на добу шість днів на тиждень; дані порівнюються з гранично допустимими концентраціями (ГДК) забруднювачів. [14]

Створення громадського моніторингу запровадили SaveEcoBot, Airly, Eco-City, YourAirTest та інші, створивши понад 500 автономних сенсорних станцій у більш ніж 150 містах України; дані актуалізуються щогодини й знаходяться у відкритому доступі на певних онлайн картах. Такі платформи дають можливість користувачам налаштовувати повідомлення про погіршення якості атмосфери, подавати скарги на викиди та переглядати аналітику за певний період. [15]

У нормативних документах України вирахунок Індексу якості повітря (ІЯП) організований таким чином [16]:

1. Початкові дані беруть як середні концентрації забруднювачів за годинну, виміряні на стаціонарних пунктах спостереження з використанням референс-методів або індикативних методів з забезпеченням достовірності інформації.

2. Вибір списку забруднювачів залежить від деяких особливостей типу пункту спостереження:

- Для транспортно-орієнтованих пунктів застосовуються дані щодо концентрацій PM2.5, PM10 та NO_2;
- Для промислових та міських пунктів —про NO_2, O_3 та PM2.5 і /або PM10.

3. Обчислення індексів індивідуально для кожного забруднювача проводиться згідно з системою, визначеною в Наказі №590-2025, яка містить чітко означені табличні градації концентрацій і відповідних значень суб-індексу.

4. Створення загального ІЯП відбувається як найбільше значення серед усіх розрахованих суб-індексів, та має відповідність до рекомендації ВООЗ і EPA щодо агрегованих індексів якості повітря.

5. Екранізація результату обчислень відбувається кольоровим кругом і текстовим індикатором згідно з категоріями, затвердженими в Наказі № 590-2025, також має особистий супровід рекомендаціями для загального та чутливого населення.

Отже, основними забруднювачами з яких формується ІЯП в Україні є такі речовини, як:

Озон;

ТЧ2.5 - тверді частки;

ТЧ10 - тверді частки;

Діоксид сірки;

Діоксид азоту. [16]

1.1.6 Актуальність моніторингу для якості повітря

Сучасний моніторинг для якості атмосфери є надзвичайно важливим для захищення здоров'я населення, планування урбаністичних стратегій, зменшення екологічних ризиків та доступного й оперативного інформування громадськості. Регулярне відстеження концентрацій забруднювачів дає можливість оперативно виявляти небезпечні спалахи, усвідомлювати локальні й глобальні тенденції, а також розробляти дієві заходи для покращення стану якості повітря.

Високі показники рівню забруднення атмосферного повітря спричиняють значну деградацію екосистем: зниження біорізноманіття, отруєння ґрунтів і водних систем, а також скупчення токсичних небезпечних сполук у харчовому ланцюгу [17]. Водночас, шкідливі гази та аерозолі призводять до утворення кисневого дефіциту в пограничних шарах атмосфери, що погіршує процеси фотосинтезу і впливає на продуктивність рослин [1]. Модельні дослідження виявляють, що вкраплення промислових і транспортних викидів місцево змінюють хімічний склад повітря, створюючи «гарячі точки» забруднення, які можуть поширюються на десятки кілометрів.

Довготривала експозиція забрудненого повітря асоціюється зі збільшенням ризику розвитку для серцево-судинних і деяких респіраторних захворювань, негативно має вплив на центральну нервову систему та сприяє онкогенезу [18]. Діти, літні люди та вагітні жінки мають підвищену вразливість до часток PM2.5 і PM10, які можуть проникати в середину альвеоли та надходити в кровотік, спричиняючи тим самим запалення й окислювальний стрес. Короткострокові викиди (пожежі, промислові аварії) збільшують захворюваність на астму та

бронхіти, в той час як хронічне забруднення — провокує розвиток хронічного обструктивного захворювання легень (ХОЗЛ) і ішемічної хвороби серця. Також, деякі дослідження виявили визначені зв’язки між впливом деяких типів забруднюючих повітря речовин і зростаючим коефіцієнтом виникнення лейкемії, різних типів раку, розладами репродуктивної та імунної системи. [19]

Глобальні мережі моніторингу і фіксації дають загальну картину, але для прийняття місцевих рішень потрібні більш густі сенсорні мережі й інформація з мінімальною затримкою. Ком’юніті-моніторинг залучає мешканців до монтажу невеликих портативних станцій, що дає змогу виявляти «гарячі точки» в мікрорайонах, поблизу доріг або великих підприємств. Спостереження поведінкових моделей показує, що персоналізовані сповіщення та інтерактивні карти підвищують обізнаність і знижують індивідуального ризику, спонукаючи людей берегти своє здоров’я [20].

Забруднення атмосфери не лише шкодить здоров’ю, але й прискорює зміну клімату через парникові гази й аерозолі, які здійснюють вплив на радіаційний баланс Землі [1]. Економічні втрати через зниження продуктивності працівників і збільшення бюджету на охорону здоров’я можуть сягати мільярдів доларів щорічно, і навіть спостереження в рамках програм по зменшенню викидів від роботи різних підприємств показують зменшення витрат на медичне обслуговування та підвищення продуктивності праці після введення обмежень викидів [21].

1.1.7 Суб’єкти сфери моніторингу якості повітря

Сфера моніторингу даних щодо якості повітря охоплює широкий спектр різних суб'єктів, що мають певну потребу у зборі, обробці, інтерпретації та використанні даних про стан атмосферного повітря. Такі суб'єкти можна класифікувати враховуючи їхню роль та функції, а також зацікавлення. До основних суб'єктів можна віднести:

Органи державної влади та місцевого самоврядування. Сюди можуть

увійти міністерства, які відповідальні за довкілля та охорону здоров'я, служби з надзвичайних ситуацій та органи місцевого самоврядування. Їх зацікавленість полягає в тому що вони можуть використовувати дані для формування державної політики, створення нормативно-правових документів, регулювання законодавства, введення різноманітних реєстрів.

Певні спеціалізовані державні установи. Це можуть бути певні станції або лабораторні центри, що здійснюють спостереження та оцінюють ризики для здоров'я населення, повідомляють про небезпеку.

Науково-освітні та дослідні заклади. Університети, спеціалізовані науково-дослідні інститути. Вони проводять різні дослідження, розробляють різні методики, моделюють процеси які пов'язані з забрудненням повітря, та готують фахівців у вузьких галузях.

Підприємства. Це можуть бути об'єкти котрі самі є джерелами викидів шкідливих речовин. Більшість країн, згідно з законодавством, зобов'язані проводити моніторинг виробничих викидів і регулювати власний вплив на забруднення повітря.

Громадські організації. Зазвичай неприбуткові організації, що займаються питанням охорони навколишнього середовища.

Громадяни. Оскільки громадськість безпосередньо на собі відчуває вплив забруднення, це спонукає як отримувати дані моніторингу, так і встановлювати власні датчики та надавати їх дані у відкритий доступ.

1.2 Аналіз вимог до архітектури програмної системи

1.2.1 Функціональні вимоги

1. <u>Отримання деталей щодо якості повітря</u>

1.1. Система має одержувати актуальні дані з визначених інтегрованих джерел (а саме API моніторингу, бази даних, та певних імпортованих з різних форматів даних).

1.2. Користувач повинен мати змогу бачити індекс якості повітря (AQI) у певній геолокації (відповідно за станціями моніторингу).

1.3. Користувачу повинно бути доступно перегляд деталізованих параметрів (за наявності таких в джерелі даних), а саме:

- PM2.5
- PM10
- CO
- NO_2
- SO_2
- O_3
- Вологість, температура, тиск (як необов'язкові додаткові дані)

1.4. Режим автоматичного оновлення даних щодо забруднення через визначений інтервал

1.5. Можливість перевірки достовірності наданої інформації (наприклад, джерело або ж останній час оновлення)

2. Підтримка розрахунку кількох стандартів AQI

2.1. Розрахунок індексу якості повітря повинен відбуватися відповідно до таких наступних стандартів:

- AQI США (EPA)
- AQI Китаю (CN-AQI)
- AQI Індії (NAQI)
- AQI Великої Британії (UK-DAQI)
- AQI Канади (CAQHI)

3. Інтерактивна візуальна карта

3.1. Система має показувати інтерактивну карту з позначенням на ній станцій для моніторингу стану повітря.

3.2. Для кожної заданої точки на карті повинно бути доступно відображення спливаючих вікон з поточними показниками, тобто розгорнута інформація при кліку на конкретний маркер (AQI, параметри, джерело).

3.3. Створювати запит до користувача про надання дозволу на визначення місцеперебування (координат) користувача.

3.4. Система має мати змогу автоматично визначити місцезнаходження для користувача, за умови якщо дозвіл на доступ до місцезнаходження надано.

3.5. Користувач може вручну вибрати будь-яку локацію на інтерактивній мапі.

4. *Рекомендації*

4.1. Автоматичне створення та показ порад на основі поточного виділеного AQI, що включатиме всі категорії, або ж одну з них:

- Безпечність для активностей на відкритому повітрі
- Поради щодо носіння маски (потрібно одягати чи ні).
- Поради щодо провітрювання приміщень

5. *Доступність та локалізація*

5.1. Інтерфейс повинен мати підтримку української мови.

5.2. Присутність адаптації під різні розміри екранів.

1.2.2 Нефункціональні вимоги

Вимоги безпеки

Захист персональних даних:

- Всі персональні дані про користувачів мають зберігатися відповідно до вимог GDPR/ЗУ «Про захист персональних даних».

Шифрування:

- Вся передача даних повинна бути захищена за допомогою з’єднання через HTTPS.
- Критично важливі дані в базі даних мають бути зашифровані (опціонально — геолокація, токени).

Стійкість до атак:

- Система має бути захищена від мінімального ряду типових атак: SQL

Injection, XSS, CSRF, DDoS.

- Обмеження числа запитів (так званий rate limiting) для захисту доступу до API.

Вимоги до продуктивності системи

Швидкість завантаження контенту:

- Головна (основна) сторінка програмної системи має завантажуватись в мінімально короткий термін, а саме менш ніж за 2 секунди.

Відповідь API:

- Звичайний середньостатистичний час відгуку API не повинен перевищувати 500 мілісекунд.

Обробка навантаження:

- Система має підтримувати одночасну взаємодію з щонайменше 1000 активних користувачів без вагомих втрат продуктивності.

Вимоги до надійності системи

Доступність:

- Система має бути доступною протягом мінімум 99 % часу

Відновлення:

- Після збоїв система повинна мати ручний механізм відновлення

Вимоги до портативності системи

Крос-браузерне розгортання:

- Система має працювати коректно в різних браузерах, принаймні в наступних: Chrome, Firefox, Safari, Edge.

Вимоги до написання коду системи

Правила написання коду:

- Написання програмного коду має відбуватись згідно зі стандартами (PEP8 для Python)
- Присутні детальні коментарі до складних для розуміння частин коду

Модульність архітектури:

- Архітектура ПЗ має надавати певні можливості для легкого додавання функцій

Вимоги до інтерфейсу системи

- Інтерфейс користувача повинен бути інтуїтивно зрозумілим, з легким та мінімальним навчанням
- Дизайн інтерфейсу має бути адаптивним та коректно відображатися на пристроях з різним розміром екрану
- Вибір кольорів і шрифтів інтерфейсу має відбуватись враховуючи тематику програмної системи.

1.3 Моделювання предметної області

1.3.1 Огляд поняття моделювання предметної області

Моделювання предметної області – процес розробки концептуальної моделі для певної предметної області, яка в свою чергу є відображенням ключових правил, понять, відношень без урахування прикладної (технічної) сторони реалізації [24].

Моделювання програмної системи є важливим та невід'ємним етапом процесу її розробки. Метою моделювання є не тільки створення зрозумілих для розробників діаграм, а й у тому щоб вирішити критично важливі завдання, необхідні для успішного створення та впровадження програмного продукту. Можна виділити кілька основних цілей моделювання:

В першу чергу моделі виконують роль так званого «містка» для усіх учасників розробки продукту, тобто спрощують розуміння для замовників, менеджерів, бізнес-аналітиків, архітекторів, розробників та тестувальників. Моделі у вигляді діаграм та схем є більш зрозумілими. На відміну від текстових описів у яких кожен з профільних спеціалістів буде використовувати зрозумілі для його напряму діяльності терміни, діаграми є уніфікованою мовою для різних учасників розробки, що в свою чергу допоможе дійти згоди у спільному розумінні усіх частин системи. Це критично важливо для роботи великих команд або проєктах що базуються на розподіленій модульній розробці.

На сьогодні системи зазвичай є достатньо складними, оскільки вони базуються на багатьох різних компонентах, яким потрібно взаємодіяти між собою. Тут моделювання надає можливості для зосередження лише на головному, абстрагуючись від менш важливих дрібниць, а також надає можливість поглянути на архітектуру проєкту з різних рівнів абстракції – починаючи з логіки окремих підсистем або модулів і розширюючись аж до загального вигляду архітектури системи. Це дозволяє кожному учаснику зосередитись на тій частині проєкту, за яку він відповідальний, без перевантаження непотрібною йому інформацією, але й не забувати про загальну архітектуру і внесок його частини в роботу системи.

Також моделювання дозволяє ще на ранніх стадіях розробки виявити потенційні вузькі місця та ризики, що може значно здешевити та пришвидшити розробку, в порівнянні з виправлення помилок під час написання коду, тестування, або в найгіршому варіанті, після впровадження та запуску системи. Тобто таким чином моделювання запобігає виникненню потреби в можливих повторних змінах, котрі пов’язані з неправильним розумінням архітектури та вимог до системи, що значно позитивно впливає на вартість розробки, запуску та подальшої підтримки проєкту.

Моделі зазвичай запроваджуються та вбудовуються безпосередньо в документацію, описуючи як працює система та її окремі частини. Якісна документація в свою чергу сприяє швидкому знайомству нових членів команди з системою, а також має велику цінність в подальшій підтримці та розвитку проєкту.

Загалом за допомогою моделей можна перевіряти узгодженість різних етапів розробки, та їх частини, наприклад:

- Узгодженість та повноту вимог до системи;
- Правильність логіки взаємодії компонентів;
- Відповідність дизайну до вимог продуктивності, надійності, безпеки та інших;
- Правильність логіки бізнес-процесів;

- «Вузькі місця», небезпеки та ризики архітектури

Для створення моделей існує Unified Modeling Language (UML) – уніфікована мова для візуального моделювання. UML не має прив’язки до конкретних методів розробки чи певних технологій, але при цьому дозволяє конструювати візуально вигляд системи або її частин, незалежно від складності усього проєкту. Вона має широкий набір різних діаграм і різних стандартних елементів нотацій, за допомогою яких можна описати як структуру системи, так і поведінку. Наприклад, для опису структури існують такі діаграми, як діаграма класів, пакетів, компонентів. А от для опису поведінки використовуються зазвичай діаграми прецедентів, активності та послідовності. Використання цієї мови зазвичай відбувається протягом усіх етапів життєвого циклу розробки.

1.3.2 Основні сутності та абстракції предметної області

Перед початком створення діаграм варто виділити основні сутності для предметної області даної роботи, а саме:

Користувач – особа що має взаємодію з системою, має на меті отримати дані щодо стану забруднення повітря.

API – джерело даних, яке не є безпосередньою частиною системи, та надає дані про показники забруднення та їх значення.

Геолокація – це місцерозташування, а саме координати, відповідно до яких система робить розрахунки та відображає найближчі доступні станції моніторингу.

Калькулятор AQI – певний програмний модуль, який виконує обробку даних про концентрації показників, та розраховує індекс якості повітря базуючись на стандартах різних країн.

Показники – дані щодо концентрацій забруднювачів відповідно до певної локації вимірювання та часу, коли виміри було отримано.

Повідомлення – певний механізм, що призначений для інформування користувача.

Також не менш важливими є абстракції, їх визначення та роль. Далі для кожної абстракції визначено послуги, які вона надає для інших частин системи (взаємодія з внутрішніми компонентами) або інших систем (взаємодія з зовнішніми компонентами).

Абстракція «Користувач» являє собою самого кінцевого користувача системи моніторингу повітря. Вона зосереджена і слугує для взаємодії користувача з системою. Важливі властивості та обов’язки даної абстракції представлено у табл. 1.1.

Таблиця 1.1

Абстракція: Користувач
Важливі властивості: Геолокація (точна або за замовчуванням) Налаштування сповіщень Обраний стандарт AQI
Обов'язки: Надати дозвіл на доступ до геолокації Переглядати AQI та значення показників Отримувати попередження про стан повітря Отримувати рекомендації щодо прогулянок

Дана абстракція надає такі послуги:

- Передає системі дані щодо згоди або відмови користувача в наданні інформації для визначення свого точного місцеперебування.
- Надає системі дані щодо обраного користувачем стандарту AQI для відображення за замовчуванням.
- Передає до системи налаштування щодо сповіщень (заборона їх відображати, або ж дозвіл)
- Створює запит до системи з метою формування рекомендацій для користувача.

Дана абстракція виконує наступні операції:

- Керує дозволом на отримання геолокації.
- Надає користувачу дані щодо AQI та окремих показниках, що отримані з зовнішніх API.
- Надає користувачу попередження про небезпеки пов’язані зі станом повітря.
- Надає користувачу рекомендації щодо прогулянок з врахуванням поточного стану повітря та концентрації в ньому різних забрудників.

Абстракція «Геолокація» є базовим джерелом геоданих, надаючи системі певні географічні координати. Важливі властивості та обов’язки даної абстракції представлено у табл. 1.2.

Таблиця 1.2

Абстракція: Геолокація
Важливі властивості: Широта Довгота Статус доступу (наданий / відхилений)
Обов'язки: Надавати координати користувача Повертати геодані (точні або за замовчуванням) Бути джерелом для підготовки запиту до API

Дана абстракція надає такі послуги:

- Надає для певної точки її координати, а саме широту та довготу.
- Надає, якщо отримано дозвіл користувача, попередньо визначене розташування, або ж у разі відсутності дозволу – координати за замовчуванням.
- Формує дані для використання їх в запитах до API, яким потрібні координати.

Дана абстракція виконує наступні операції:

- Надає точні координати за запитом компонентам системи (відповідно до тих, які надані користувачем, або встановлені за замовчуванням)
- Реєструє та зберігає поточний актуальний стан доступу до геоданих користувача.
- Використовує програмні або апаратні (якщо доступно GPS) засоби для отримання точних координат.

Абстракція «API» є умовним інтерфейсом до зовнішнього сервісу, що є основним джерелом даних про якість повітря. Важливі властивості та обов'язки даної абстракції представлено у табл. 1.3.

Таблиця 1.3

Абстракція: API (зовнішнє джерело даних про повітря)
Важливі властивості: URL-адреса Доступність сервісу Структура відповіді Ключ доступу до API
Обов'язки: Приймати запити із координатами Надсилати дані про забруднення повітря (PM2.5, PM10, CO2 тощо) Повідомляти про помилки або недоступність

Дана абстракція надає такі послуги:

- Надає можливість отримати дані показників про забруднення повітря, що надаються певними зовнішніми сервісами.
- Виконує парсинг даних з файлів та відповідей наданих зовнішніми сервісами для подальшого їх використання у системі.
- Інформує про можливість доступу до кожного з зовнішніх сервісів.

Дана абстракція виконує наступні операції:

- Отримує дані з показниками забруднення повітря за допомогою вхідних HTTP-запитів, та опрацьовує їх.
- Надсилає запити до самих API на отримання даних з датчиків, використовуючи конкретні для кожного сервісу структури цих запитів.
- Повідомляє систему про помилки, які виникли при надсиланні або отримані запиту.

Абстракція «Показники» є відображенням даних щодо якості повітря в середині системи. Тут відбувається інкапсуляція зібраної інформації задля її зберігання та використання іншими модулями. Важливі властивості та обов'язки даної абстракції представлено у табл. 1.4.

Таблиця 1.4

Абстракція: Показники
Важливі властивості: Час отримання даних Показники якості повітря (PM2.5, PM10, NO2, O3, CO, тощо) Джерело даних Адреса датчика
Обов'язки: Зберігати отримані від API дані Повертати дані показників

Дана абстракція надає такі послуги:

- Надає доступ до збережених даних – значень різних забруднювачів, а також пов'язану з ними інформацію про час отримання показників та деталі про джерело даних.

Дана абстракція виконує наступні операції:

- Фіксує час отримання та відомості про джерело даних
- Здійснює збереження отриманих даних від API

- Повертає збережені дані про показники на вимогу певних частин системи.

Абстракція «Калькулятор AQI» відповідає за проведення обчислень індексу якості повітря за стандартами різних країн та регіонів на основі наданих «сирих» даних про значення показників. Важливі властивості та обов’язки для даної абстракції наведено у табл. 1.5.

Дана абстракція надає такі послуги:

- Надає обчислене значення ІЯП за різними стандартами на основі заданої формули та даних про показники забруднення.
- Надає граничні значення для кожного з ІЯП для обрахунку

Дана абстракція виконує наступні операції:

- Приймає дані щодо концентрацій забруднювачів від інших компонентів системи.
- Обирає для обрахунку ІЯП відповідну до стандарту формулу та порогові значення.
- Виконує обчислення, та повертає точне значення ІЯП разом з визначеним рівнем небезпеки.

Таблиця 1.5

Абстракція: Модуль розрахунку AQI
Важливі властивості: Обраний стандарт AQI Граничні значення (пороги AQI) Формули розрахунку
Обов'язки: Виконувати розрахунок AQI Визначати рівень небезпеки Повертати числове значення AQI

Абстракція «Повідомлення» є відповідальною за формування та відправку

повідомлень з інформацією для користувачів, надаючи дані про негативні чи позитивні зміни в стані якості повітря. У табл. 1.6 наведено властивості та обов’язки для абстракції.

Дана абстракція надає такі послуги:

- Повідомляє користувача про потенційно небезпечні збільшення концентрації забрудників у повітрі.
- Дає поради які стосуються безпеки здоров’я.

Дана абстракція виконує наступні операції:

- Створює текстове повідомлення врахувавши рівень важливості, канал доставки та тип повідомлення.
- Відправляє текстове повідомлення відповідно по обраному каналу доставки.

Таблиця 1.6

Абстракція: Повідомлення
Важливі властивості: Тип повідомлення Рівень важливості Канал доставки
Обов'язки: Інформувати користувача про погіршення якості повітря Передавати рекомендації або обмеження Актуалізуватися відповідно до нового AQI

Абстракція «Візуалізація даних» є відображенням інформації у графічному вигляді, зрозумілому для користувача. Наносить числові дані на карту та відображає різні візуальні елементи. ЇЇ важливі властивості та обов’язки перераховано у таблиці 1.7.

Ця абстракція надає такі послуги:

- Відображає візуальну карту

- Відображає на карті датчики або станції моніторингу

Ця абстракція виконує наступні операції:

- Отримує дані про якість повітря та координати датчиків, станцій та користувачів.
- Створює візуальні елементи системи, в тому числі карту та відображення розміщення джерел даних у вигляді кольорових кругів.
- Реагує на дії користувача направлені на зміну масштабу карти, вибір точок, натискання на певні елементи.

Таблиця 1.7

Абстракція: Інтерактивна карта
Важливі властивості: Географічне охоплення Позначення точок вимірювання
Обов'язки: Відображати якість повітря по регіонах Візуалізувати місце користувача та інші джерела Підтримувати вибір місця вручну

1.3.3 Діаграма прецедентів для конструюваного ПЗ

Діаграма прецедентів (англ. Use Case Diagram) – частина UML, яка відображає функціональність системи з точки зору її учасників (акторів), демонструючи функції (прецеденти) виконувані системою. Вона відображає насамперед те, що система повинна виконувати, та які зв’язки присутні між користувачами і функціями у системі. Саме тому ця діаграма допомагає виявляти вимоги на ранніх етапах проєктування. [25]

Діаграма має кілька елементів, що виконують свою задану роль:

Прецеденти – є загальним описом функції, яку виконує система, при цьому не розкриваючи та не вказуючи на механізми її виконання.

Актор – є користувачем, що певним чином взаємодіє з системою. Таким користувачем має бути людина або організація, або ж певна зовнішня система чи модуль.

Відношення – зв’язки між прецедентами та акторами є асоціаціями, які демонструють як саме відбуваються ініціації різних сценаріїв функціонування системи.

Межі системи – це рамка, яка окреслює саму систему з прецедентами всередині. При цьому всі актори знаходяться за межами цієї рамки, тобто за межами системи.

Стереотипи – є відображенням зв’язків між прецедентами, які відображають їх залежність одне від одного. Ці зв’язки бувають двох типів - <<include>> та <<extend>>, тобто включаючий зв’язок та зв’язок розширення.

Для створення програмного забезпечення також було спроєктовано діаграму прецедентів, яку зображено на рис. 1.1. Ця діаграма надасть можливість розгляду системи з точки зору взаємодії з користувачем. На діаграмі добре видно які саме дії може виконувати користувач у системі, та як ці дії пов'язані одна з одною.

Рис. 1.1 Діаграма прецедентів для проєкту

Прецеденти що надані безпосередньо в системі є наступними:

Перегляд рівня AQI за допомогою якого користувач має змогу переглядати значення індексу якості повітря. Цей прецедент має два розширення, а саме:

Вибір стандарту AQI, тобто користувач може обрати потрібний йому стандарт ІЯП, за яким буде відображено значення для індексу.

Перегляд показників якості повітря, що включає в себе перегляд деталей усіх значень доступних показників (наприклад концентрація дрібнодисперсних часток, озону, діоксиду сірки, тощо), перегляд яких доступний як окрема опція.

Також прецедент «Перегляд рівня AQI» має розширення у вигляді прецеденту «Інтерактивна карта».

Інтерактивна карта відображає певні дані щодо стану якості повітря, а саме індекс AQI у вказаних областях. Цей прецедент в свою чергу має розширення у

вигляді прецеденту «Визначення геолокації».

Визначення геолокації є опціональною функцією, котра дозволяє визначити певне місцерозташування, включаючи також місцезнаходження користувача, при умові що він надав на це дозвіл.

Отримання сповіщень – цей прецедент вказує на те, що користувач може отримувати різні сповіщення від системи. Це можуть бути як сповіщення про погіршення стану повітря, так і про поради для здоров’я та рекомендації щодо часу та місць прогулянок.

1.3.4 Діаграма послідовності

Діаграма послідовності є однією з UML діаграм. Вона слугує для моделювання взаємодії між об’єктами у певному часовому порядку. Це надає можливість відобразити поведінку системи відповідно до хронології, задля визначення деталей взаємодії між різними компонентами. В той же час вона відображає які повідомлення та коли надсилаються між компонентами. [26]

Діаграма послідовності зазвичай використовується для опису деталей реалізації окремих прецедентів або операцій, що допомагає кожному члену команди розуміти очікувані результати та порядок здійснюваних викликів. Саме через це налагоджується спільне розуміння функціональних вимог до певних частин системи між аналітиками та розробниками вказуючи та визначаючи те, які реакції на дії акторів повинно мати програмне забезпечення. Також це дозволяє виявити вузькі місця пов’язані з часовими залежностями, і водночас перевірити логіку для взаємодії компонентів, що в подальшому вплине на саму реалізацію коду. Дана діаграма дуже допомагає з визначенням методів, котрі мають бути реалізовані відповідно до класів. Це створює міцне підґрунтя для документації поведінки системи. [26]

Для діаграми послідовності можна виділити кілька ключових елементів [26]:

Лінії життя, або ще їх називають учасниками – це вертикальні пунктирні

лінії, що вказують на існування певних акторів чи об’єктів відносно до часу.

Повідомлення – це вже горизонтальні лінії зі стрілками між різними лініями життя. Стрілки можуть бути кількох типів:

- Якщо ці лінії суцільні із заповненим трикутником у вигляді стрілки на кінці, то це значить що вони використовуються для синхронних викликів.
- Якщо ж стрілки зображені у вигляді трикутника без однієї сторони та без заливки, то це вказує на асинхронний виклик.
- Якщо стрілка у вигляді трикутника без однієї сторони та без заливки, і при цьому сама лінія пунктирна, то цей тип повідомлення є відповіддю.

Також існує стрілка, що виходить та повертається до однієї лінії життя, показуючи повідомлення, яке об’єкт або актор відправляє собі ж самому.

Прямокутники, що розміщені на лінії життя, показують інтервал активності об’єкта або актора у часі, тобто час виконання певних дій або методів. Їх зазвичай називають активаціями.

Системні рамки окреслюють межі певного розглядуваного сценарію по відношенню до зовнішніх систем.

Діаграма має кілька фрагментів комбінацій, що використовуються у випадках коли потрібно змоделювати складну логіку. Існують такі фрагменти, як:

Option – це блок з однією умовою, що виконується якщо вона є істинною.

Alternative – це блоки з умовами, за істинності яких виконується тільки істинний блок, всі інші не виконуються.

Loop – це блок, що має виконуватися кілька разів та при цьому може мати певну умову, при істинності якої він буде завершувати своє виконання, тобто відбуватиметься переривання.

Parallel – це блоки виконання яких відбувається паралельно.

Reference – при великій складності та нагромаджені об’єктів в діаграмі дозволяє використовувати посилання на інші діаграми послідовності, що

дозволяє значно спростити поточну діаграму.

Як правило, діаграма послідовності має таку визначену послідовність: Спочатку один із акторів діаграми ініціює якесь повідомлення до одного з елементів системи. Далі об’єкти виконують обробку цих повідомлень та надсилають повідомлення до інших об’єктів та можуть надсилати відповіді до акторів. Таким чином за допомогою повідомлень відображається часова взаємодія різних об’єктів системи. Також коли об’єкт виконав свою певну дію, то він на лінії життя позначається як той, що обірвав активність. В кінцевому результаті тоді коли мета досягнута, то сценарій завершується.

Для прецедентів перегляду інформації щодо індексу якості повітря та перегляду показників повітря спроєктовано діаграму послідовності, яку зображено на рисунку 1.2. Діаграма відображає часовий потік повідомлень між об’єктами для виконання запиту користувача.

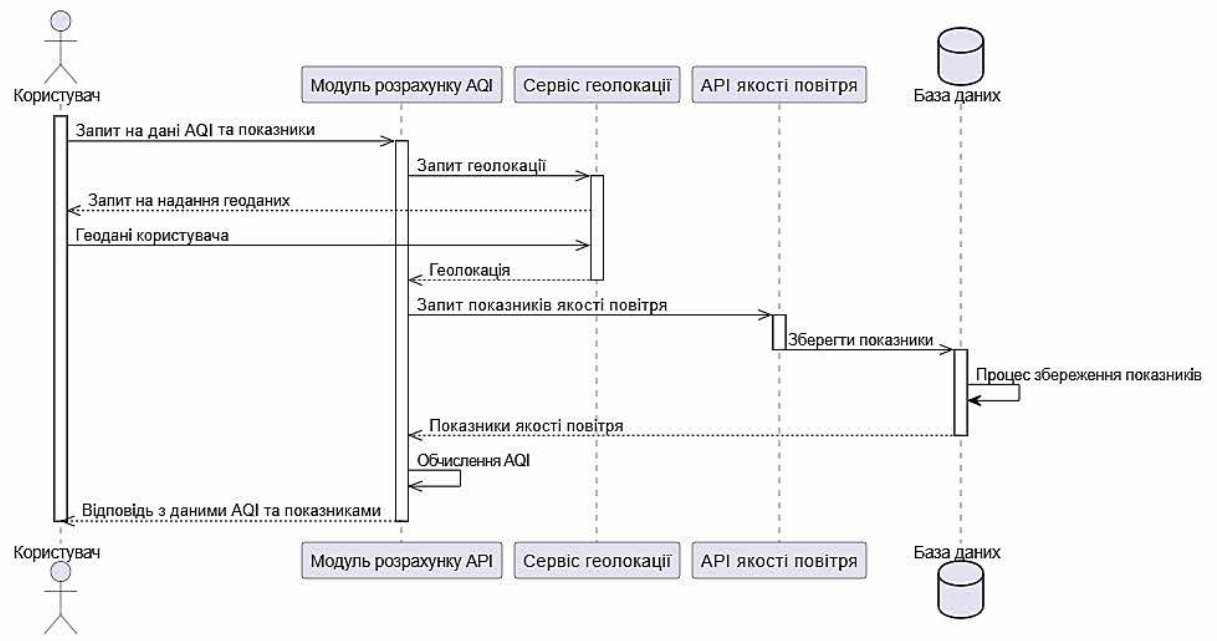

Рис. 1.2 Діаграма послідовності для прецедентів проєкту

Користувач – зовнішній актор який безпосередньо взаємодіє з системою. Він надсилає повідомлення з запитом на отримання даних щодо значення індексу якості повітря.

Модуль розрахунку AQI виступає як компонент системи, що безпосередньо відповідає за обрахунок ІЯП і час його діяльності для опису цих прецедентів є

досить довгим.

Сервіс геолокації є компонентом що відповідає за визначення координат для геолокації користувача або певного визначеного місця.

API якості повітря є адаптером що надає дані про забруднення повітря для системи.

База даних є сховищем для зберігання та передавання значень отриманих деталей показників.

Оскільки діаграма послідовності читається зверху до низу, то послідовність дій для даної діаграми буде такою:

1. Користувач надсилає у вигляді асинхронного повідомлення запит на отримання даних AQI до Модулю розрахунку AQI на отримання даних показників якості повітря.
2. Модуль розрахунку AQI надсилає повідомлення з запитом геолокації до сервісу геолокації, який в свою чергу надсилає асинхронне повідомлення з запитом на надання геоданих до користувача.
3. Користувач надсилає геодані назад до сервісу геолокації, після чого сервіс геолокації визначає координати перебування користувача та надсилає їх до модулю розрахунку AQI. Далі цей модуль формує повідомлення з запитом показників якості повітря до API якості повітря.
4. API якості повітря виконує запит, отримуючи дані, та надсилає повідомлення з показниками до бази даних, яка в свою чергу виконує процес збереження показників та самостійно в асинхронному режимі надсилає відповідь до модулю розрахунку API з даними щодо показників якості повітря.
5. Після цього модуль розрахунку API виконує обчислення індексу якості повітря та надсилає повідомлення з відповіддю що містить дані про AQI та показники користувачу.

1.4 Огляд інформаційних джерел та існуючих рішень

Аналіз існуючих рішень дозволяє виявити сильні та слабкі сторони уже існуючих систем на ринку, та спираючись на їхні помилки поліпшувати різні аспекти модельованої системи.

Оглянувши функціональні можливості різних існуючих рішень, можна виділити деяку інформацію щодо їх роботи. Такий сервіс як SaveEcoBot має більше 500 станцій для моніторингу в Україні з оновленням даних в інтервалі годину, та має інтеграцію з відкритими джерелами. В свою чергу такий сервіс як Airly надає рішення під ключ для бізнесу та муніципалітетів, маючи за основу мережу державних та приватних сенсорів, формуючи аналітику і надаючи бізнес-підписки. Eco-City виділяє значимість для легкості підключення нових станцій, що надаються самими користувачами на платформу з відкритим доступом до інформації. YourAirTest, існує як науковий стартап, та моделює шкідливі викиди, і на основі моделей пропонує прогнози та імітації. [27, 28, 29, 30, 31, 32, 33, 34]

При цьому великі глобальні компанії такі як IQAir надають дані у вигляді 3D карт та інтеграцією даних з супутника, тобто комбінують різні дані з різних джерел. Ploom Labs використовує для злиття даних із близько 12 000 станцій машинне навчання, і при цьому випускає пристрій для індивідуального моніторингу під назвою Flow. BreezoMeter базується на API з прогнозами терміном до чотирьох днів та інтеграцією технології heatmap-шарів для сторонніх карт. OpenAQ виступає в ролі агрегатора відкритих даних. [27, 28, 29, 30, 31, 32, 33, 34]

Для взаємодії з даними більшість з оглянутих сервісів використовують RESTful API за допомогою якого отримують у сирому вигляді концентрації забруднювачів та метадані станцій. Наприклад OpenWeatherMap надає дані щодо різних показників таких як: CO, NO2, O3, NH3, PM2.5, PM10. IQAir надає US-AQI та CN-AQI як показники розраховані за різними стандартами. BreezoMeter в свою чергу надає додаткові рекомендації для здоров’я. Краудсорсингові мережі

(такі як SaveEcoBot, Eco-City, Airly) надають за допомогою GSM/Wi-Fi - підключені сенсори та станції, дані з яких передаються до центральних сервісів для відповідної обробки, а саме агрегації та валідації. Наукові стартапи YourAirTest та Plume Labs користуються в своїй роботі моделями емісіонного моделювання та ML-алгоритмами для виконання корекції недоступних, втрачених чи неточних замірів показників. [27, 28, 29, 30, 31, 32, 33, 34]

Форматом візуалізації для платформ (SaveEcoBot, Airly) є інтерактивні карти з точами та popup-вікнами, а в Eco-City навіть надано можливість для додавання станцій користувачами. IQAir (AirVisual) надає, як вказувалося раніше, 3D-карти та heatmap, висвітлює історичні діаграми й надає рейтинги міст. BreezoMeter та Plume Labs мають фокус на мобільних версіях, тобто додатках з персоналізованими під користувача віджетами та різними сповіщеннями. В усіх випадках можна виділити те, що ключовим є адаптивність дизайну, легка навігація та відповідно до стандартів - кольорова індикація категорій AQI. [27, 28, 29, 30, 31, 32, 33, 34]

Якщо ж розгядати проблеми існуючих систем, то наприклад для локальних сенсорних мереж існують втрати даних через нерівномірне покриття датчиками та низьку точність пов'язану з калібрування (похибка в районі 15 %). Також усі API-сервіси мають ліміти запитів, пов'язані з безпекою та відмовостійкістю, та недоступність під час технічних робіт. Також різні методики та стандарти обчислення ІЯП ускладнюють порівняння та інтерпретацію результатів на різних платформах, що явно вказує на необхідність у підтримці кількох стандартів в одному UI. Візуалізація даних про показники за допомогою теплокарти може «розмити» певні деталі в різних зонах, через що варто комбінувати кілька шарів для відображення інформації. [27, 28, 29, 30, 31, 32, 33, 34]

1.5 Постановка завдання

Розробити веб-сервіс, а саме програмне забезпечення для інформаційної системи аналізу моніторингу якості повітря, яке забезпечить автоматичний збір,

обробку, зберігання, візуалізацію та інтерпретацію даних про стан якості повітря, та використанням кількох стандартів різних країн для розрахунку ІЯП та формуванням рекомендацій і сповіщень.

Мета проєкту. Створити надійну та зручну інформаційну систему, яка дозволить зацікавленим користувачам (громадянам, дослідникам та муніципалітетам) швидко отримувати і виконувати порівняння індексу якості повітря за стандартами США (EPA), Китаю (CN-AQI), Індії (NAQI), Великої Британії (UK-DAQI), Канади (CAQHI) і національним, а також здійснювати планування діяльності на відкритому повітрі з урахуванням безпосередніх ризиків для здоров’я.

Основними завданнями можна визначити:

1. Інтеграція джерел даних. Налаштувати отримання (імпорт) та підтримку "сирих" показників (PM2.5, PM10, NO2, SO2, O3, CO, температура, вологість, тиск - за доступністю) через відкриті сервіси даних про стан повітря, тобто різні API
2. Геолокація. Реалізувати визначення місцезнаходження для користувача в автоматичному режимі з можливістю ручного встановлення альтернативної точки на карті.
3. Обробка та зберігання даних. Спроектувати реляційну схему бази даних (PostgreSQL) для зберігання отриманих первинних даних параметрів вимірювань та результатів обчислень.
4. Розрахунок AQI. Реалізувати модуль, що дозволить обчислювати індекси якості повітря за п’ятьма міжнародними стандартами; надати можливість для порівняння результатів розрахунків.
5. Візуалізація. Надати відображення інтерактивної карти з відповідними маркерами для кожної станції.
6. Рекомендації та сповіщення. Розробити систему правил та шаблонів для автоматичної генерації рекомендацій (наприклад план прогулянок, обмеження активностей) і push-сповіщень при змінах якості повітря.

7. Інтерфейс користувача. Забезпечити адаптивний UI/UX дизайн для веб- та мобільних пристроїв (екранів різного розміру та формату) з обов'язковою підтримкою української мови.

2 ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Логічна модель даних: ER-діаграма

Поняття логічної моделі даних (Logical Data Model) включає в себе опис сутностей, їхніх атрибутів та зв’язків між ними без врахування прив’язки до конкретної СУБД чи технологій, таким чином виконуючи роль «каркасу» для подальшої фізичної реалізації бази даних. Вона утворює собою деякий проміжний рівень між концептуальною схемою та самою фізичною базою даних, фокусуючись лише на бізнес-сутностях, їх зв'язках та властивостях, при цьому не беручи до уваги інформацію про конкретні типи даних, індекси чи фізичні таблиці. За допомогою логічної моделі можна легко узгодити бізнес вимоги та розробку БД завдяки тому, що як бізнес-аналітики та і розробники мають змогу створити спільну візуалізацію інформаційних потоків та взаємозв'язків між ними без огляду технічних деталей. [35]

ER-діаграма являється стандартною графічною нотацією, яка зазвичай використовується для моделювання структури даних реляційних БД, і при цьому відображає сутності, їх атрибути та асоціації. Ці діаграми мають є основою при переході від логічної до фізичної моделі даних і дозволяють візуально відобразити структуру майбутньої БД. [36]

Включають в себе відображення таких елементів як [36]:

Сутності (англ. Entities) - об'єкти або ж певні поняття реального світу, про які необхідно зберігати інформацію. На діаграмі позначаються прямокутником.

Атрибути (англ. Attributes) - є описом властивостей для сутності.

Зв’язки (англ. Relationships) - є відображенням асоціацій між сутностями, зображуються лініями з кардинальностями та підписами для ідентифікації зв'язку. Кардинальність в свою чергу визначає кількість екземплярів сутності,

котрі пов'язані з екземплярами іншої сутності, і зазвичай мають відношення один до одного, один до багатьох, або ж багато до багатьох.

ERwin — це програмний продукт, який без перебільшення є галузевем стандартом для моделювання даних, при цьому підтримуючи всі три рівні методолігії моделювання – концептуальний, логічний і фізичний. Його функцінал включає потужний набір різних функцій спрямованих на створення візуальних моделей, документування і керування базами даних. Має підтрику повного життєвого циклу моделювання. [37]

Саме для створення логічних моделей даних Erwin має широкий інструментарій. Для пришвидшення моделювання його інтерфейс є досить простим, та має широкий функціонал, що включає в себе технологію «перетягування» (англ. drag and drop) для елементів діаграм. Присутня підтримка різних нотацій ER-діаграми, що дозволяє обрати зрозумілий стиль для відображення моделі. Можна створити та розмістити на полотні сутності, та для кожної з них детально визначити атрибути їх імена, типи даних, визначити первинні та зовнішні ключі (англ. - Primary key, Foreign key). Дозволяє з легкістю налаштовувати зв’язки, визначати вид зв’язку (кардинальність) та автоматично генерувати зовнішні ключі. [37]

Для проєктованого програмного забезпечення на рисунку 2.1 зображено ER -діаграму, що представляє собою логічну модель даних.

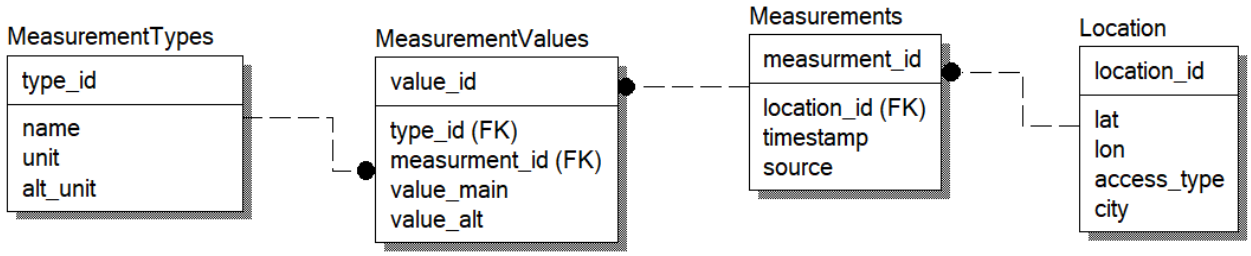

Рис. 2.1 ER-діаграма для моделювання проєкту

Діаграма містить в собі чотири сутності (таблиці) та відповідні зв'язки між ними.

Сутність «MeasurementTypes» призначена для зберігання даних про різні забруднюючі речовини, які можна виміряти. Ця таблиця служить як довідкова,

та забезпечує стандартизацію разом з уникненням дублювання даних.

Має такі атрибути:

- type_id (Первинний ключ - PK): Це унікальний ідентифікатор-ключ для кожного із типів вимірюваних речовин (наприклад, T001 для PM2.5, T002 для CO, і так далі).
- name: Назва вимірюваної речовини - забрудника
- unit: Основна одиниця вимірювання що використовується для заданого типу забрудника.
- alt_unit: Для точності використання даних - альтернативна одиниця вимірювання

Має такі зв'язки:

MeasurementTypes має зв'язок "один-до-багатьох" з MeasurementValues. Це вказує на те, що один тип забруднювача може мати багато різних значень вимірювань, що будуть міститися в таблиці MeasurementValues. Первинний ключ таблиці MeasurementTypes мігрує до неключових атрибутів таблиці MeasurementValues.

Сутність "MeasurementValues" призначена для зберігання точних числових значень для виміряних показників якості повітря (концентрацій забруднювачів). Вона містить певне числове значення та пов'язує його з типом забрудника та конкретними даними щодо вимірювання.

Атрибути:

- value_id (Первинний ключ - PK): Це унікальний ідентифікатор-ключ для кожного окремого запису вимірювання.
- type_id (Зовнішній ключ - FK): Має посилання на поле type_id у таблиці MeasurementTypes, визначаючи до якого типу забруднювачів належить запис зі значенням.
- measurement_id (Зовнішній ключ - FK): Має посилання на поле measurement_id у таблиці Measurements, пов'язуючи запис зі значенням з певними метаданими щодо вимірювання.
- value_main: Саме числове значення вимірювання.

- value_alt: Числове значення для альтернативної одиниці вимірювання, якщо така існує або значення може бути конвертоване з інших одиниць вимірювання.

Має такі зв'язки:

MeasurementValues має зв'язок "багато до одного" з таблицею MeasurementTypes через поле type_id.

MeasurementValues має зв'язок формату "багато до одного" з таблицею Measurements через поле measurement_id і вказує на те, що один запис вимірювання (в Measurements) може мати багато записів значень (у MeasurementValues) для різних забруднювачів.

Сутність "Measurements" призначена для збереження відомостей про так звану "подію" вимірювання якості повітря. Тобто виконується збирання метаданих про саме вимірювання, а не про якісь інші конкретні значення.

Атрибути:

- measurement_id (Первинний ключ - PK): Це унікальний ідентифікатор-ключ для кожної "події" вимірювання.
- location_id (Зовнішній ключ - FK): Має посилання на поле location_id у таблиці Location пояснюючи де було зроблено певне вимірювання.
- timestamp: Це просто дата і час, коли було здійснено певне вимірювання. Є важливим атрибутом для відслідкування точності та актуальності даних, що надаються системою.
- source: Джерело даних з яких взято значення про показники певного факту вимірювання.

Має такі зв'язки:

Measurements має зв'язок "один до багатьох" з таблицею MeasurementValues і вказує на те, що певна "подія" вимірювання показників (у Measurements) може мати багато пов'язаних записів зі значеннями (у MeasurementValues) для різних забруднювачів. Measurements має зв'язок у вигляді "багато до одного" з Location через поле location_id і означає, що одна локація може мати багато записів вимірювань, зроблених у ній.

Сутність "Location" призначена для зберігання географічних даних про місцезнаходження, де виконувалися вимірювання показників для якості повітря. За її допомогою можна зв'язати певні вимірювання з конкретними точками на карті.

Атрибути:

- location_id (Первинний ключ - PK): Це унікальний ідентифікатор-ключ для кожного окремого місця.
- lat: Значення широта для координат
- lon: Значення довготи для координат
- access_type: Тип доступу до джерела вимірювання даних
- city: Назва містав якому було виконано певне вимірювання (для зручності відображення даних).

Має такі зв'язки:

Location має зв'язок формату один до багатьох з таблицею Measurements. Тобто одна локація може мати зв'язок з багатьма вимірюваннями.

Усі чотири сутності (таблиці) мають призначені первинні ключі (PK) і при цьому не містять вкладених або повторюваних атрибутів, тому дана модель є реляційною.

Спроєктована модель ER-діаграми також повністю відповідає 3-й нормальній формі (3НФ), оскільки:

Відповідність до 1НФ є, тому що всі сутності мають чітко визначені первинні ключі (type_id, value_id, measurement_id, location_id), і всі атрибути при цьому є атомарними (неподільними). Інформація не дублюється, через те що кожна концепція зберігається в окремій сутності.

Відповідність до 2НФ присутня, оскільки кожна з сутностей задовольняє 1НФ та всі неключові атрибути в кожній сутності (наприклад, name у MeasurementTypes, lat у Location) мають повну функціональну залежність від усього первинного ключа своєї сутності.

Відповідність 3НФ також виконана, оскільки кожна з сутностей задовольняє 2НФ та при цьому відсутні транзитивні залежності, тобто жоден

неключовий атрибут не залежить від іншого неключового атрибута. Усі неключові атрибути мають залежність безпосередньо від первинного ключа своєї сутності, забезпечуючи цим мінімізацію надмірності даних.

2.2 Діаграма класів

Діаграма класів у мові UML — є видом структурних діаграм, і тому описує структуру системи шляхом візуалізації її класів, атрибутів, методів та зв’язків між об’єктами. Іншими словами, діаграма класів демонструє "будівельні блоки" системи, їх властивості, поведінку та зв'язки між ними. Слугує як основний артефакт при об’єктно-орієнтованому проєктуванні ПЗ, який дозволяє узгодити як бізнес вимоги, так і технічну реалізацію, документувати структуру коду та виконувати роль довідника для розробників і тестувальників. [38]

На рис. 2.2 зображено діаграму класів UML, що візуалізує структуру проєкту. Наочно демонструє основні класи системи, їхні атрибути, операції та взаємозв'язки (асоціації) між ними.

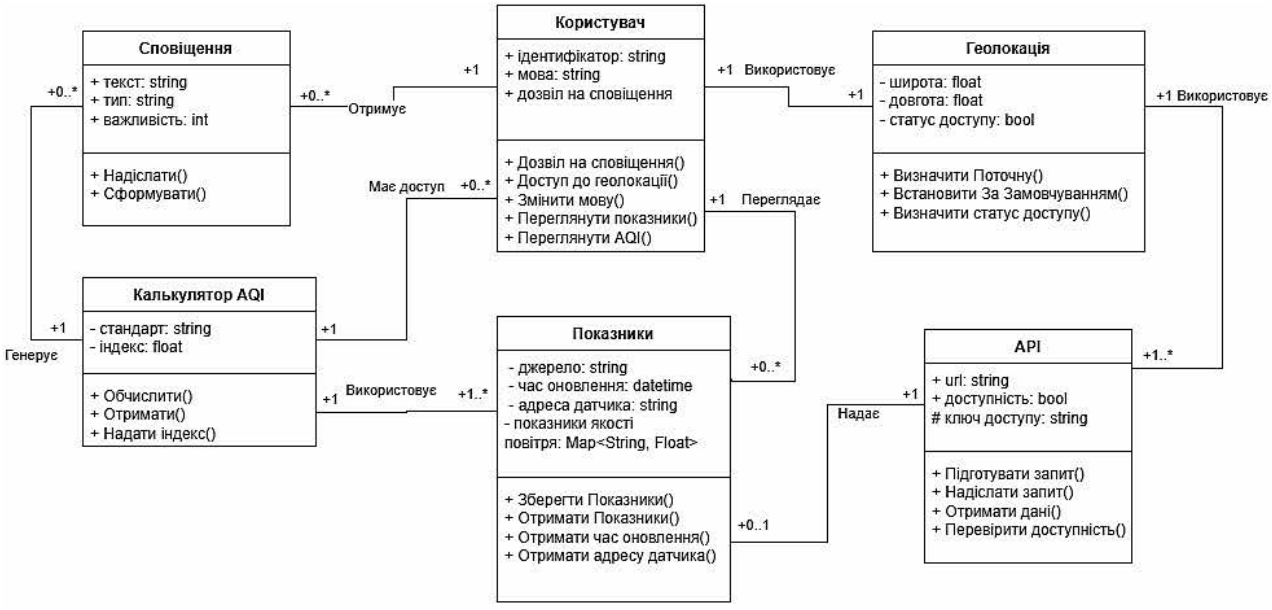

Рис. 2.2 Діаграма класів на основі асоціацій

Діаграма включає в себе шість основних класів кожен з яких включає в себе певну сферу відповідальності.

Користувач, як клас є так би мовити основним актором, що представляє

людину яка працює з системою, та має атрибути такі як ідентифікатор, мова, налаштування інтерфейсу та дозвіл на сповіщення, яким керує користувач. Функції, які виконує даний клас, дозволяють йому керувати дозволом на сповіщення, доступом до геолокації, змінами мови інтерфейсу, та надають можливість ініціювати перегляд показників та перегляд розрахованого AQI.

Геолокація, як клас в свою чергу зберігає інформацію про місцезнаходження, а саме такі атрибути як широту, довготу, статус доступу до геолокації та виконує функції визначення поточної геолокації, встановлення геолокації за замовчуванням якщо дозвіл на визначення не надано, та автоматично визначає статус доступу.

Такий клас як Калькулятор AQI є основним засобом для розрахунку індексу якості повітря відповідно до стандартів різних країн, та має атрибути у вигляді стандарту та індексу, котрий і містить числове значення розрахованого показника, а також має функції для обчислення ІЯП, отримання показників та надання індексу.

В класі Показники зберігаються отримані виміряні дані щодо кожного забруднювача та його значень, а саме в атрибутах: джерело, час оновлення, адреса датчика та самих показників якості повітря. Має функції для збереження показників, ініціації отримання показників, отримання часу оновлення (тобто внесення даних) та отримання адреси датчика з якого й було отримано дані.

Клас API слугує певним містком для отримання даних про якість повітря. Саме тому він містить такі атрибути як URL - який є посиланням на джерело, доступність - що є поточним значенням для доступності сервісу, та ключ доступу - унікальний для кожного окремого API. Має такі функції, як підготовка запиту, надсилання запиту, отримання даних від API, перевірка доступності API на даний момент часу.

Також є клас Сповіщення, який слугує для формування та надсилання повідомлень користувачу. Має такі атрибути як текст сповіщення, тип сповіщення та важливість, а також функціонал який дозволяє надсилати сповіщення та формувати їх.

Зв'язки між класами організовані у вигляді асоціацій та дозволяють деталізувати потік інформації. Все працює приблизно так: один користувач може отримувати багато сповіщень, або ж не отримати жодного; один користувач використовує одну геолокацію; один користувач може переглядати багато наборів даних з показників; калькулятор AQI використовує багато показників; один запит API може надавати жодного або ж один набір показників.

2.3 Діаграма компонентів

Діаграма компонентів є частиною UML що слугує для відображення структури системи на її фізичному рівні. Вона показує з яких частин, а саме компонентів побудовано програмний продукт, та яким чином вони мають взаємодію через інтерфейси та порти. Вона ідеально підходить для відображення модульного розгортання системи та водночас слугує для документування самої архітектури проєкту. [39]

На рис 2.3 зображено діаграму компонентів для проекту, яка демонструє як саме різні функціональні блоки системи взаємодіють одне з одним.

Центральним елементом виступає компонент Користувач, який є представником клієнтської частини системи, що безпосередньо взаємодіє з кінцевим користувачем. При цьому він надає інтерфейс, який дозволяє іншим компонентам надсилати сповіщення користувачеві та одночасно вимагає послуги в компонента Геолокація через інший відповідний інтерфейс.

Сам компонент Сповіщення є відповідальним за генерацію та надсилання сповіщень користувачу, знову ж таки через відповідний інтерфейс. При цьому компонент сповіщення має залежність від іншого компонента, а саме Калькулятор AQI, оскільки йому потрібні актуальні дані про індекс якості повітря або інформація про перевищення граничних значень.

Далі калькулятор AQI, що слугує для обчислення ІЯП, викликає компонент Парсер даних саме через відповідний інтерфейс, тобто завдання з обробки та отримання даних виконує саме цей компонент.

Отже, таким чином компонент Парсер даних є проміжним шаром для збору так званих сирих даних. Він взаємодіє з компонентом База даних через відповідний інтерфейс, і відповідає за збереження отриманих даних в БД та їхнє читання звідти. Також цей компонент має інтерфейс із компонентом Запити до API, що вказує на, знову ж таки, його роль лише як проміжного компоненту для даних.

Компонент запити до API слугує для формування та відправки запиту до зовнішніх сервісів через відповідний інтерфейс.

Загалом, дана діаграма компонентів явно вказує на модульну архітектуру розроблюваного проєкту.

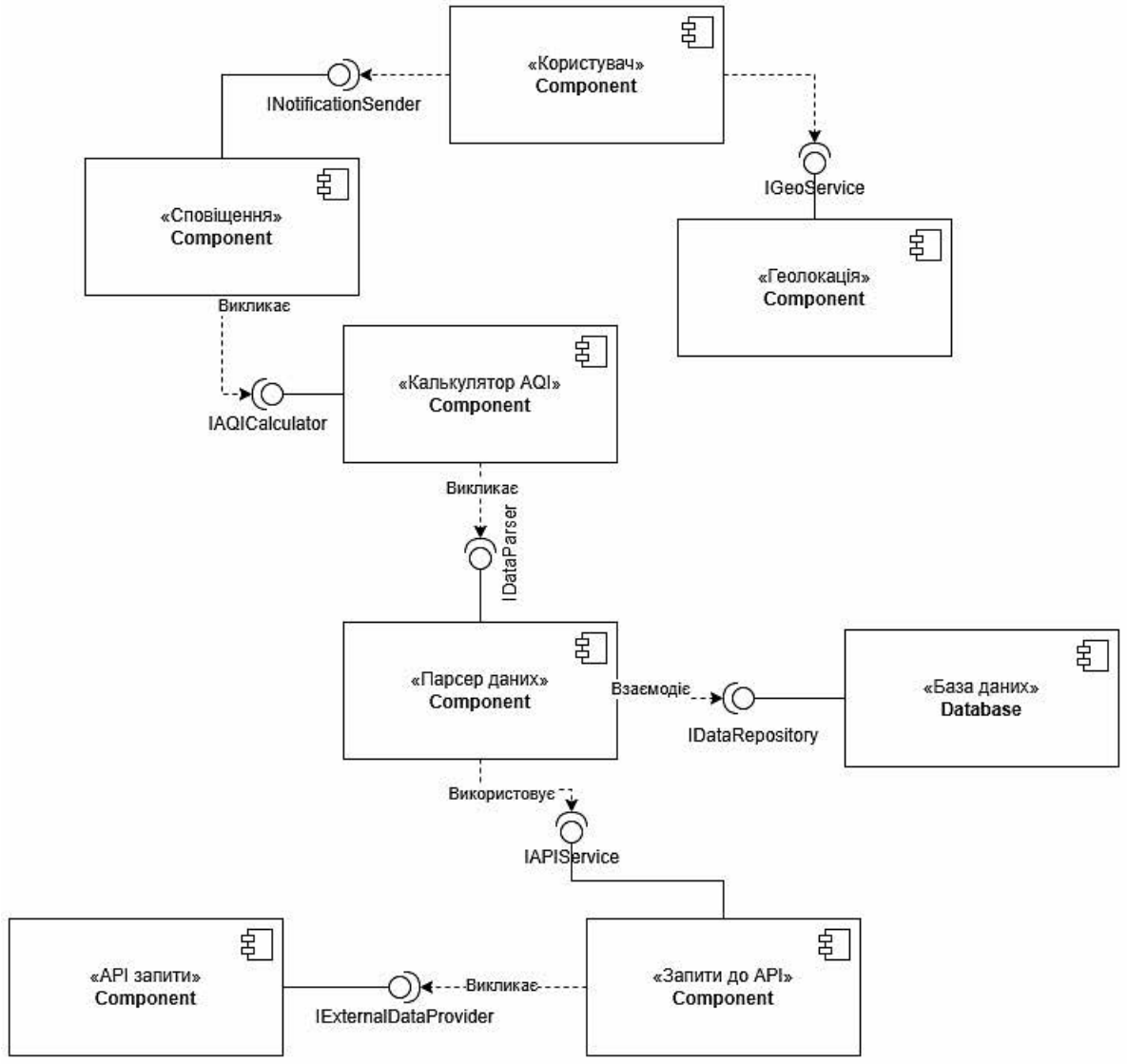

Рис. 2.3 Діаграма компонентів

3 РОЗРОБКА ІНФОРМАЦІЙНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Система для управління інформаційною базою

Система управління інформаційною базою являє собою певну сукупність процедур правил засобів які спрямовані на зберігання та обробку інформації задля узгодженості доступу до даних їхньої цілісності безпеки незалежно від певної реалізації самого фізичного сховища. Таким чином вона має включати в себе моделі даних стандарти для взаємодії які гарантують те що дані будуть коректними та доступними у будь-який час. [40]

Якщо ж розглядати поняття системи управління базою даних (СУБД), то воно вже стосується конкретизованого програмного продукту, що має реалізувати функціональні можливості системи управління для інформаційної бази за допомогою надання певних інструментів для створення, редагування і адміністрування баз даних. Саме цей програмний комплекс має виконувати керування командами пов'язаними з маніпулювання даними (наприклад SQL або інші) та водночас керувати пріоритетністю доступу різних об'єктів до даних, забезпечувати відновлення після збоїв, підтримувати стандартизовані механізми безпеки та резервного копіювання інформації. При цьому такий продукт має забезпечувати достатньо високу продуктивність для системи та можливість масштабування задля можливостей роботи з великими об'ємами даних.

Також важливо зазначити, що СУБД має значно спрощувати роботу для розробників і адміністраторів проекту, спрощуючи і приховуючи фізичне зберігання даних за зрозумілими поняттями, за допомогою яких користувачі мають змогу керувати сутностями, зв'язками та атрибутами замість управління такими складними процесами, як читанням і записом файлів до блоків пам'яті. [40]

Саме тому для проєкту було прийнято рішення використовувати систему управління базами даних (СУБД) – PostgreSQL.

PostgreSQL є безкоштовним об'єктно-реляційним сервером для керування базами даних, що поєднує в собі стандартні SQL системи та гнучкість додаткових розширень і підтримку неструктурованих даних. За історичними даними, цей проект започатковано в Університеті Каліфорнії в Берклі ще в 1986 році, що вказує на те, що проект багато років вдосконалюється та розвивається, і при цьому залишається актуальним. Саме ця СУБД має повну підтримку транзакцій ACID, використовує контроль конкурентності доступу до даних, що й власне дозволяє забезпечити цілісність даних без певних блокувань для читання. [41]

Окрім стандартних реляційних типів, які ще можна назвати класичними, дана система підтримує роботу з json, xml, різними масивами, ба навіть геопросторовими форматами (через деякі розширення) та іншими комплексними структурами. Це все дозволяє маніпулювати як реляційними, так і файловими нагрузками в межах однієї і тої самої СУБД. Для зберігання налаштувань та внутрішніх метаданих PostgreSQL використовує таблиці системного каталогу, і забезпечує таким чином можливість додавання користувацьких типів даних, різних доменів та операторів. Також вона дозволяє створювати як асинхронні, так і синхронні копії бази даних, що в свою чергу створює певні гарантії збереження даних та можливості створення резервних копій на рівні конкретних транзакцій або сесій. [41]

Також дана СУБД підтримує підключення зовнішніх джерел за допомогою Foreign Data Wrappers, що робить її зручним містком між файловими системами, різними базами даних та REST-сервісами, і при цьому надає великий набір інтерфейсів за допомогою яких забезпечує доступ до неї із більшості сучасних мов програмування. [41]

Отже загалом, використання саме PostgreSQL є надійним та масштабованим, позитивний досвід використання якого підтверджується, в тому числі, застосуванням його у великих корпораціях і організаціях.

3.2 Розробка сховища інформаційної бази

Для проекту створено реляційну базу даних, що зображена на рис 3.1, та яка містить 4 пов'язані між собою таблиці, кожна з яких відповідає за конкретне зберігання окремого типу даних про якість повітря. SQL-код створення таблиць міститься у додатку Ж.

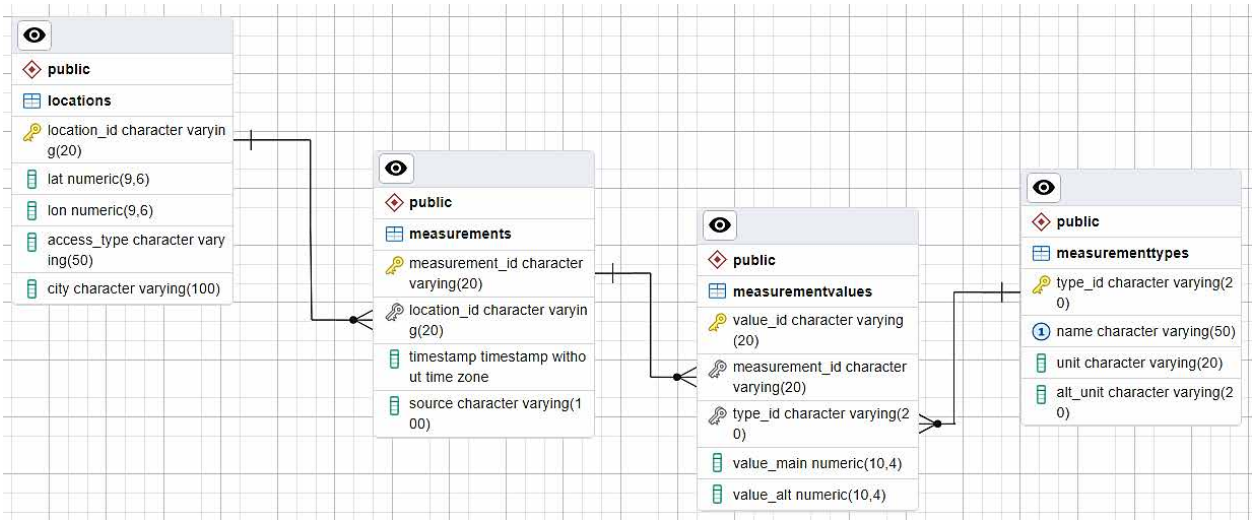

Рис. 3.1 Схема бази даних створеної в PostgreSQL

Таблиця Locations має в собі географічні координати та метадані щодо місця де було здійснено заміри. Кожен запис ідентифікується коротким ключем довжиною до 20 символів, супроводжуючись такими полями як широта, довгота та поле для позначення способу отриманих даних, а також поле для міста, в якому і розташоване джерело даних.

Для стандартизації даних що стосуються типів забруднювачів, було створено таблицю MeasurementTypes, кожен запис якої визначається ідентифікатором рядкового типу розмірністю, знову ж таки, до 20 символі. Ця таблиця має поле ім'я, яке містить назву забруднювача, та поля для зберігання основної і альтернативної назви одиниці вимірювання.

Значення вимірювань здійснених під час збору даних у певний момент, в певному місці зберігаються в таблиці Measurements, де для кожного запису вимірювання є ідентифікатор у вигляді первинного ключа, та є поля які вказують на зовнішній ключ таблиці Location, що реалізований через правило ON DELETE

CASCADE, яке гарантує видалення всіх взаємопов'язаних вимірювань разом із видаленням записів із таблиці локації. Час здійснення вимірювання зберігається у форматі timestamp, а в полі source зберігається саме джерело отриманих даних.

Таблиця MeasurementValues містить в собі деталі конкретних значень концентрації забруднювачів, та має зв'язок з конкретною подією вимірювання, її типом за допомогою зовнішнього ключа і, звісно ж, з дотриманням правила ON DELETE CASCADE для підтримки загальної цілісності даних. Фактичне значення концентрації у таблиці зберігається за допомогою десяткового числа з точністю у чотири знаки після коми, як для основного значення так і альтернативної одиниці вимірювання.

3.3 Визначення інструментів для реалізації програмного забезпечення проєкту

Як основною мовою програмування було обрано Python, через певні ключові причини, які пов'язані з технічними перевагами цієї мови та відповідністю її до вимог системи. Python - це мова програмування, що є високорівневою та інтерпретованою і вважається універсальною, оскільки вирізняється доволі простим та зрозумілим синтаксисом, великою кількістю існуючих бібліотек, що водночас робить її дуже зручною для розробки програмних продуктів різної складності. [42]

Чи не найважливішою причиною для вибору саме Python, як мови програмування, є її можливості пов'язані з обробкою даних, оскільки система повинна працювати з великою кількістю інформації про показники. Обрана мова програмування легко інтегрується з іншими компонентами, наприклад такими як веб-фреймворк Flask, що дає змогу створювати зручні REST API як backend для обміну даними з інтерфейсом програми або іншими зовнішніми сервісами. Також можна зауважити, що Python використовується дуже широко для розробки систем візуалізації та науки про дані, і через це має велику спільноту

професійних розробників, котрі публікують різні ідеї та технічні рішення у відкритому доступі, що може позитивно застосовуватись для вирішення типових завдань та масштабування проекту використовуючи, так звані, «best practice». Також, до всього вище сказаного, Python добре підтримується великою кількістю різних хмарних платформ, що забезпечує можливості для подальшого розвитку програмного продукту. [42, 43]

Фреймворк Flask використовується для реалізації серверної частини проекту, та його було обрано через його гнучкість, легкість та відповідність до вимог середніх за складністю веб-систем. Сам Flask є таким собі мікрофреймворком на мові Python, який забезпечує базовий необхідний функціонал веб-серверу, при цьому залишаючи місце для вибору різних архітектурних рішень та інших бібліотек для впровадження їх в проект. Саме ця особливість є важливою для створення рішення що не обтяжене зайвими непотрібними додатковими конфігураціями а містить у собі лише необхідний набір функціоналу. [43]

Flask легко справляється з завданнями обробки запитів до бази даних, формуванням відповідей формату JSON, підтримкою REST API, застосуванням для роботи з інтерфейсом AJAX-запитів, простотою обробки HTTP запитів. [43]

Для даного проекту саме Flask є доволі зручним, оскільки має повний необхідний набір інструментарію, що покриває вирішення всіх необхідних задач.

Для розробки інтерфейсу користувача було обрано досить стандартну комбінацію технологій, а саме: HTML, CSS та JavaScript. Як відомо, HTML є мовою розмітки, що дозволяє відображати текстові дані і виступати основою для подальшої стилізації та взаємодії, а саме стилізації за допомогою CSS. Саме CSS допомагає в оформленні сторінки, дозволяючи створювати адаптивний дизайн інтерфейсу. При цьому, JavaScript використовується для забезпечення інтерактивності і оновлення інформації у динамічному режимі на сторінках без необхідності завантаження всього контенту статично, тобто дозволяє завантажувати лише частину контенту сторінки замість усієї сторінки. Через використання JavaScript також реалізовуються асинхронні запити до сервера,

обробка отриманих JSON даних і сама динамічна побудова сторінки, а також обробка дій користувача. [44]

Оскільки в проекті необхідно відображати інтерактивну карту то було прийнято рішення про використання такої бібліотеки як Leaflet. Дана бібліотека є сучасним інструментом відкритого типу для JavaScript що дозволяє інтегрувати просту карту з певними оптимізаціями що позитивно впливає на навантаження на браузер. Ця бібліотека підтримує використання шарів на карті різні формати заголовків маркерів та геооб'єктів що дозволяє їй візуалізувати геопросторові дані у форматі реального часу, що вкрай важливо для даного проекту. [44]

3.4 Процес створення програмних компонентів

Перед початком написання коду для окремих модулів необхідно створити сам проєкт, організувати його структуру та файли і завантажити всі необхідні залежності.

Для створення проєкту необхідно використовувати певне ізольоване середовище, що створює можливості для окремої інсталяції інтерпретатора Python та різних наборів бібліотек, які необхідні для роботи в проєкті, при цьому не впливаючи на налаштування глобального інтерпретатора і таким чином інших систем або проєктів. Коли створюється кожне окреме віртуальне оточення, то в папку з ним копіюється сам Python з набором сховища пакетів, що і забезпечує ізоляцію системи.

Для проєкту впроваджено віртуальне середовище, яке має назву fiialo.

То ж, віртуальне середовище вже існує, і тепер в нього встановлюються необхідні для роботи проєкту пакети, насамперед вищезгаданий Flask і спеціальний драйвер для взаємодії з PostgreeSQL. В даному випадку використовується доволі поширений адаптер psycopg2. Вже після встановлення всіх необхідних бібліотек визначаються параметри підключення до нашої БД, які зазначаються у файлі конфігурації. Варто зауважити, що база даних має бути попередньо створена в середовищі PostgreeSQL, і до неї має бути правильно

налаштований доступ. Те, як організовано безпосереднє підключення до бази даних можна переглянути на рис. 3.2.

```
def initialize(cls):
    try:
        cls.__connection_pool = psycopg2.pool.SimpleConnectionPool(
            minconn=1,
            maxconn=10,
            host=os.getenv('DB_HOST'),
            database=os.getenv('DB_NAME'),
            user=os.getenv('DB_USER'),
            password=os.getenv('DB_PASSWORD'),
            port=os.getenv('DB_PORT', '5432')
        )
        current_app.logger.info("Пул з'єднань до PostgreSQL ініціалізовано")
    except OperationalError as e:
        current_app.logger.error(f"Помилка підключення до БД: {e}")
        raise
```

Рис. 3.2 Функція підключення до PostgreSQL

У ролі контролера (Controller у MVC) виступає саме ядро додатку, тобто файл app.py, який ініціалізує Flask додаток, визначає маршрути та організовує зв'язки між шаблонами, бізнес логікою, даними.

За відображення даних, тобто View у MVC, відповідає модуль, що містить папку з шаблонами – тобто HTML сторінками з певними вбудованими всередину макросами за системою шаблонів Jinja2.

Папка static/ містить в собі файли CSS, JavaScript та всі зображення, використані в проєкті.

Далі варто згадати що використовується неявна модель, яка має вигляд функції для роботи з даними, тобто відповідає за частину моделі (Model у MVC).

Особливо важливою є функція для розрахунку AQI, за стандартами різних країн. Дана функція приймає такі значення як концентрації забруднюючих речовин та стандарт розрахунку. Надалі використовується певна формула для розрахунку суб-індексів для кожної речовини окремо.

Формула у функції для розрахунку суб-індексу працює наступним чином: обраховується різниця між нижньою межою AQI та верхньою межою AQI, яка ділиться на різницю верхньої межі діапазону та нижньої межі діапазону, і результат цього ділення множиться на різницю нашої концентрації забруднювача та нижньої межі діапазону. В кінці до цього всього додається нижня межа AQI. Результат заокруглюється до цілого і повертається. Функція

розраховує індекс для кожного доступного забруднювача окремо, а потім відповідно до стандартів різних країн: або ж обирається і повертається найгірший з показників з його категорією, або вираховується і повертається середнє значення для усіх показників. Код для функції розрахунку суб-індексів можна переглянути в додатку К.

Отже можна зробити висновок, що відбувається формування структури застосунку, де одну з найважливіших ролей займає функція розрахунку AQI. Вона виконується на сервері, та результати її обрахунків відправляються у контексті HTML сторінки для відображення користувачу. Структуру кінцевого каталогу проєкту відображено на рис. 3.3.

Рис. 3.3 Структура каталогу проєкту

Сам шаблон HTML будується за допомогою вбудованого у Flask механізму Jinja2, в якому визначається розмітка сторінки, підключаються різні CSS стилі і звісно ж, виділяється окремий блок у вигляді контейнеру з заданими певними розмірами в який і буде вбудовано карту. Ну і на останок у секції <head> відбувається підключення бібліотеки Leaflet.

Далі в результаті завантаження сторінки браузер одержує від Flask готовий

HTML файл з вбудованим JSON масивом даних, щодо координат станцій вимірювання і кольору, відповідно до категорії яким потрібно позначити точки, а також відповідні AQI значення. JavaScript виконує ініціалізацію карти Leaflet, задає за початкову точку як центр міста Київ та відповідний масштаб, після чого перебирає перелік координат з показниками формуючи для кожного кольоровий відповідний маркер.

4 МЕТОДИЧНІ ВКАЗІВКИ ДЛЯ ВПРОВАДЖЕННЯ ТА ТЕХНІЧНОЇ ЕКСПЛУАТАЦІЇ ПРОГРАМНОГО ПРОЄКТУ

4.1 Тестування функціоналу програмного забезпечення

Тестування ПЗ є процесом напрямленим на перевірку та оцінювання якості програмного продукту, що має на меті виявити помилки, баги і невідповідності між поведінкою та вимогами. Допомагає забезпечити ефективність безпеку та надійність роботи програмного забезпечення, через що стає важливою частиною життєвого циклу продукту. [45]

Метою тестування є саме виявлення дефектів ще до того, як буде впроваджено програмне забезпечення. Це позитивно впливає на проект зменшуючи ризики, які напряму пов'язані з помилками роботи системи та водночас підвищує якість продукту і дає змогу уникнути значних фінансових втрат після запуску. [45]

Функціональне тестування є одним із видів тестування, яке направлене на перевірку відповідності вимогам всіх функцій системи відповідно до їх специфікації. Функціональне тестування включає в себе перевірку роботи окремих функцій, запитів, надійності взаємодії з базою даних, коректність відображення інформації в інтерфейсі користувача, та виконується імітуючи можливі реальні сценарії використання програмної системи. [45]

Для системи було обрано саме функціональне тестування та створено декілька сценаріїв, які охоплюють важливу функціональність системи.

1. Тестувальний сценарій - Відкриття головної сторінки з картою

Метою цього тестування є перевірка успішного завантаження сторінки з відображенням інтерактивної карти та відображеням на ній даних, отриманих від API.

Вхідними даними для тестування є запит до основної URL адреси.

Очікуваним результатом є відкривання сторінки без виникнення помилок, відображення на сторінці базової карти Києва та відсутні в консолі помилки, а також відображення на карті точок отриманих від API (тобто база даних немає записів про вимірювання, станції та точні показники).

Результат тестування засвідчив успішне завантаження сторінки при переході за прямим посиланням, відображення на ній навігаційної стрічки карти з позначеними точками отриманими з БД, в яку їх було заздалегідь підв'язано через API та показав відсутність повідомлень про помилки у консолі браузера. Наочно можна переглянути результат тестування на рис. 4.1.

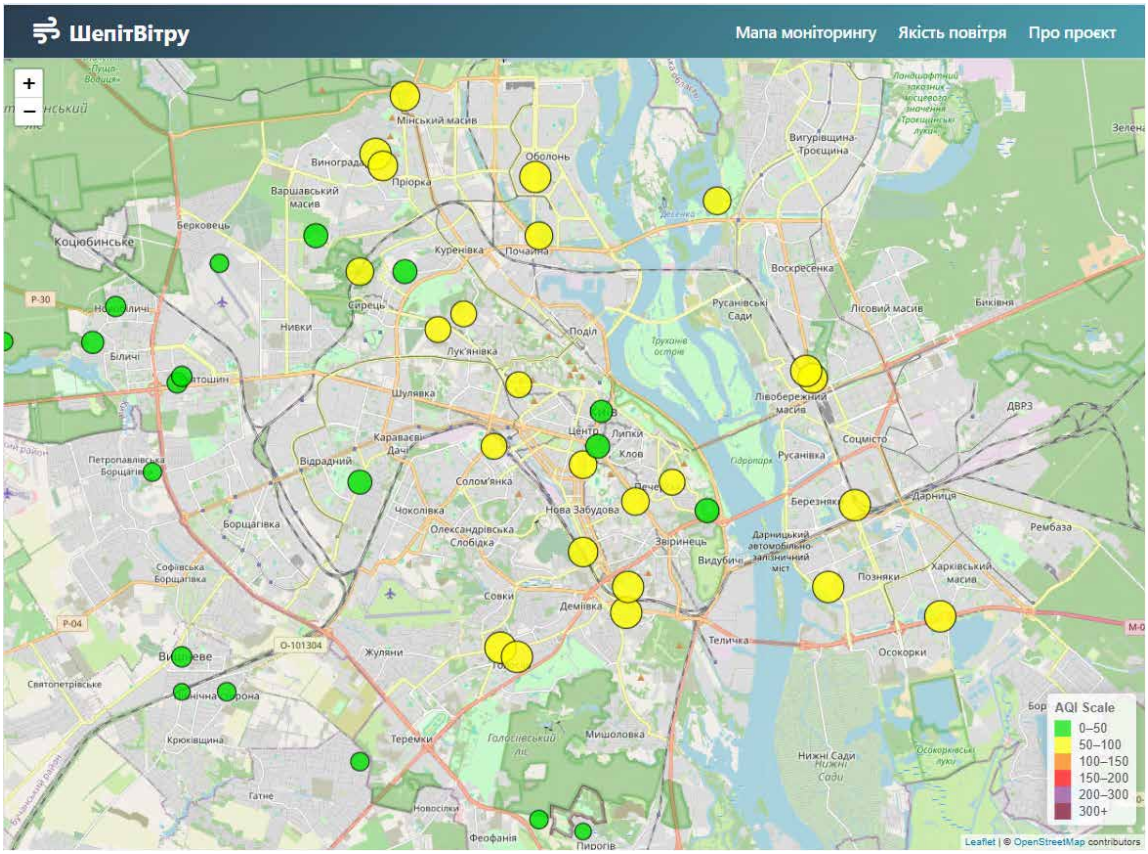

Рис. 4.1 Головна сторінка проєкту з відображенням карти з даними

2. Тестувальний сценарій - Завантаження та відображення маркерів якості повітря на карті

Метою цього тестування є перевірка правильності відтворення маркерів на карті отриманих з бази даних, а також перевірка зв'язку з БД.

Вхідними даними є записи занесені в таблиці бази даних, що відповідають

трьом маркерам з розташуванням на Майдані Незалежності (центр), на Оболонській набережній та на Святошині, з відповідними кольорами і розмірами точок – зеленим, жовтим та червоним.

Очікуваним результатом є поява трьох кругових маркерів на карті, розташованих згідно з заданими координатами, при цьому колір та розмір маркеру має відповідати значенню AQI.

Проведене тестування показало коректність відображення точок на карті відповідно до значення AQI, а також довело що присутнє працююче з'єднання з базою даних та механізм отримання даних відповідає очікуванням. Також в консолі браузера відсутні помилки щодо роботи JavaScript. Результат тестування можна переглянути на рис. 4.2, де добре видно виведення трьох точок в зазначених локаціях та їх кольорова ідентифікація відповідно до рівня забруднення.

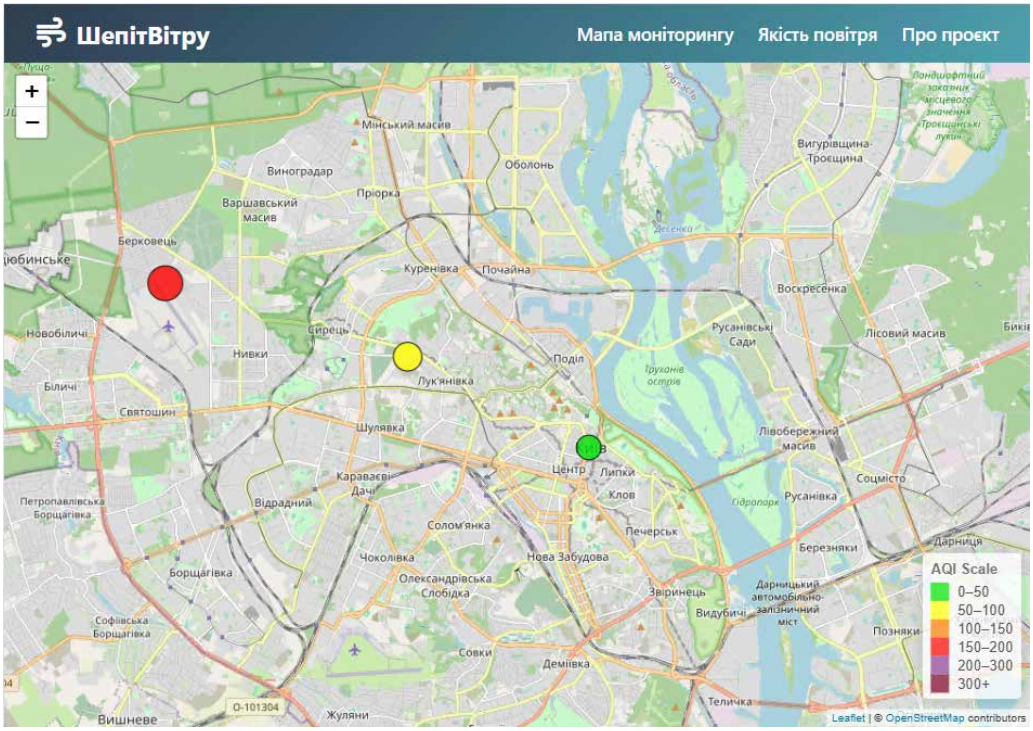

Рис. 4.2 Тестування виведення маркерів з даних БД

3. Тестувальний сценарій - Розкриття вікна з інформацією про AQI за різними стандартами та деталі показників для будь-якої точки на карті.

Метою цього тестування є перевірка отримання та відображення деталей вимірювання по конкретним станціям та отримання списку розрахованих

значень AQI відповідно до стандартів різних країн.

Вхідними даними є точки на карті по яких відбувається подія клік.

Очікуваним результатом є поява після кліку певного вікна з відображенням даних про різні AQI та деталей забруднювачів у вигляді списку.

В результаті тестування при натисканні на одну із точок що знаходиться в Шевченківському районі міста Києва було виведено свіжі дані про показники та розраховані за різними стандартами індекси якості повітря та відповідно їм текстове повідомлення. Результат відображення можна переглянути на рис. 4.3.

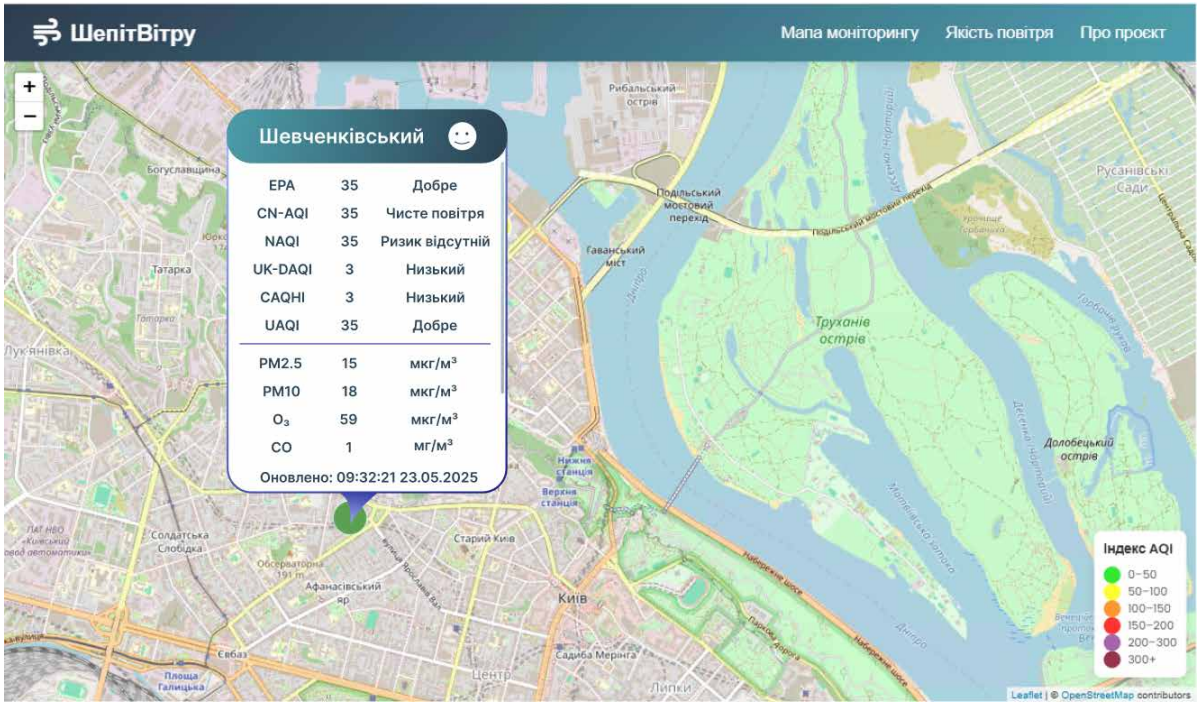

Рис. 4.3 Відображення вікна з різними AQI стандартами та деталями показників

4.2 Апаратні та програмні вимоги системи

Для визначення вимог було побудовано діаграму розміщення, зображену на рис. 4.4 , яка наочно демонструє вузли системи та розміщення компонентів у кожному з них.

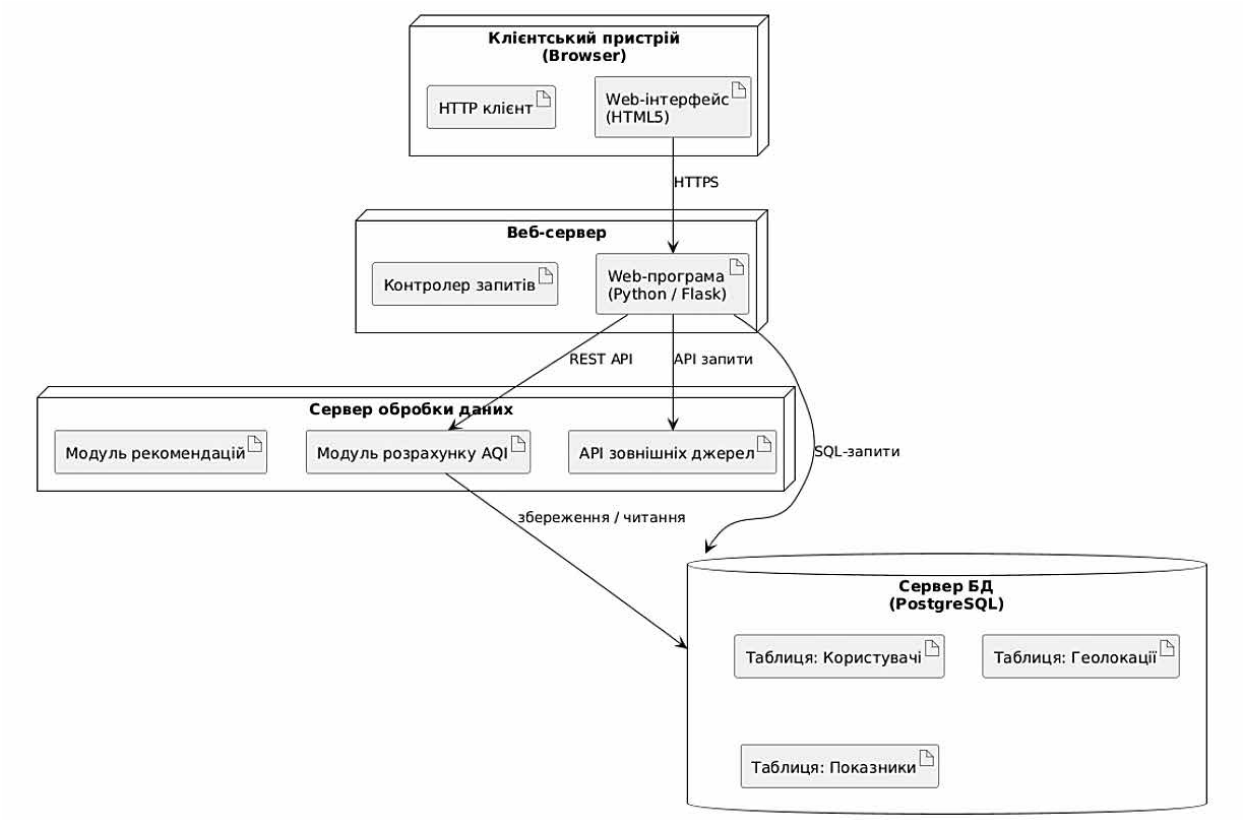

Рис. 4.4 Діаграма розміщення для системи

Для роботи програмної системи необхідно наявність одного комп'ютера, що забезпечить виконання серверної частини додатку, а саме обробку запитів користувача, зберігання даних у базі, надсилання результатів для відображення через веб-інтерфейс.

Для мінімальних апаратних вимог розміщення сервісу:

- Процесор сімейства Intel Core i3 або ж AMD Ryzen 3, з 2-ма або більше ядрами; оперативна пам’ять від 4 Гб;
- місце на диску для зберігання файлів проєкту має становити від 5 Гб;
- стабільне інтернет-з’єднання для підвантаження залежностей.

Для мінімальних апаратних вимог клієнтської частини:

- Процесор сімейства Intel Pentium або ж AMD Athlon, з 2-ма або більше ядрами;
- оперативна пам’ять від 4 Гб;
- стабільне інтернет-з’єднання.

Що стосується мінімальних вимог до програмного забезпечення системи, то для користувача це:

- Операційна система не раніше зазначених версій - Windows 10/11, macOS 10.15+, Linux (Ubuntu 20.04+);
- Браузер останніх актуальних версій - Chrome 100+ / Firefox 90+ / Edge

Для сервера вимоги ПЗ є складнішими, оскільки це впливає на роботоспроможність системи:

- Python версії 3.8 або вище,
- pip останньої доступної версії,
- PostgreeSQL версії 13+,
- pgAdmin 4.

4.3 Структура інсталяційного набору

Інсталяційний пакет повинен включати в себе набір з усіх необхідних компонентів системи, які забезпечують повноцінне його функціонування в певному середовищі.

В основі інсталяційного набору для розробленої системи включено серверну частину, написану на мові Python - до складу якої входить директорія з вихідним кодом, що містить файли модулів написаних на Flask, шаблони HTML, ресурси для відображення інтерфейсу (тобто фронтенду), а саме CSS файли, зображення, скрипти JavaScript та файли конфігурації.

Також окремо виділено SQL скрипт та дамп бази даних PostgreSQL, який дозволяє швидко створити всі таблиці та забезпечити їх даними. В пакеті передбачено також файл залежностей «requirements.txt» за допомогою якого, використовуючи менеджер pip можна легко встановити усі супутні бібліотеки проєкту. Вміст цього файлу з залежностями для проєкту можна переглянути на рис. 4.5.

requirements.txt

```
blinker==1.9.0
certifi==2025.4.26
charset-normalizer==3.4.2
click==8.2.0
colorama==0.4.6
Flask==3.1.1
idna==3.10
itsdangerous==2.2.0
Jinja2==3.1.6
MarkupSafe==3.0.2
psycopg2-binary==2.9.10
requests==2.32.3
urllib3==2.4.0
Werkzeug==3.1.3
```

Рис. 4.5 Файл requirements.txt для проєкту

Для зручності та швидкого розгортання файли інсталяційного набору буде поміщено до архіву. Структуру інсталяційного набору подано на рис. 4.6.

```
/AirWisp_project
|
├── app.py                  # Основний Flask-додаток
├── aqi_calculator.py       # Модуль для розрахунків AQI
├── config.py               # Конфігураційні параметри
├── requirements.txt        # Залежності
├── templates/
|   └── index.html          # Шаблон сторінки
└── static/                  # Статичні файли (CSS, JS)
```

Рис. 4.6 Ієрархічна модель структури інсталяційного набору

ВИСНОВКИ

Під час виконання роботи було виконано повноцінне моделювання, проектування та безпосередньо реалізацію програмного забезпечення для інформаційної системи аналізу моніторингу якості повітря. Для цього було використано широкий сучасний стек різних технологій, таких як Python, Flask, PostgreSQL, HTML, CSS, JavaScript та бібліотека Leaflet. Цей проект мав за мету не лише зібрати, обробити та виконати візуалізацію даних про стан повітря міста Києва, а й забезпечити зручний інтерфейс та надати користувачу можливості через нього отримувати оперативну інформацію про забруднення у різних районах столиці. Також вперше було реалізовано розрахунок індексу якості повітря за стандартами різних країн, що дозволить користувачу кількісно та якісно оцінювати небезпеку забруднення для свого здоров'я.

Першим етапом розробки був аналіз предметної області, який дозволив виокремити ключові сутності та дати визначення вимогам до функціоналу системи. Було також спроектовано структуру бази даних в PostgreSQL, що логічно інтерпретує реальні об'єкти моніторингу, типи забруднювачів, значення вимірювань, джерела даних та географічні координати для розміщення джерел. Забезпечено структурованість достовірність і надійність збереження даних за рахунок того, що усі таблиці мають чітко визначені зв'язки та правила обмеження цілісності.

Для написання коду серверної частини було обрано таку мову програмування, як Python, разом із застосуванням фреймворку Flask для того, щоб швидко реалізувати логіку взаємодії з базою даних, налаштувати маршрутизацію запитів та передачу інформації у зручні HTML шаблони для відображення інтерфейсу. Інтерфейс реалізовано використовуючи HTML та CSS, що забезпечило просту базову структуру для сторінок та їх привабливий зовнішній вигляд. Для того щоб відображати просторові дані інтерактивно, було

використано JavaScript бібліотеку Leaflet, котра дозволила створити динамічну мапу та відобразити на ній маркери забруднення з різними кольорами залежно від рівня AQI та певних категорій забруднення.

Багато уваги приділялося чистоті архітектури, модульності коду та розмежуванню відповідальностей між різними компонентами системи. Було також здійснено перевірку усіх функціональних можливостей проекту, зокрема виконувалися тестування функціональними сценаріями для перевірки відображення даних взаємодії з базою даних, а також перевірки модуля розрахунку AQI за різними стандартами. Було також сформовано вимоги апаратного та програмного забезпечення для системи, виокремлюючи клієнтську та серверну частину, а також описано склад інсталяційного набору для розгортання системи в інших середовищах або на інших машинах.

Отже, у результаті виконання роботи було створено повністю робочу функціональну інформаційну систему та виконано всі функціональні та нефункціональні вимоги до її створення.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ambient (outdoor) air pollution - World Health Organization [Електронний ресурс]. – Режим доступу: https://www.who.int/news-room/fact-sheets/detail/ambient-(outdoor)-air-quality-and-health (дата звернення: 11.04.2025).

2. Ecosystems and Air Quality – EPA [Електронний ресурс]. – Режим доступу: https://www.epa.gov/eco-research/ecosystems-and-air-quality (дата звернення: 11.03.2025).

3. Risk Assessment for the Population of Kyiv, Ukraine as a Result of Atmospheric Air Pollution – EHP [Електронний ресурс]. – Режим доступу: https://ehp.niehs.nih.gov/doi/full/10.5696/2156-9614-10.25.200303 (дата звернення: 11.04.2025).

4. The Importance of Air Quality Assessment – IAQ.WORKS [Електронний ресурс]. – Режим доступу: https://iaq.works/source-control/the-importance-of-air-quality-assessment-2/ (дата звернення: 12.04.2025)

5. Need for Air Quality Monitors System In Smart Cities – OIZOM [Електронний ресурс]. – Режим доступу: https://oizom.com/need-for-air-quality-monitors-system-in-smart-cities/ (дата звернення: 12.04.2025)

6. Северин Л. І., Петрук В. Г., Безвозюк І. І., Васильківський І. В. *Природоохоронні технології. Частина перша. Захист атмосфери* [Електронний ресурс] / Вінницький національний технічний університет. – Режим доступу: https://web.posibnyky.vntu.edu.ua/iebmd/severin_priodoohoronni_tehnologii/11-7.html – 11.7 Екологічне нормування якості атмосферного повітря (дата звернення: 14.04.2025)

7. Що таке індекс якості повітря? – НУБіП [Електронний ресурс]. – Режим доступу: https://nubip.edu.ua/node/80156 (дата звернення: 16.04.2025)

8. What is Air Quality? – NASA [Електронний ресурс]. – Режим доступу: https://www.nasa.gov/general/what-is-air-quality/#hds-sidebar-nav-3 (дата звернення: 16.04.2025)

9. Technical Assistance Document for the Reporting of Daily Air Quality – the Air Quality Index (AQI) – USA Enviromental Protection Agency [Електронний

ресурс]. – Режим доступу: https://document.airnow.gov/technical-assistance-document-for-the-reporting-of-daily-air-quailty.pdf (дата звернення: 17.04.2025)

10. About National Air Quality Index – CPCB [Електронний ресурс]. – Режим доступу: https://cpcb.nic.in/displaypdf.php?id=bmF0aW9uYWwtYWlyLXF1YWxpdHktaW5kZXgvQWJvdXRfQVFJLnBkZg== (дата звернення: 17.04.2025)

11. Air quality in China – IQAir [Електронний ресурс]. – Режим доступу: https://www.iqair.com/china (дата звернення: 18.04.2025)

12. About the Air Quality Health Index – Environment and Climate Change Canada (ECCC) [Електронний ресурс]. – Режим доступу: https://www.canada.ca/en/environment-climate-change/services/air-quality-health-index/about.html (дата звернення: 18.04.2025)

13. What is the Daily Air Quality Index? – Department for Environment Food & Rural Affairs [Електронний ресурс]. – Режим доступу: https://uk-air.defra.gov.uk/air-pollution/daqi?view=more-info (дата звернення: 18.04.2025)

14. Контроль якості повітря: як працює система моніторингу в Києві та Київській області – ДЕІ [Електронний ресурс]. – Режим доступу: https://stolreg.dei.gov.ua/post/1057 (дата звернення: 18.04.2025)

15. Платформи. Перелік сайтів, проектів, органів місцевого самоврядування та інших джерел, з яких збираються дані – SaveEcoBot [Електронний ресурс]. – Режим доступу: https://www.saveecobot.com/platforms (дата звернення: 18.04.2025)

16. Про затвердження Порядку інформування населення про якість повітря за основними показниками з використанням індексу якості повітря в Україні: Наказ – Міністерство захисту довкілля та природних ресурсів України [Електронний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/z0536-25#Text (дата звернення: 18.04.2025)

17. Environmental and Health Impacts of Air Pollution: A Review – Frontier [Електронний ресурс]. – Режим доступу: https://www.frontiersin.org/journals/public-

health/articles/10.3389/fpubh.2020.00014/full (дата звернення: 18.04.2025)

18. Air quality, energy and health. Health impacts – World Health Organization [Електронний ресурс]. – Режим доступу: https://www.who.int/teams/environment-climate-change-and-health/air-quality-energy-and-health/health-impacts (дата звернення: 18.04.2025)

19. Air Pollution and Your Health – USA NIEHS [Електронний ресурс]. – Режим доступу: https://www.niehs.nih.gov/health/topics/agents/air-pollution (дата звернення: 18.04.2025)

20. Towards Integrated Air Pollution Monitoring and Health Impact Assessment Using Federated Learning: A Systematic Review – PubMed Central [Електронний ресурс]. – Режим доступу: https://pmc.ncbi.nlm.nih.gov/articles/PMC9160600/ (дата звернення: 18.04.2025)

21. Low emission zones are successful in cutting air pollution, study finds – The Guardian [Електронний ресурс]. – Режим доступу: https://www.theguardian.com/environment/2025/may/16/low-emission-zones-successful-cutting-air-pollution-researchers-find (дата звернення: 18.04.2025)

22. WHO global air quality guidelines. Particulate matter (PM2.5 and PM10), ozone, nitrogen dioxide, sulfur dioxide and carbon monoxide – World Health Organization [Електронний ресурс]. – Режим доступу: https://iris.who.int/bitstream/handle/10665/345329/9789240034228-eng.pdf (дата звернення: 18.04.2025)

23. Indoor air quality guidelines from across the world: An appraisal considering energy saving, health, productivity, and comfort – Environment International / Volume 178, August 2023, 108127 [Електронний ресурс]. – Режим доступу: https://www.sciencedirect.com/science/article/pii/S0160412023004002 (дата звернення: 18.04.2025)

24. Domain Modeling - Software Engineering [Електронний ресурс]. – Режим доступу: https://www.geeksforgeeks.org/software-engineering-domain-modeling/ (дата звернення: 20.04.2025)

25. Use-case diagrams - Rational Software Architect / version 9.6.1 – IBM

[Електронний ресурс]. – Режим доступу: https://www.ibm.com/docs/en/rational-soft-arch/9.6.1?topic=diagrams-use-case (дата звернення: 22.04.2025)

26. What is Sequence Diagram? – Visual Paradigm [Електронний ресурс]. – Режим доступу: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-sequence-diagram/ (дата звернення: 22.04.2025)

27. Air quality monitoring map – SaveEcoBot [Електронний ресурс]. – Режим доступу: https://www.saveecobot.com/maps (дата звернення: 22.04.2025)

28. Air quality monitoring solution – Airly [Електронний ресурс]. – Режим доступу: https://airly.org/en/about-airly/ (дата звернення: 22.04.2025)

29. Моніторинг якості повітря – Eco City [Електронний ресурс]. – Режим доступу: https://eco-city.org.ua/ (дата звернення: 22.04.2025)

30. Система Перевірки Моделювання Моніторингу якості повітря – YourAirTest [Електронний ресурс]. – Режим доступу: https://youairtest.com/uk/ (дата звернення: 23.04.2025)

31. Air quality in Ukraine – IQAir [Електронний ресурс]. – Режим доступу: https://www.iqair.com/us/ukraine (дата звернення: 23.04.2025)

32. Air Report & Flow – Plume Labs [Електронний ресурс]. – Режим доступу: https://plumelabs.com/en/ (дата звернення: 23.04.2025)

33. Air Quality API v2 – BreezoMeter [Електронний ресурс]. – Режим доступу:https://docs.breezometer.com/api-documentation/air-quality-api/v2/#current-conditions (дата звернення: 23.04.2025)

34. Fighting air inequality through open data – OpenAQ [Електронний ресурс]. – Режим доступу: https://openaq.org/ (дата звернення: 23.04.2025)

35. What is a logical data model? – Tibco [Електронний ресурс]. – Режим доступу: https://www.tibco.com/glossary/what-is-a-logical-data-model (дата звернення: 23.04.2025)

36. What is an Entity Relationship Diagram (ERD)? – Lucidchart [Електронний ресурс]. – Режим доступу: https://www.lucidchart.com/pages/er-diagrams (дата звернення: 23.04.2025)

37. erwin Data Modeler – ERwin [Електронний ресурс]. – Режим доступу:

https://www.erwin.com/products/erwin-data-modeler/ (VPN: Франція, дата звернення: 23.04.2025)

38. What is Class Diagram? – Visual Paradigm [Електронний ресурс]. – Режим доступу: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/ (дата звернення: 23.04.2025)

39. What is Component Diagram? - Visual Paradigm [Електронний ресурс]. – Режим доступу: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-component-diagram/ (дата звернення: 23.04.2025)

40. Database management system (DBMS) – TechTarget [Електронний ресурс]. – Режим доступу: https://www.techtarget.com/searchdatamanagement/definition/database-management-system (дата звернення: 23.04.2025)

41. What is PostgreSQL? – Amazon [Електронний ресурс]. – Режим доступу: https://aws.amazon.com/rds/postgresql/what-is-postgresql/ (дата звернення: 23.04.2025)

42. What is Python? Executive Summary – Python [Електронний ресурс]. – Режим доступу: https://www.python.org/doc/essays/blurb/ (дата звернення: 23.04.2025)

43. Flask’s documentation – Flask [Електронний ресурс]. – Режим доступу: https://flask.palletsprojects.com/en/stable/ (дата звернення: 23.04.2025)

44. Learn Web Development Basics – HTML, CSS, and JavaScript Explained for Beginners – FreeCodeCamp [Електронний ресурс]. – Режим доступу: https://www.freecodecamp.org/news/html-css-and-javascript-explained-for-beginners/ (дата звернення: 23.04.2025)

45. Огляд видів тестування – QATestLab [Електронний ресурс]. – Режим доступу: https://training.qatestlab.com/blog/technical-articles/review-the-types-of-testing/ (дата звернення: 23.04.2025)

ДОДАТКИ

ДОДАТОК А

Таблиця граничних значень для розрахунку AQI США (стандарт EPA)

Категорія AQI	**AQI**	**PM2.5 (µg/m³)**	**PM10 (µg/m³)**	**O_3 (ppb)**	**NO_2 (ppb)**	**CO (ppm)**	**SO_2 (ppb)**	**Загальне повідомлення**
Добра *(Good)*	0–50	0.0 – 12.0	0 – 54	0 – 54	0 – 53	0.0 – 4.4	0 – 35	Якість повітря вважається задовільною, забруднення повітря становить мінімальний ризик.
Помірна *(Moderate)*	51–100	12.1 – 35.4	55 – 154	55 – 70	54 – 100	4.5 – 9.4	36 – 75	Прийнятна якість повітря, однак для деяких людей із чутливістю це може бути ризиковано.
Нездорова для чутливих груп *(Unhealthy for Sensitive Groups)*	101–150	35.5 – 55.4	155 – 254	71 – 85	101 – 360	9.5 – 12.4	76 – 185	Може спричинити проблеми у людей з хворобами легенів, літніх та дітей.
Нездорова *(Unhealthy)*	151–200	55.5 – 150.4	255 – 354	86 – 105	361 – 649	12.5 – 15.4	186 – 304	Кожен може почати відчувати проблеми зі здоров’ям.
Дуже нездорова *(Very Unhealthy)*	201–300	150.5 – 250.4	355 – 424	106 – 200	650 – 1249	15.5 – 30.4	305 – 604	Попередження про здоров’я: надзвичайно шкідливо для всіх.
Небезпечна *(Hazardous)*	301–500	250.5 – 500.4	425 – 604	> 200	1250 – 2049	30.5 – 50.4	605 – 1004	Надзвичайні умови: кожен може зазнати серйозного впливу на здоров’я.

ДОДАТОК Б

Таблиця граничних значень для розрахунку AQI Китаю

Категорія AQI	**AQI**	**PM2.5 (мкг/м³)**	**PM10 (мкг/м³)**	**O_3 (8-год) (мкг/м³)**	**CO (мг/м³)**	**SO_2 (мкг/м³)**	**NO_2 (мкг/м³)**	**Опис рівня**	**Повідомлення для населення**
Добрий	0–50	0–35	0–50	0–100	0–2	0–50	0–40	Чисте повітря	Без обмежень для населення
Помірний	51–100	36–75	51–150	101–160	2–4	51–150	41–80	Прийнятна якість	Малий ризик для чутливих людей
Нездоровий для чутливих груп	101–150	76–115	151–250	161–200	4–14	151–475	81–180	Може впливати на чутливих людей	Людям з хворобами серця/легенів уникати тривалого перебування на вулиці
Нездоровий	151–200	116–150	251–350	201–300	14–24	476–800	181–280	Шкідливо для всіх	Обмежити активність на відкритому повітрі
Дуже нездоровий	201–300	151–250	351–420	301–400	24–36	801–1600	281–565	Серйозна шкода	Уникати перебування надворі, особливо дітям і літнім людям
Небезпечний	301–500	251–500	421–600	401–800	36–60	1601–2100	566–940	Надзвичайно небезпечно	Надзвичайна ситуація — всі мають залишатися в приміщенні

ДОДАТОК В

Таблиця граничних значень для розрахунку AQI Індії

Категорія AQI	**AQI**	**PM2.5 (мкг/м³)**	**PM10 (мкг/м³)**	**NO_2 (мкг/м³)**	**O_3 (8-год) (мкг/м³)**	**CO (мг/м³)**	**SO_2 (мкг/м³)**	**Опис**	**Повідомлення для населення**
Добра (Good)	0–50	0–30	0–50	0–40	0–50	0–1.0	0–40	Якість повітря задовільна	Немає ризику
Задовільна (Satisfactory)	51–100	31–60	51–100	41–80	51–100	1.1–2.0	41–80	Допустима для більшості	Незначний ризик для чутливих
Помірно забруднена (Moderately polluted)	101–200	61–90	101–250	81–180	101–168	2.1–10	81–380	Дискомфорт у чутливих осіб	Людям із захворюваннями уникати тривалого перебування надворі
Погана (Poor)	201–300	91–120	251–350	181–280	169–208	10–17	381–800	Нездорово	Людям з легеневими проблемами уникати навантажень на відкритому повітрі
Дуже погана (Very Poor)	301–400	121–250	351–430	281–400	209–748	17–34	801–1600	Серйозний вплив на здоров'я	Залишатися в приміщенні, обмежити активність на вулиці
Серйозна (Severe)	401–500	251–500	431–500	401–>400	>748	>34	>1600	Надзвичайно шкідливо	Надзвичайна ситуація — триматися в приміщенні, медична допомога для вразливих

ДОДАТОК Д

Таблиця граничних значень для розрахунку AQHI Канади

Індекс AQHI	Категорія ризику	Рекомендації для загального населення	Рекомендації для чутливих груп
1–3	Низький	Насолоджуйтесь звичними видами активності на свіжому повітрі	Без обмежень
4–6	Помірний	Якщо ви маєте симптоми, розгляньте зменшення або перенесення активності	Розгляньте скорочення або зміну інтенсивних видів діяльності
7–10	Високий	Зменште або перенесіть інтенсивні види активності на відкрите повітря	Уникайте інтенсивної активності на вулиці
10+	Дуже високий	Уникайте фізичної активності на відкритому повітрі	Перебувайте якомога більше в приміщенні, обмежуйте фізичну активність повністю

ДОДАТОК Е

Таблиця граничних значень для розрахунку DAQI у Великій Британії

Індекс (1–10)	**Категорія**	**PM2.5 (мкг/м³)**	**PM10 (мкг/м³)**	**O_3 (мкг/м³)**	**NO_2 (мкг/м³)**	**SO_2 (мкг/м³)**	**Рекомендації для населення**
1–3	Low (Низький)	0–11	0–16	0–59	0–95	0–88	Нормальна активність для всіх
4–6	Moderate (Помірний)	12–23	17–33	60–119	96–199	89–267	Люди з захворюваннями можуть помітити симптоми
7–9	High (Високий)	24–35	34–50	120–179	200–399	268–532	Чутливим групам варто обмежити перебування на вулиці
10	Very High (Дуже високий)	≥36	≥51	≥180	≥400	≥533	Уникати фізичних навантажень надворі, усім з респіраторними проблемами залишатися в приміщенні

ДОДАТОК Ж

SQL-код для створення таблиць бази даних

```
DROP TABLE IF EXISTS MeasurementValues;
DROP TABLE IF EXISTS Measurements;
DROP TABLE IF EXISTS MeasurementTypes;
DROP TABLE IF EXISTS Locations;
-- Таблиця: Locations
CREATE TABLE Locations (
    id VARCHAR(20) PRIMARY KEY,
    lat DECIMAL(9,6) NOT NULL,
    lon DECIMAL(9,6) NOT NULL,
    access_type VARCHAR(50),
    city VARCHAR(100)
);
-- Таблиця: MeasurementTypes
CREATE TABLE MeasurementTypes (
    id VARCHAR(20) PRIMARY KEY,
    name VARCHAR(50) NOT NULL UNIQUE,
    unit VARCHAR(20) NOT NULL,
    alt_unit VARCHAR(20)
);
-- Таблиця: Measurements
CREATE TABLE Measurements (
    id VARCHAR(20) PRIMARY KEY,
    location_id VARCHAR(20) NOT NULL,
    timestamp TIMESTAMP WITHOUT TIME ZONE NOT NULL,
    source VARCHAR(100),
    CONSTRAINT fk_measurements_location FOREIGN KEY (location_id)
        REFERENCES Locations(id) ON DELETE CASCADE
);
```

Продовження SQL-коду для створення таблиць бази даних

```
-- Таблиця: MeasurementValues
CREATE TABLE MeasurementValues (
    id VARCHAR(20) PRIMARY KEY,
    measurement_id VARCHAR(20) NOT NULL,
    type_id VARCHAR(20) NOT NULL,
    value_main DECIMAL(10,4) NOT NULL,
    value_alt DECIMAL(10,4),
    CONSTRAINT fk_value_measurement FOREIGN KEY (measurement_id)
        REFERENCES Measurements(id) ON DELETE CASCADE,
    CONSTRAINT fk_value_type FOREIGN KEY (type_id)
        REFERENCES MeasurementTypes(id) ON DELETE CASCADE
);
```

Код для функції розрахунку суб-індексу

```
def calculate_subindex(pollutant, concentration, country="US"):
    # 1. Знаходимо діапазон концентрації (C_min, C_max) і AQI (I_min, I_max)
    ranges = get_ranges_for_pollutant(pollutant, country)

    for range in ranges:
        if range["min"] <= concentration <= range["max"]:
            C_min, C_max = range["min"], range["max"]
            I_min, I_max = range["aqi_min"], range["aqi_max"]
            break

    # 2. Розраховуємо AQI за формулою
    aqi = ((I_max - I_min) / (C_max - C_min)) * (concentration - C_min) + I_min

    # 3. Повертаємо результат
    return {
        "pollutant": pollutant,
        "concentration": concentration,
        "aqi_subindex": round(aqi),
        "category": get_category(aqi, country)
    }
```