НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
Факультет інформаційних технологій

ПОГОДЖЕНО	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ**
Декан факультету	**Завідувач кафедри комп'ютерних наук**
_____ Ігор БОЛБОТ (підпис)	**_____ Белла ГОЛУБ** (підпис)
«___» _____ 2026 р.	«___» _____ 2026 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Програмне забезпечення оцінки розповсюдження забруднюючої аерозольної хмари в антропогенному середовищі»

Спеціальність «121 Інженерія програмного забезпечення»

Освітньо-професійна програма «Інженерія програмного забезпечення»

Гарант освітньої програми
к.т.н., доцент _____ (підпис) **Ганна ВАЙГАНГ**

Керівник бакалаврської кваліфікаційної роботи
доцент, к.ф.-м.-н. _____ (підпис) **Андрій БРИТАН**

Виконав _____ (підпис) **Олександр ФЕДОСЕНКО**

Київ-2026

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук

к.т.н., доцент _____ Белла ГОЛУБ
(підпис)

«___» _____ 2025 р.

ЗАВДАННЯ
ДО ВИКОНАННЯ БАКАЛАВРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
ЗДОБУВАЧУ
Федосенко Олександру Леонідовичу

Спеціальність «121 Інженерія програмного забезпечення»
Освітньо-професійна програма «Інженерія програмного забезпечення»
Тема бакалаврської кваліфікаційної роботи

«Програмне забезпечення оцінки розповсюдження забруднюючої аерозольної хмари в антропогенному середовищі.»

затверджена наказом від «19» грудня 2025 р. № 3196 «С»

Термін подання завершеної роботи на кафедру «___» _____ 2026 р.

Вихідні дані до бакалаврської кваліфікаційної роботи:
_____

Перелік питань, що підлягають дослідженню:
1. Дослідження предметної області
2. Проєктування інформаційного забезпечення
3. Розробка програмного забезпечення
4. Рекомендації щодо впровадження та експлуатації системи

_____

Перелік графічних документів (за необхідності).
_____

Дата видачі завдання «19» __грудня__ 2026 р.

Керівник бакалаврської кваліфікаційної роботи _____ **Андрій БРИТАН**
(підпис)

Завдання взяв до виконання _____**Олександр ФЕДОСЕНКО**
(підпис)

ЗМІСТ

ВСТУП 4
1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ 7
1.1 Аналіз предметної області 7
1.2 Моделювання предметної області 8
1.3 Огляд існуючих аналогічних систем 16
1.4 Постановка завдання 19
2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ 22
2.1 Логічна модель даних у вигляді ER-діаграми 22
2.2 Вибір системи управління базами даних 26
2.3 Розробка структури інформаційної бази 27
2.4 Схема та фізична структура бази даних 32
3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 34
3.1 Архітектура системи 34
3.2 Моделювання структури та взаємодії об'єктів 35
3.3 Організаційна структура програмного забезпечення 37
3.4 Компонентна структура системи 39
3.5 Вибір інструментарію для створення ПЗ 42
3.6 Алгоритмізація та програмування програмних модулів 43
4 РЕКОМЕНДАЦІЇ ЩОДО ВПРОВАДЖЕННЯ ТА ЕКСПЛУАТАЦІЇ СИСТЕМИ 48
4.1 Тестування системи 48
4.2 Вимоги до апаратного та програмного забезпечення 55
4.3 Склад інсталяційного пакету 56
ВИСНОВКИ 57
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 59
ДОДАТКИ 61

ВСТУП

Актуальність. Проблема забруднення атмосферного повітря в антропогенних середовищах (міські агломерації, промислові комплекси) є однією з найактуальніших глобальних екологічних проблем сучасності. Викиди забруднюючих речовин з промислових джерел, автотранспорту та інших джерел призводять до значних негативних наслідків для здоров'я населення та довкілля.

Прогнозування розповсюдження забруднюючих речовин в атмосфері виконує важливу роль прогнозування того, як забруднюючі речовини розповсюджуються в атмосфері, потрібне в багатьох практичних ситуаціях. Зокрема, без таких розрахунків не обійтись при проєктуванні нових промислових підприємств, коли необхідно оцінити їхній вплив на довкілля. Не менш важливу роль вони відіграють у надзвичайних ситуаціях — для оперативного планування заходів із мінімізації наслідків викидів. Окрім цього, подібні інструменти використовуються екологічними службами для перевірки дотримання нормативів якості повітря, а також науковцями у дослідженнях у галузі екологічної безпеки.

Традиційні методи розрахунку розповсюдження аерозольних забруднень потребують складних розрахунків та дорогого спеціалізованого ПЗ. Розробка доступного веб-орієнтованого рішення робить такі розрахунки більш доступними для освітніх установ, дослідницьких центрів та невеликих компаній.

Мета розробки програмного додатку. Метою роботи є розробка веб-застосування (AeroSpread Pro) для оцінки розповсюдження забруднюючої аерозольної хмари в антропогенному середовищі з використанням математичної моделі Гаусівського шлейфу та сучасних веб-технологій.

Для досягнення поставленої мети було визначено кілька конкретних завдань. По-перше, реалізувати математичну модель Гаусівського шлейфу з урахуванням актуальних метеорологічних параметрів. Паралельно з цим розроблявся веб-інтерфейс, через який користувач може зручно вводити параметри викиду і одразу бачити результати. Щоб розрахунки спирались на

реальні дані, а не вигадані, було підключено OpenWeather API. Результати моделювання виводяться безпосередньо на інтерактивній карті, що робить їх наочними навіть для людей без технічної підготовки. Нарешті, передбачено можливість вивантаження звітів у стандартизовані формати для подальшого використання.

Методи та технології. Математичним підґрунтям системи слугує модель Гаусівського шлейфу (Gaussian Plume Model), яка дає змогу розраховувати концентрації забруднюючих речовин на різних відстанях від джерела викиду. Для коректного визначення параметрів дисперсії залежно від стану атмосфери застосовується класифікація Паскіля–Гіффорда, а стійкість атмосфери за наявними метеорологічними даними визначається за формулами Брідоса.

Що стосується технічної реалізації, клієнтська частина побудована на HTML5, CSS3 та JavaScript ES6+, а для відображення інтерактивної карти використовується бібліотека Leaflet з тайлами OpenStreetMap. Серверна частина написана на Node.js із використанням фреймворку Express.js. Дані зберігаються в PostgreSQL — зокрема завдяки підтримці типу JSONB, який добре підходить для збереження сіток концентрацій. Актуальні метеорологічні дані надходять через OpenWeather API. Для формування звітів підключено бібліотеки експорту в PDF, а версіонування коду ведеться через Git.

Апробація програмного додатку. Перед поданням до захисту AeroSpread Pro пройшло перевірку розрахунку розповсюджень. Система розгорнута на віддаленому сервері і доступна для використання.

В ході апробації перевірялись різні аспекти роботи застосунку. Окремо тестувались компоненти API, окремо — взаємодія між фронтендом і бекендом. Особлива увага приділялась точності математичних розрахунків, оскільки від цього безпосередньо залежить практична цінність системи. Також перевірялась стабільність інтеграції з OpenWeather API при різних умовах запиту, і наостанок — поведінка системи під навантаженням.

Структура пояснювальної записки. Записка кваліфікаційної роботи складається зі вступу, чотирьох основних розділів та висновків.

У першому розділі розглядається предметна область — досліджуються типи забруднюючих речовин та їхні характеристики, аналізуються метеорологічні параметри, що впливають на розповсюдження викидів. Окремо проведено огляд наявних програмних рішень у цій галузі та визначено їхні сильні і слабкі сторони.

Другий розділ присвячений проєктуванню інформаційного забезпечення системи. Тут описано процес побудови логічної моделі даних у вигляді ER-діаграми, обґрунтовано вибір PostgreSQL як СУБД, а також розкрито фізичну структуру бази даних із деталізацією таблиць та зв'язків між ними.

У третьому розділі висвітлюється безпосередня розробка програмного забезпечення. Описано архітектуру системи, принципи організації модулів, реалізацію математичної моделі Гаусівського шлейфу у вигляді програмного коду, а також побудову REST API для взаємодії між клієнтською та серверною частинами.

Четвертий розділ охоплює етап впровадження та експлуатації. Наведено результати тестування, сформульовано вимоги до апаратного і програмного оточення, а також подано покрокові інструкції з розгортування системи.

До складу роботи входить 66 сторінки, список з 21 джерел даних, 24 рисунки, та додатки А, Б, В.

.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз предметної області

Предметна область роботи охоплює процеси розповсюдження забруднюючих речовин в атмосфері в результаті їхнього викиду з точкових та площадних джерел у міських та промислових системах.

Забруднення атмосферного повітря — проблема, яка існує стільки ж, скільки існує промисловість. Заводські труби, котельні, автомагістралі, вентиляційні шахти — усе це щодня викидає в повітря сотні тонн речовин, частина з яких є відверто небезпечною для здоров'я. При цьому саме по собі джерело викиду — це лише половина проблеми. Не менш важливо розуміти, куди і як далеко ці речовини розповсюджуються, і яка їхня концентрація на тій чи іншій відстані.

Розповсюдження забруднень в атмосфері — процес складний і багатофакторний. На нього впливає буквально все: висота труби, температура і швидкість газів на виході, швидкість і напрямок вітру, стан атмосфери, навіть характер підстилаючої поверхні. В умовах нестійкої атмосфери з сильним вітром шлейф забруднень може розсіятися порівняно швидко, тоді як у тиху ясну ніч ті самі речовини здатні накопичуватися у приземному шарі і досягати небезпечних концентрацій на значній відстані від джерела.

Саме тому в екологічній практиці давно використовуються математичні моделі атмосферної дифузії. Вони дозволяють не чекати, поки щось станеться, а заздалегідь прорахувати можливі сценарії. Найбільш поширеною з таких моделей є модель Гаусівського шлейфу — вона відносно проста у реалізації, добре вивчена і при цьому дає достатньо точні результати для більшості практичних задач. За її допомогою можна визначити концентрацію забруднювача в будь-якій точці простору, якщо відомі параметри викиду та метеорологічні умови.

Орієнтиром для оцінки небезпеки слугують граничнодопустимі концентрації — ГДК. Це нормативно встановлені значення, перевищення яких

вже несе реальний ризик для здоров'я людини. Короткочасний вплив понаднормових концентрацій може викликати гострі отруєння, а тривалий — стати причиною хронічних захворювань дихальної та серцево-судинної систем.

Незважаючи на те що інструменти для подібних розрахунків існують давно, більшість із них залишаються малодоступними для широкого кола користувачів. Професійні системи на кшталт AERMOD чи CALPUFF потребують серйозної підготовки, коштують чимало і явно не розраховані на використання в навчальних або невеликих дослідницьких цілях. Це і визначає актуальність розробки простішого, але функціонального веб-орієнтованого інструменту для оцінки розповсюдження забруднень.

1.2 Моделювання предметної області

Перш ніж писати код, варто зрозуміти, що саме ти збираєшся автоматизувати. Звучить очевидно, але на практиці саме на цьому етапі найчастіше виникають проблеми — розробник будує одне, а користувач очікував зовсім інше. Щоб цього уникнути, предметну область прийнято описувати у вигляді формальних моделей.

Модель — це спрощений опис реальності, який виділяє головне і відкидає зайве. Вона може описувати об'єкти системи та їхні властивості, процеси що відбуваються, зв'язки між учасниками або типові сценарії роботи користувача. Залежно від того, що саме потрібно показати, використовуються різні види діаграм та нотацій.

Для системи моделювання розповсюдження забруднень такий підхід особливо виправданий, оскільки предметна область поєднує в собі і фізичні процеси, і взаємодію користувача з інтерфейсом, і складну розрахункову логіку. Чітке структурування цих аспектів на етапі аналізу суттєво спрощує подальше проєктування та реалізацію системи.

1.2.1 Діаграма прецедентів.

Діаграма прецедентів — один із базових інструментів UML, який використовується на етапі аналізу предметної області. Її головна цінність у тому, що вона дозволяє одразу побачити, хто взаємодіє з системою і що саме ці учасники роблять, не заглиблюючись у технічні деталі реалізації.

Основних елементів тут два. Актори — це всі, хто так чи інакше взаємодіє з системою: люди, зовнішні сервіси, інші програмні компоненти. Прецеденти — це конкретні дії або сценарії, які актор виконує для досягнення певної мети. Між прецедентами можуть існувати зв'язки — зокрема, відношення include вказує на те, що один прецедент обов'язково включає виконання іншого.

На рис. 1 представлено діаграму прецедентів системи AeroSpread Pro. В системі визначено трьох акторів: Користувач, Адміністратор та зовнішній сервіс OpenWeather API. Користувач виконує основні розрахункові функції — налаштовує параметри викиду, запускає моделювання, переглядає результати на карті, контролює перевищення ГДК та експортує звіти. Отримання метеоданих винесено в окремий прецедент із зв'язком include, оскільки воно є обов'язковою складовою як запуску розрахунку, так і налаштування параметрів. Адміністратор має власну зону відповідальності — управління обліковими записами користувачів та базою даних забруднюючих речовин.

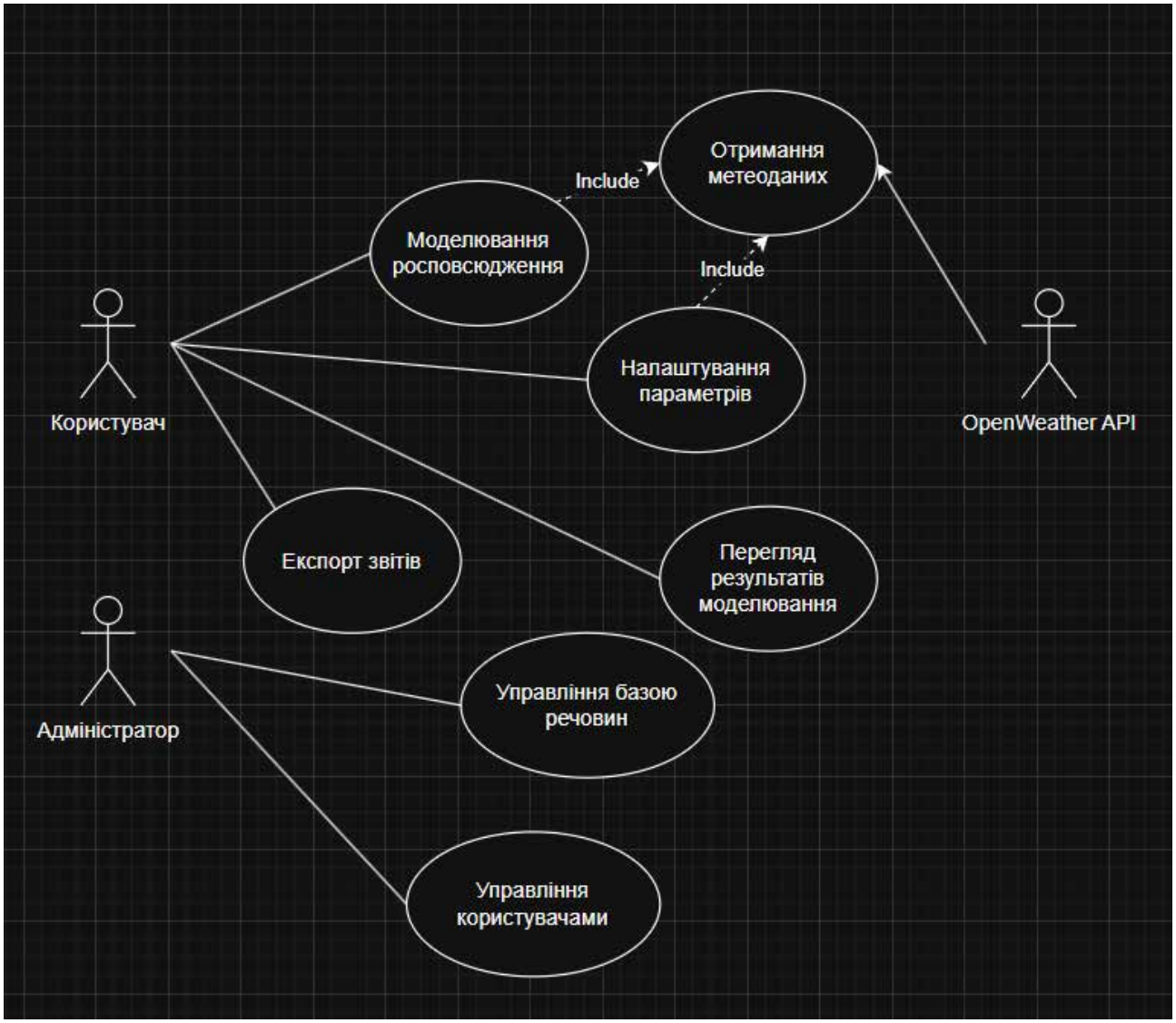


Рис. 1 Діаграма прецедентів ПЗ оцінки розповсюдження забруднюючої аерозольної хмари в антропогенному середовищі.

Опис акторів

У системі визначено двох основних акторів та один зовнішній сервіс, кожен з яких має чітко окреслену зону відповідальності.

Користувач — це фахівець або дослідник, який безпосередньо працює з системою: створює проєкти, задає параметри викиду та метеорологічні умови, запускає розрахунки і аналізує отримані результати. Саме він є головним споживачем функціональних можливостей системи.

Адміністратор — відповідає за підтримку коректної роботи системи в цілому. До його функцій належить управління обліковими записами користувачів та актуалізація бази даних забруднюючих речовин.

OpenWeather API — зовнішній сервіс, який виступає джерелом актуальних метеорологічних даних. Система звертається до нього автоматично під час налаштування параметрів розрахунку або запуску моделювання.

Опис прецедентів

Користувач:

- налаштування параметрів викиду — введення висоти джерела, потужності, температури газів та вибір забруднюючої речовини з бази даних;
- запуск розрахунку — ініціація моделювання розповсюдження забруднень за моделлю Гаусівського шлейфу;
- перегляд результатів на карті — візуальний аналіз розподілу концентрацій у вигляді ізоліній та теплової карти;
- контроль перевищення ГДК — автоматична перевірка розрахованих концентрацій відносно нормативних значень;
- експорт звіту — формування та завантаження звіту у форматі PDF.

Адміністратор:

- управління користувачами — перегляд, додавання та блокування облікових записів;
- управління базою речовин — додавання нових забруднювачів, редагування їхніх характеристик та нормативних показників.

OpenWeather API:

- надання метеоданих — передача поточних значень швидкості та напрямку вітру, температури, вологості та атмосферного тиску за географічними координатами об'єкта.

Діаграма прецедентів демонструє загальну структуру організаційної взаємодії у предметній області що дає змогу краще зрозуміти логіку процесів та ролі учасників перед переходом до проєктування інформаційної системи.

1.2.2 Діаграма послідовності.

Діаграма послідовності — це ще один тип UML-діаграм, який показує не структуру системи, а її поведінку в часі. Якщо діаграма прецедентів відповідає на питання "хто що робить", то діаграма послідовності показує "як саме це відбувається" — в якому порядку, між якими учасниками і які повідомлення при цьому передаються.

Основу діаграми складають кілька елементів. Об'єкти розміщуються вгорі — кожен із них має вертикальну лінію життєвого циклу, що тягнеться вниз і позначає його існування протягом усього сценарію. Горизонтальні стрілки між цими лініями — це повідомлення, тобто виклики, запити або відповіді, що передаються від одного учасника до іншого. У моменти активної роботи на лінії з'являється смуга активації, яка наочно показує, коли саме об'єкт щось обробляє.

На рис. 2 представлено діаграму послідовності для сценарію запуску розрахунку розповсюдження забруднень у системі AeroSpread Pro. Вона охоплює взаємодію між користувачем, клієнтською частиною застосунку, серверною частиною, базою даних та зовнішнім сервісом OpenWeather API — від моменту введення параметрів до отримання результатів моделювання на карті.

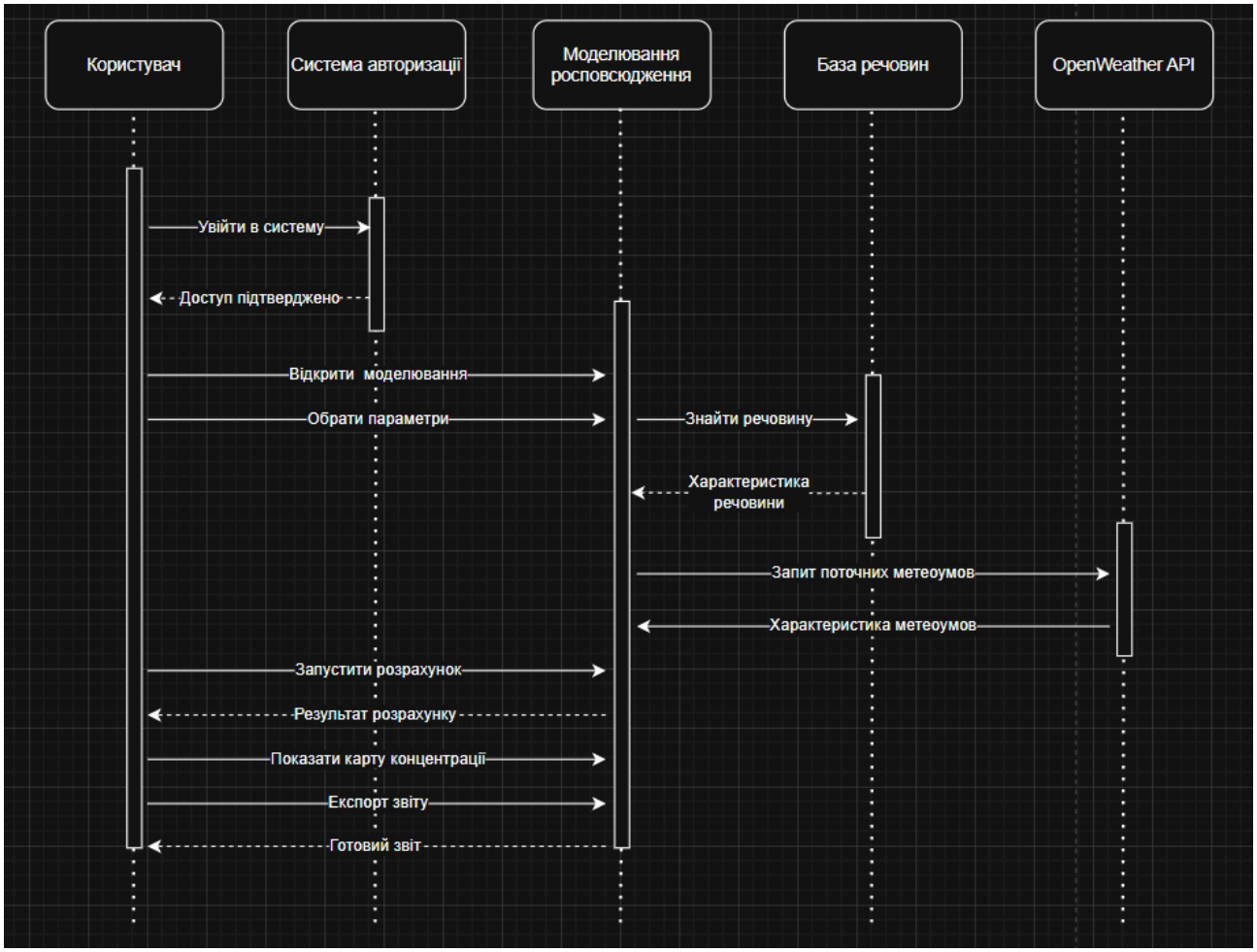


Рис. 2 Діаграма послідовності ПЗ оцінки розповсюдження забруднюючої аерозольної хмари в антропогенному середовищі.

Опис акторів та об'єктів діаграми

- *Користувач* — основний актор системи, який ініціює весь процес моделювання: від входу в систему до отримання готового звіту.
- *Система авторизації* — відповідає за перевірку облікових даних та надання доступу до функціоналу.
- *Модуль моделювання розповсюдження* — центральний компонент системи, який координує отримання даних, виконує розрахунки та повертає результати.
- *База речовин* — зберігає фізико-хімічні характеристики забруднюючих речовин та їхні нормативні показники ГДК.
- *OpenWeather API* — зовнішній сервіс, що надає актуальні метеорологічні дані за географічними координатами об'єкта.

Опис сценарію

Авторизація:

- користувач надсилає запит на вхід у систему;
- система авторизації перевіряє облікові дані та підтверджує доступ.

Налаштування параметрів:

- користувач відкриває модуль моделювання та обирає параметри розрахунку;
- модуль звертається до бази речовин для отримання характеристик обраного забруднювача;
- база повертає фізико-хімічні властивості речовини та значення ГДК;
- модуль надсилає запит до OpenWeather API для отримання поточних метеоумов;
- сервіс повертає значення швидкості та напрямку вітру, температури і тиску.

Виконання розрахунку:

- користувач запускає моделювання;
- модуль виконує обчислення за моделлю Гаусівського шлейфу з урахуванням усіх отриманих даних;
- результати розрахунку повертаються користувачу у вигляді карти концентрацій із позначенням зон перевищення ГДК.

Експорт звіту:

- користувач ініціює формування звіту;
- модуль генерує документ у форматі PDF та повертає його користувачу у готовому вигляді.

Представлений сценарій відображає повний цикл роботи користувача в системі AeroSpread Pro — від авторизації до отримання результатів моделювання. Діаграма наочно показує часову послідовність взаємодій між компонентами системи та допомагає зрозуміти логіку обміну даними між її складовими частинами.

1.2.3 Діаграма активності. Діаграма активності дозволяє відобразити не просто що робить система, а в якій послідовності і за якою логікою це відбувається. На відміну від діаграми прецедентів, яка показує лише перелік функцій, діаграма активності розкриває внутрішній сценарій їх виконання - з розгалуженнями, умовами та можливими альтернативними шляхами.

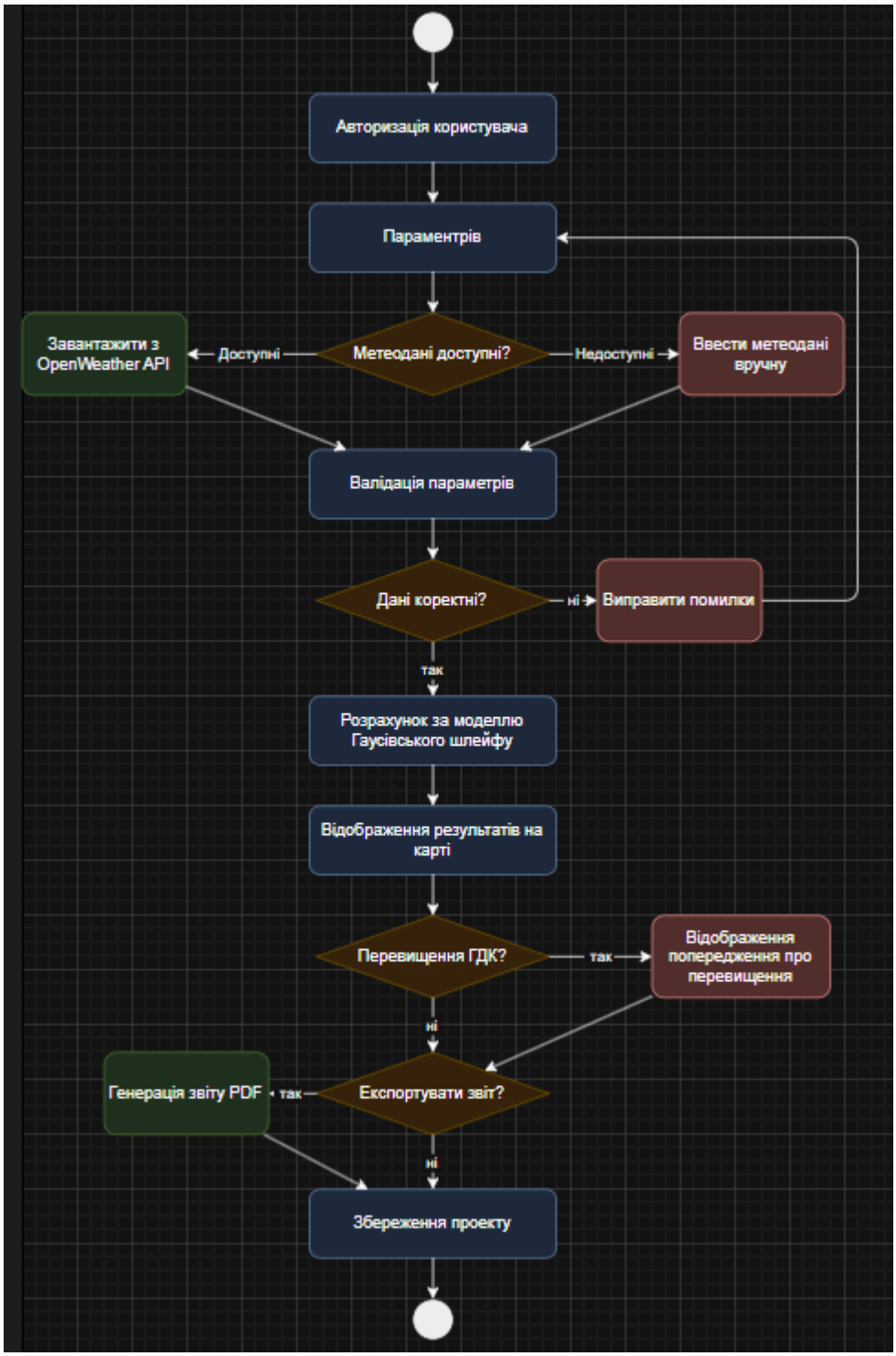


Рис. 3 Діаграма активності ПЗ оцінки розповсюдження забруднюючої аерозольної хмари в антропогенному середовищі.

Опис діаграми. Процес починається з авторизації користувача в системі. Після успішного входу користувач вводить географічні координати об'єкта та обирає забруднюючу речовину з бази даних. Наступним кроком є введення параметрів викиду — висоти джерела, потужності, температури газів тощо.

Після цього система перевіряє доступність метеорологічних даних. Якщо з'єднання з OpenWeather API доступне — дані завантажуються автоматично. В іншому випадку користувач вводить метеопараметри вручну.

Перед запуском розрахунку виконується валідація всіх введених параметрів. Якщо виявлено некоректні значення — користувач повертається до етапу введення параметрів для їх виправлення. Якщо дані коректні — запускається розрахунок концентрацій за моделлю Гаусівського шлейфу.

Результати відображаються на інтерактивній карті. Система автоматично перевіряє, чи перевищують розраховані концентрації нормативні значення ГДК - у такому разі формується відповідне попередження з кольоровим кодуванням зон забруднення. Після аналізу результатів користувач може експортувати звіт у форматі PDF або завершити роботу, зберігши проект для подальшого використання.

1.3 Огляд існуючих аналогічних систем

На сучасному ринку існує кілька рішень для розрахунку розповсюдження забруднень в атмосфері, кожне з яких має свої переваги та недоліки.

CALPUFF - одна з найбільш відомих і авторитетних систем для моделювання розповсюдження забруднень в атмосфері. Розроблена за замовленням Агентства з охорони навколишнього середовища США (EPA) і протягом десятиліть використовується як еталонний інструмент для екологічної оцінки промислових об'єктів у багатьох країнах світу. Система реалізує нестаціонарну лагранжеву модель, що дозволяє враховувати складні метеорологічні умови, змінний напрямок вітру, рельєф місцевості та хімічні

перетворення речовин в атмосфері. Діапазон розрахунків охоплює відстані від кількох сотень метрів до 3000 км від джерела викиду.

Переваги:

- офіційно рекомендована EPA і визнана міжнародними екологічними організаціями;
- висока точність розрахунків навіть у складних метеорологічних умовах;
- підтримка широкого спектру забруднюючих речовин та типів джерел викиду;
- детальне врахування рельєфу та приземного шару атмосфери.

Недоліки:

- складна у встановленні та налаштуванні, потребує спеціальної підготовки;
- відсутній веб-інтерфейс - робота виключно через командний рядок;
- потребує окремого підготовленого метеорологічного файлу у специфічному форматі;
- практично недоступна для використання в освітніх або дослідницьких цілях без відповідної інфраструктури.

AERMOD — це вдосконалена стаціонарна модель розповсюдження забруднень, розроблена спільно EPA та Американським метеорологічним товариством як сучасна альтернатива застарілій моделі ISC3. В основі системи лежить модель Гаусівського шлейфу, проте суттєво розширена - вона враховує ефекти рельєфу місцевості, конвективне перемішування в примежовому шарі атмосфери, а також коректно обробляє слабкі точкові джерела, що є критично важливим для міських промислових зон. Наразі AERMOD є основним нормативним інструментом EPA для оцінки якості повітря поблизу промислових підприємств на відстанях до 50 км.

Переваги:

- офіційно затверджена EPA як нормативна модель для більшості задач екологічної оцінки;
- коректно враховує складний рельєф та турбулентність примежового шару;

- підтримує різні типи джерел — точкові, лінійні, площадні та об'ємні;
- добре задокументована, існує велика база наукових публікацій.

Недоліки:

- відсутній зручний графічний інтерфейс, налаштування виконується через текстові файли конфігурації;
- потребує окремо підготовлених метеорологічних даних у форматі AERMET;
- висока складність налаштування без відповідної технічної підготовки;
- ліцензована версія з повноцінною підтримкою є платною, що обмежує її використання в навчальних цілях.

ALOHA (Areal Locations of Hazardous Atmospheres) — безкоштовна система моделювання розповсюдження небезпечних речовин в атмосфері, розроблена спільно Національним управлінням океанічних і атмосферних досліджень США (NOAA) та Агентством з охорони навколишнього середовища (EPA). Система створювалась передусім для підтримки оперативного реагування на аварійні викиди — вибухи, розливи токсичних речовин, пожежі на промислових об'єктах. Завдяки цьому вона набула широкого поширення серед рятувальних служб, підрозділів цивільного захисту та екологічних інспекцій у багатьох країнах світу.

Переваги:

- повністю безкоштовна і офіційно підтримується NOAA та EPA;
- має зрозумілий графічний інтерфейс, не потребує знання командного рядка;
- працює офлайн — не залежить від наявності інтернет-з'єднання;
- містить вбудовану базу даних понад 1000 хімічних речовин;
- широко використовується і добре задокументована.

Недоліки:

- орієнтована переважно на аварійні сценарії, а не на систематичний моніторинг промислових викидів;

- відсутня веб-версія — потребує встановлення на локальний комп'ютер;
- обмежені можливості візуалізації результатів порівняно з сучасними веб-застосунками;
- не підтримує інтеграцію з зовнішніми метеорологічними сервісами в реальному часі.

1.4 Постановка завдання

На основі аналізу предметної області та огляду існуючих рішень встановлено, що наявні інструменти або надто складні у використанні, або не охоплюють повний цикл роботи з даними. Це обумовлює необхідність розробки доступного веб-застосунку з інтуїтивним інтерфейсом.

Сформульовано наступне завдання: розробити веб-застосування AeroSpread Pro для оцінки розповсюдження забруднюючої аерозольної хмари, яке реалізує модель Гаусівського шлейфу, інтегрується з OpenWeather API для отримання актуальних метеоданих, візуалізує результати на інтерактивній карті з позначенням зон перевищення ГДК та забезпечує експорт звітів у форматі PDF.

1.4.1 Функціональні вимоги

Система повинна забезпечити виконання таких функцій:

1. Управління користувачами: реєстрація нових облікових записів та вхід у систему.

2. Управління проектами: створення проектів із зазначенням назви об'єкта та географічних координат; редагування і видалення проектів;; пошук і фільтрація по збережених проектах.

3. Параметри викиду та метеодані: введення висоти джерела, потужності викиду та температури газів; вибір класу стійкості атмосфери; ручне введення метеорологічних параметрів або їх автоматичне отримання з OpenWeather API за координатами об'єкта.

4. База даних забруднюючих речовин: зберігання характеристик 15+ речовин — назви, CAS-номер, молекулярна маса, щільність, ГДК, клас

небезпеки; пошук речовин; можливість додавання нових речовин адміністратором.

5. Математичні розрахунки: обчислення концентрацій на відстанях від 10 до 5000 м за моделлю Гаусівського шлейфу; визначення максимальної концентрації та відстані до неї; розрахунок радіусу зони перевищення ГДК.

6. Візуалізація результатів: відображення результатів на інтерактивній карті Leaflet/OpenStreetMap з маркером джерела викиду; теплова карта з кольоровим кодуванням зон — від критичного забруднення до фонового рівня; таблиця з детальними значеннями концентрацій; графік залежності концентрації від відстані.

7. Експорт: вивантаження звіту у форматі PDF з таблицями та графіками; експорт у CSV для подальшої обробки; збереження результатів у базі даних.

1.4.2 Нефункціональні вимоги

Окрім функціональних, до системи висунуто ряд технічних вимог. Розрахунок для одного проекту має виконуватись не довше 1–3 секунд. Система повинна витримувати одночасну роботу щонайменше 10 користувачів без помітного зниження продуктивності. З точки зору безпеки передбачено використання JWT-токенів для аутентифікації, HTTPS для передачі даних. Інтерфейс має бути зрозумілим без додаткового навчання і підтримувати всі актуальні браузери.

1.4.3 Вхідні дані системи

Система приймає та обробляє такі вхідні дані:

- параметри джерела викиду: висота, потужність, температура газів, діаметр устя, швидкість виходу суміші;
- географічні координати об'єкта: широта та довгота;
- метеорологічні дані: швидкість і напрямок вітру, температура повітря, атмосферний тиск, вологість, клас стійкості атмосфери;

- характеристики забруднюючої речовини: назва, CAS-номер, молекулярна маса, ГДК;
- облікові дані користувача: логін, пароль.

1.4.4 Вихідні дані системи

За результатами розрахунків система формує:

- теплову карту розповсюдження забруднення на інтерактивній карті з кольоровим кодуванням зон перевищення ГДК;
- таблицю концентрацій із зазначенням відстані від джерела та відсотку від граничнодопустимого значення;
- звіт у форматі PDF з усіма параметрами розрахунку та отриманими результатами;

2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Логічна модель даних у вигляді ER-діаграми

Перш ніж переходити до фізичної реалізації бази даних, необхідно чітко визначити, які дані зберігатимуться в системі і як вони пов'язані між собою. Саме для цього будується логічна модель даних — вона описує структуру інформації на рівні сутностей та зв'язків, ще не прив'язуючись до конкретної СУБД чи технічних деталей реалізації. Наочним способом представити таку модель є ER-діаграма, яка дозволяє побачити всю структуру даних одним поглядом і виявити можливі протиріччя ще до початку розробки.

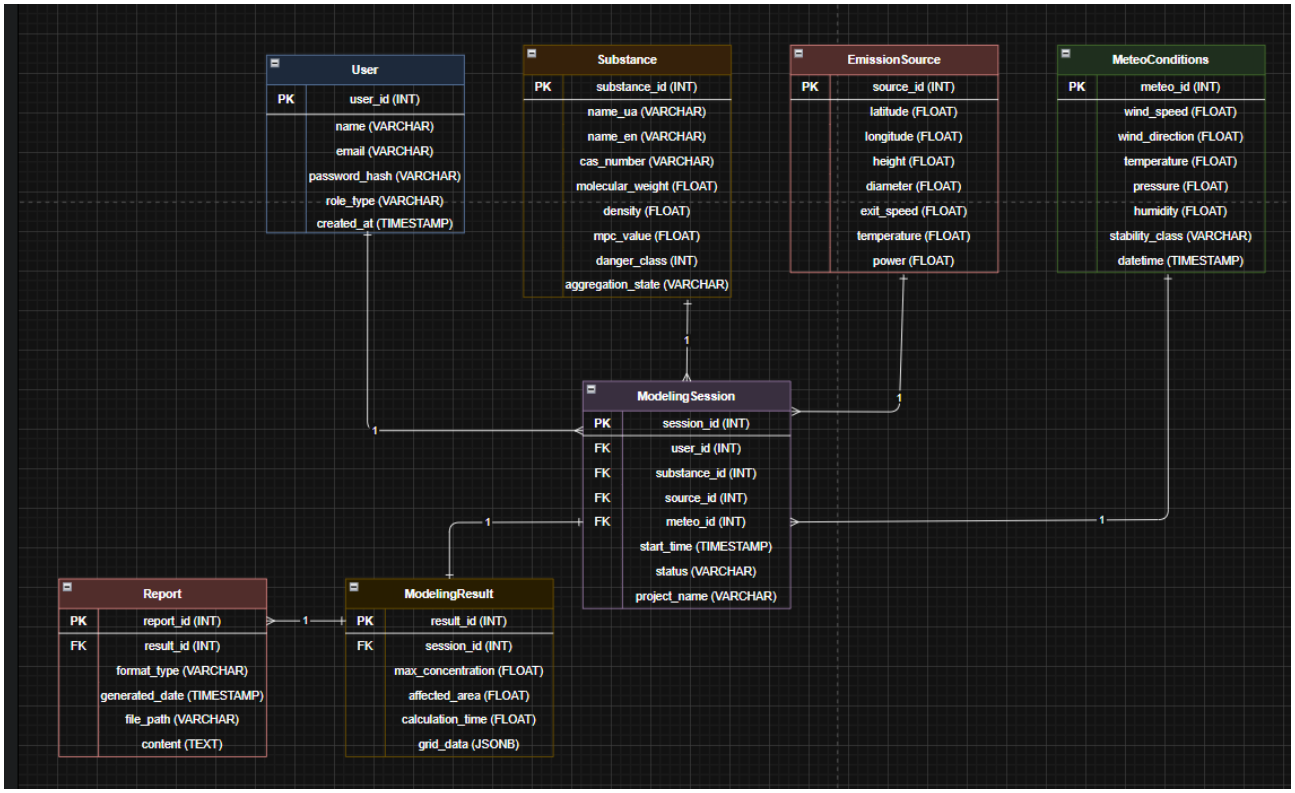


Рис. 4 ER-діаграма ПЗ оцінки розповсюдження забруднюючої аерозольної хмари в антропогенному середовищі.

Опис сутностей та атрибутів

User (Користувач) — базова сутність:

- user_id — унікальний ідентифікатор користувача
- name — ім'я користувача
- email — електронна адреса
- password_hash — хеш паролю (bcrypt)
- role_type — роль у системі (user / admin)
- created_at — дата реєстрації

Substance (Забруднююча речовина) — базова сутність:

- substance_id — унікальний ідентифікатор речовини
- name_ua — назва речовини українською
- name_en — назва речовини англійською
- cas_number — міжнародний реєстраційний номер CAS
- molecular_weight — молекулярна маса
- density — щільність речовини
- mpc_value — значення ГДК максимально разове
- danger_class — клас небезпеки (1–4)
- aggregation_state — агрегатний стан (газ, рідина, аерозоль)

EmissionSource (Джерело викиду) — базова сутність:

- source_id — унікальний ідентифікатор джерела
- latitude — географічна широта
- longitude — географічна довгота
- height — висота джерела над поверхнею землі, м
- diameter — діаметр устя джерела, м
- exit_speed — швидкість виходу газоповітряної суміші, м/с
- temperature — температура газів на виході, °C
- power — потужність викиду, г/с

MeteoConditions (Метеорологічні умови) — базова сутність:

- meteo_id — унікальний ідентифікатор запису
- wind_speed — швидкість вітру, м/с

- wind_direction — напрямок вітру, градуси
- temperature — температура повітря, °C
- pressure — атмосферний тиск, гПа
- humidity — відносна вологість, %
- stability_class — клас стійкості атмосфери за Паскіль–Гіффордом (A–F)
- datetime — дата і час отримання метеоданих

ModelingSession (Сеанс моделювання) — дочірня сутність:

- session_id — унікальний ідентифікатор сеансу
- user_id (FK) — посилання на користувача
- substance_id (FK) — посилання на забруднюючу речовину
- source_id (FK) — посилання на джерело викиду
- meteo_id (FK) — посилання на метеорологічні умови
- project_name — назва проєкту
- start_time — дата і час запуску розрахунку
- status — поточний стан сеансу (виконується / завершено / помилка)

ModelingResult (Результат моделювання) — дочірня сутність:

- result_id — унікальний ідентифікатор результату
- session_id (FK) — посилання на сеанс моделювання
- max_concentration — максимальна розрахована концентрація, мг/м³
- affected_area — площа зони перевищення ГДК, м²
- calculation_time — час виконання розрахунку, с
- grid_data — сітка концентрацій у форматі JSONB для побудови теплової карти

Report (Звіт) — дочірня сутність:

- report_id — унікальний ідентифікатор звіту
- result_id (FK) — посилання на результат моделювання
- format_type — формат звіту (PDF / CSV)
- generated_date — дата та час формування звіту
- file_path — шлях до файлу звіту на сервері
- content — вміст звіту у текстовому вигляді

Опис зв'язків

- Один користувач може створювати необмежену кількість сеансів моделювання. Зв'язок реалізовано через зовнішній ключ user_id у таблиці ModelingSession, що посилається на первинний ключ таблиці User.
- Одна забруднююча речовина може використовуватись у багатьох сеансах, однак кожен сеанс прив'язаний до однієї конкретної речовини. Зв'язок реалізовано через зовнішній ключ substance_id у таблиці ModelingSession.
- Одне джерело викиду може фігурувати в кількох сеансах моделювання. Зв'язок здійснюється через зовнішній ключ source_id у таблиці ModelingSession.
- Кожен сеанс використовує один набір метеорологічних умов, зафіксованих на момент розрахунку. Зв'язок реалізовано через зовнішній ключ meteo_id у таблиці ModelingSession.
- Кожен сеанс моделювання породжує рівно один результат. Зв'язок між ModelingSession та ModelingResult є відношенням один до одного і реалізується через зовнішній ключ session_id у таблиці ModelingResult.
- Один результат моделювання може мати кілька звітів у різних форматах. Зв'язок між ModelingResult та Report реалізовано через зовнішній ключ result_id у таблиці Report.

Перевірка відповідності нормальним формам

Щоб уникнути надлишковості даних та забезпечити їхню цілісність, розроблена модель приведена до третьої нормальної форми. Нормалізація передбачає послідовне усунення аномалій зберігання даних шляхом дотримання певних правил організації таблиць і залежностей між атрибутами.

Перша нормальна форма (1НФ)

Перша нормальна форма вимагає атомарності всіх атрибутів — кожне поле таблиці повинно містити лише одне неподільне значення. У розробленій моделі це правило витримано в усіх таблицях. Зокрема, дані про речовину не зберігаються одним рядком — кожна характеристика винесена в окреме поле: назва, CAS-номер, щільність, ГДК тощо. Сітка концентрацій зберігається у полі

grid_data типу JSONB як єдиний структурований об'єкт, а не як множина окремих рядків, що відповідає вимогам 1НФ. У кожній таблиці визначено первинний ключ, що гарантує унікальність записів.

Друга нормальна форма (2НФ)

Друга нормальна форма вимагає, щоб усі неключові атрибути повністю залежали від первинного ключа, а не від його частини. В розробленій моделі всі таблиці мають прості первинні ключі, тому часткових залежностей виникнути не може. Наприклад, у таблиці ModelingSession атрибути project_name, start_time та status залежать виключно від session_id, а не від будь-якого з зовнішніх ключів окремо. Аналогічно побудовані й інші таблиці, що підтверджує відповідність 2НФ.

Третя нормальна форма (3НФ)

Третя нормальна форма виключає транзитивні залежності — ситуації, коли один неключовий атрибут залежить від іншого неключового атрибуту. У розробленій моделі характеристики забруднюючих речовин не зберігаються у таблиці ModelingSession, а винесені в окрему таблицю Substance — зв'язок здійснюється через зовнішній ключ. Так само метеорологічні умови виокремлені в таблицю MeteoConditions, а параметри джерела — в EmissionSource. Завдяки такому підходу жоден атрибут не залежить від іншого неключового поля, що повністю відповідає вимогам третьої нормальної форми.

Таким чином, логічна модель даних системи AeroSpread Pro відповідає всім трьом нормальним формам. Це дає змогу уникнути дублювання інформації, мінімізувати ризик аномалій при оновленні чи видаленні записів і забезпечити стабільну роботу бази даних у довгостроковій перспективі.

2.2 Вибір системи управління базами даних

Вибір системи управління базами даних — не менш важливе рішення, ніж вибір мови програмування. Від нього залежить надійність зберігання даних, зручність роботи з ними та можливість масштабування системи в майбутньому.

Після аналізу доступних варіантів для реалізації логічної моделі даних обрано PostgreSQL версії 12 і вище.

PostgreSQL — це відкрита об'єктно-реляційна СУБД з багаторічною історією розвитку. За понад двадцять років активного використання вона здобула репутацію одного з найнадійніших інструментів у своїй категорії. Система повністю підтримує принципи ACID, що гарантує цілісність і консистентність даних навіть у випадку збоїв. Серед технічних можливостей варто відзначити підтримку типу JSONB, який у даному проєкті використовується для збереження сіток концентрацій та ізоліній — структур, що погано вкладаються в класичну реляційну схему. PostgreSQL також добре інтегрується з Node.js через бібліотеку pg, що спрощує взаємодію між серверною частиною та базою даних. Окремою перевагою є безкоштовна ліцензія, яка не накладає обмежень на використання та модифікацію.

Паралельно розглядались інші варіанти. MySQL є простішим і добре відомим рішенням, однак поступається PostgreSQL у роботі зі складними запитами та нестандартними типами даних. MongoDB як документно-орієнтована база даних могла б підійти для зберігання JSON-структур, але погано справляється з реляційними зв'язками між сутностями, які є ключовими для даної моделі. SQLite зручна для локальної розробки, проте не розрахована на мережеві застосунки з одночасним доступом кількох користувачів.

Таким чином, PostgreSQL виявилась найбільш збалансованим вибором з огляду на функціональність, надійність та відповідність технічному стеку проєкту.

2.3 Розробка структури інформаційної бази

Структура інформаційної бази є фундаментом будь-якої програмної системи — від того, наскільки грамотно вона спроєктована, залежить і швидкість роботи застосунку, і зручність підтримки, і можливість розширення функціональності в майбутньому. Саме тому на етапі розробки AeroSpread Pro цьому питанню приділено особливу увагу.

База даних системи побудована не лише як набір таблиць для зберігання даних, а як повноцінний рівень обробки інформації. Окрім основних таблиць, структура включає уявлення для спрощення складних вибірок, збережені процедури для автоматизації повторюваних операцій, тригери для підтримки цілісності даних та налаштованих користувачів з розмежованими правами доступу. Такий підхід дозволяє перенести частину логіки на рівень СУБД, що знижує навантаження на серверну частину і зменшує ризик помилок при роботі з даними. У наступних підрозділах детально описано реалізацію кожного з цих компонентів.

2.3.1 Створення таблиць

Основою будь-якої реляційної бази даних є таблиці — саме в них зберігаються всі дані системи. Для AeroSpread Pro було розроблено шість основних таблиць, кожна з яких відповідає окремій сутності логічної моделі. Усі таблиці створюються з перевіркою на існування через конструкцію IF NOT EXISTS, що дозволяє уникнути помилок при повторному розгортанні бази.

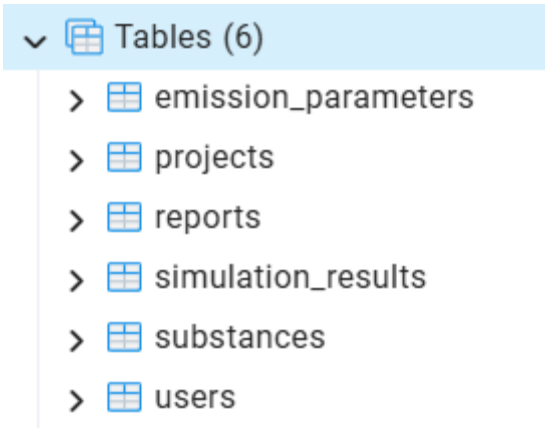


Рис. 5 Створені таблиці ПЗ оцінки розповсюдження забруднюючої аерозольної хмари в антропогенному середовищі.

Таблиця users зберігає облікові дані користувачів системи — ім'я, електронну пошту, хеш паролю, роль та статус активності. Таблиця substances містить базу даних забруднюючих речовин із їхніми фізико-хімічними характеристиками та нормативними значеннями ГДК. Таблиця projects об'єднує проєкти моделювання, прив'язані до конкретного користувача і географічних

координат об'єкта. Параметри кожного розрахунку — висота джерела, потужність викиду, метеорологічні умови тощо — зберігаються в таблиці emission_parameters. Результати моделювання, включно із сіткою концентрацій у форматі JSONB, розміщуються в таблиці simulation_results. Нарешті, таблиця reports фіксує метадані згенерованих звітів та шляхи до файлів.

Повний код створення всіх таблиць наведено у Додатку А.

2.3.2 Початкове наповнення бази даних

Для коректної роботи системи відразу після розгортання база даних наповнюється початковим набором забруднюючих речовин. До таблиці substances внесено 15 типових промислових забруднювачів — діоксид азоту, оксид вуглецю, діоксид сірки, аміак, бензол, толуол, фенол, хлор, формальдегід, озон, сірководень, свинець, ртуть та дрібнодисперсні частинки PM2.5 і PM10. Для кожної речовини вказано CAS-номер, молекулярну масу, щільність, значення ГДК максимально разової та середньодобової, клас небезпеки і агрегатний стан. Вставка виконується з конструкцією ON CONFLICT DO NOTHING, що запобігає дублюванню даних при повторному запуску скрипту.

```
-- Початкові дані: типові забруднюючі речовини для моделювання

INSERT INTO substances (name_ua, name_en, cas_number, molecular_weight, density, mac_single, mac_daily, danger_class, aggregat
('Діоксид азоту', 'Nitrogen dioxide', '10102-44-0', 46.01, 1.448, 0.085, 0.04, 2, 'газ', 'Газ бурого кольору з різким запахом.
('Оксид вуглецю', 'Carbon monoxide', '630-08-0', 28.01, 0.789, 5.0, 3.0, 4, 'газ', 'Безбарвний отруйний газ без запаху. Неповне
('Діоксид сірки', 'Sulfur dioxide', '7446-09-5', 64.07, 2.263, 0.5, 0.05, 3, 'газ', 'Безбарвний газ з різким запахом. Спалюванн
('Сірководень', 'Hydrogen sulfide', '7783-06-4', 34.08, 1.363, 0.008, null, 2, 'газ', 'Безбарвний газ із запахом тухлих яєць. К
('Аміак', 'Ammonia', '7664-41-7', 17.03, 0.771, 0.2, 0.04, 4, 'газ', 'Безбарвний газ з різким запахом. Хімічна промисловість, х
('Формальдегід', 'Formaldehyde', '50-00-0', 30.03, 0.815, 0.035, 0.01, 2, 'газ', 'Безбарвний газ з різким запахом. Деревообробк
('Бензол', 'Benzene', '71-43-2', 78.11, 0.879, 0.3, 0.1, 2, 'рідина/пара', 'Безбарвна рідина із специфічним запахом. Нафтохімія
('Толуол', 'Toluene', '108-88-3', 92.14, 0.867, 0.6, 0.6, 3, 'рідина/пара', 'Безбарвна рідина. Розчинники, фарби, лаки.'),
('Фенол', 'Phenol', '108-95-2', 94.11, 1.071, 0.01, 0.003, 2, 'тверда/пара', 'Кристалічна речовина. Хімічне виробництво, дезінф
('Хлор', 'Chlorine', '7782-50-5', 70.91, 2.898, 0.1, 0.03, 2, 'газ', 'Жовто-зелений газ з різким запахом. Водоочистка, хімічна
('Завислі частинки (PM10)', 'Particulate Matter PM10', null, null, null, 0.3, 0.15, 3, 'аерозоль', 'Тверді або рідкі частинки р
('Завислі частинки (PM2.5)', 'Particulate Matter PM2.5', null, null, null, 0.16, 0.035, 2, 'аерозоль', 'Дрібні частинки розміро
('Озон', 'Ozone', '10028-15-6', 48.00, 1.962, 0.16, 0.03, 1, 'газ', 'Блакитний газ з різким запахом. Фотохімічний смог.'),
('Свинець та його сполуки', 'Lead compounds', '7439-92-1', 207.2, 11.34, 0.001, 0.0003, 1, 'аерозоль', 'Високотоксичний метал.
('Ртуть та її сполуки', 'Mercury compounds', '7439-97-6', 200.59, 13.534, 0.0003, null, 1, 'рідина/пара', 'Надзвичайно токсична
ON CONFLICT DO NOTHING;
```

Рис. 6 Початкове наповнення бази даних

2.3.3 Індекси

Для прискорення виконання найбільш поширених запитів до бази даних створено п'ять індексів. Індекси побудовано на полях зовнішніх ключів таблиць projects, emission_parameters, simulation_results та reports — це суттєво пришвидшує вибірку даних за ідентифікатором користувача або проєкту.

Окремо створено індекс на полі name_ua таблиці substances для оптимізації пошуку речовини за назвою.

```
-- Індекс для швидкого пошуку проєктів за користувачем
CREATE INDEX idx_projects_user_id
    ON projects(user_id);

-- Індекс для вибірки параметрів викиду за проєктом
CREATE INDEX idx_emission_parameters_project_id
    ON emission_parameters(project_id);

-- Індекс для вибірки результатів моделювання за проєктом
CREATE INDEX idx_simulation_results_project_id
    ON simulation_results(project_id);

-- Індекс для вибірки звітів за проєктом
CREATE INDEX idx_reports_project_id
    ON reports(project_id);

-- Індекс для пошуку речовин за українською назвою
CREATE INDEX idx_substances_name
    ON substances(name_ua);
```

Рис. 7 Створення індексів

2.3.4 Тригери

Одним із механізмів автоматизації в базі даних є тригери — процедури, що виконуються автоматично у відповідь на певну подію. У системі AeroSpread Pro реалізовано тригер update_updated_at_column, який автоматично оновлює поле updated_at при кожній зміні запису. Тригер підключено до таблиць users, substances, projects та emission_parameters. Завдяки цьому час останнього оновлення фіксується на рівні бази даних без необхідності явно передавати його з серверної частини застосунку.

```sql
-- Функція оновлення updated_at
CREATE OR REPLACE FUNCTION update_updated_at_column()
RETURNS TRIGGER AS $$
BEGIN
    NEW.updated_at = CURRENT_TIMESTAMP;
    RETURN NEW;
END;
$$ language 'plpgsql';

-- Тригер для таблиці users
CREATE TRIGGER update_users_updated_at
    BEFORE UPDATE ON users
    FOR EACH ROW EXECUTE FUNCTION update_updated_at_column();

-- Тригер для таблиці substances
CREATE TRIGGER update_substances_updated_at
    BEFORE UPDATE ON substances
    FOR EACH ROW EXECUTE FUNCTION update_updated_at_column();

-- Тригер для таблиці projects
CREATE TRIGGER update_projects_updated_at
    BEFORE UPDATE ON projects
    FOR EACH ROW EXECUTE FUNCTION update_updated_at_column();

-- Тригер для таблиці emission_parameters
CREATE TRIGGER update_emission_parameters_updated_at
    BEFORE UPDATE ON emission_parameters
    FOR EACH ROW EXECUTE FUNCTION update_updated_at_column();
```

Рис. 8 Створення тригерів

2.3.5 Міграції

У процесі розробки структура бази даних зазнавала змін відповідно до нових вимог. Для внесення змін без втрати існуючих даних використовувались міграції — окремі SQL-скрипти, що додають або модифікують елементи схеми. Зокрема, до таблиці emission_parameters було додано поля wind_direction_code, substance_name, substance_name_en та mac_single для підтримки розширеного формату вхідних даних. Поле substance_id переведено з обов'язкового у необов'язкове, що дозволило вводити параметри речовини вручну без прив'язки до запису в базі. Також додано таблицю email_verifications для зберігання токенів підтвердження електронної пошти та поле is_verified у таблиці users.

2.4 Схема та фізична структура бази даних

Фізична структура бази даних AeroSpread Pro реалізована засобами PostgreSQL з використанням SQL-запитів. Такий підхід дозволив точно визначити типи даних для кожного поля, встановити обмеження цілісності та налаштувати зв'язки між таблицями відповідно до логічної моделі.

Загалом у базі даних реалізовано 6 основних таблиць:

- **users** — зберігає облікові дані користувачів: ім'я, електронну пошту, хеш паролю, роль та статус активності;
- **substances** — довідкова таблиця забруднюючих речовин із фізико-хімічними характеристиками та значеннями ГДК;
- **projects** — проєкти моделювання, прив'язані до користувача та географічних координат об'єкта;
- **emission_parameters** — параметри кожного розрахунку: характеристики джерела викиду та метеорологічні умови;
- **simulation_results** — результати моделювання, включно із сіткою концентрацій у форматі JSONB;
- **reports** — метадані згенерованих звітів та шляхи до файлів на сервері.

Кожна таблиця має первинний ключ типу SERIAL, що забезпечує автоматичну генерацію унікальних ідентифікаторів. Зв'язки між таблицями реалізовано через зовнішні ключі з відповідними обмеженнями. Наприклад, поле user_id у таблиці projects посилається на первинний ключ таблиці users, а поле project_id у таблицях emission_parameters та simulation_results — на таблицю projects з каскадним видаленням ON DELETE CASCADE. Це гарантує, що при видаленні проєкту всі пов'язані з ним дані також буде автоматично вилучено.

Для зберігання різних типів даних використовувались такі типи PostgreSQL:

- VARCHAR(n) — для текстових полів обмеженої довжини: імена, електронні адреси, назви речовин, шляхи до файлів;
- TEXT — для довгих текстових описів без обмеження розміру;

- DECIMAL(n, m) — для числових значень з фіксованою точністю: координати, концентрації, фізичні параметри;
- INTEGER та BOOLEAN — для цілочисельних значень і логічних прапорців;
- TIMESTAMP — для фіксації часу створення, оновлення та останнього входу;
- JSONB — для зберігання сіток концентрацій та даних ізоліній, що за своєю природою є складними вкладеними структурами і не вкладаються в плоску реляційну схему.

Таким чином, фізична структура бази даних AeroSpread Pro є логічно впорядкованою і відповідає вимогам як поточної версії системи, так і можливих напрямків її розвитку. Каскадні зв'язки між таблицями, індекси на ключових полях та тригери автоматичного оновлення забезпечують цілісність і актуальність даних без додаткового навантаження на серверну логіку. При необхідності структуру можна розширити новими таблицями або полями без шкоди для стабільності вже існуючих компонентів.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Архітектура системи

На етапі проєктування програмного забезпечення важливо виділити ключові абстракції предметної області — узагальнені моделі реальних об'єктів і процесів системи, що містять їхні основні властивості та обов'язки. На рис. 9 зображено структуру абстракцій предметної області AeroSpread Pro.

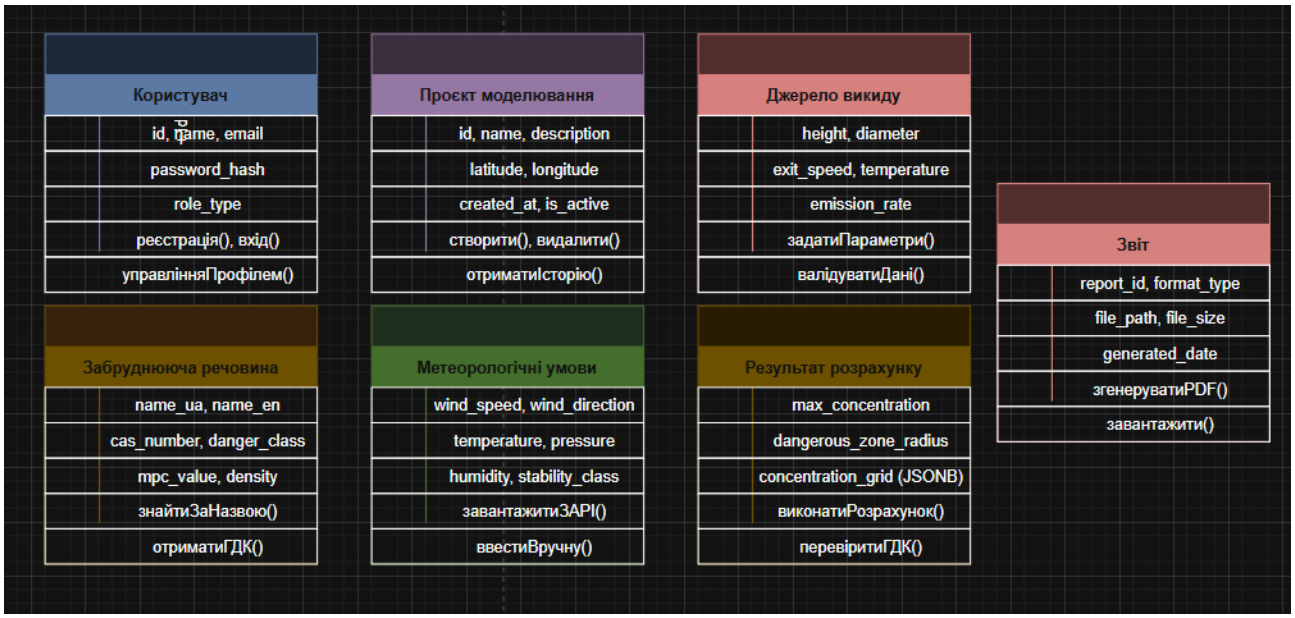


Рис. 9 Абстракції предметної області

До абстракцій системи належать як суб'єкти — користувач — так і об'єкти, що опосередковують взаємодію між ними: проєкт моделювання, джерело викиду, забруднююча речовина, метеорологічні умови, результат розрахунку та звіт. Кожна абстракція визначає характерні властивості та обов'язки, необхідні для побудови функціональної структури системи. Таким чином, виділені абстракції формують логічну основу для подальшого проєктування архітектури програмного забезпечення.

Кожна абстракція визначає характерні властивості та обов’язки, які необхідні для побудови функціональної структури системи. Це дозволяє описати логіку взаємодії між користувачами, забезпечити належну обробку даних, а також встановити зв’язки між об’єктами системи.

3.2 Моделювання структури та взаємодії об’єктів

Визначивши основні абстракції предметної області, наступним кроком стає моделювання того, як ці об'єкти пов'язані між собою і яким чином взаємодіють у процесі роботи системи. Без чіткого розуміння структури та зв'язків між компонентами складно перейти до програмної реалізації — саме тому цьому етапу приділяється окрема увага.

Для побудови моделей використовується мова UML, яка є стандартом у об'єктно-орієнтованому проєктуванні. Вона дозволяє описати систему одразу з двох точок зору: статичної — яка показує структуру та зв'язки між класами, і динамічної — яка відображає поведінку об'єктів під час виконання конкретних сценаріїв. Реалізуємо це за допомогою діаграми класів

Якщо подивитись на систему з точки зору програмної реалізації, то ключовим питанням стає не що зберігається в базі даних, а які об'єкти існують у коді, які властивості вони мають і як між собою взаємодіють. Саме для цього будується діаграма класів — вона є своєрідним планом майбутньої програми, де кожен клас відповідає за конкретну частину логіки.

В AeroSpread Pro виділено шість основних класів. User відповідає за авторизацію та зберігання облікових даних. Project є центральним об'єктом системи — він пов'язує користувача з усіма параметрами моделювання і фактично є точкою входу для запуску розрахунку. EmissionParameters акумулює в собі і характеристики джерела викиду, і метеорологічні умови — саме ці дані передаються в розрахунковий модуль. Substance — довідниковий клас, який надає фізико-хімічні властивості речовини та значення ГДК. SimulationResult зберігає все що повертає математична модель: максимальну концентрацію, радіус небезпечної зони та сітку даних для теплової карти. Нарешті, Report відповідає за формування і збереження підсумкового документу на основі результатів розрахунку.

На рис. 10 представлено діаграму класів системи AeroSpread Pro.

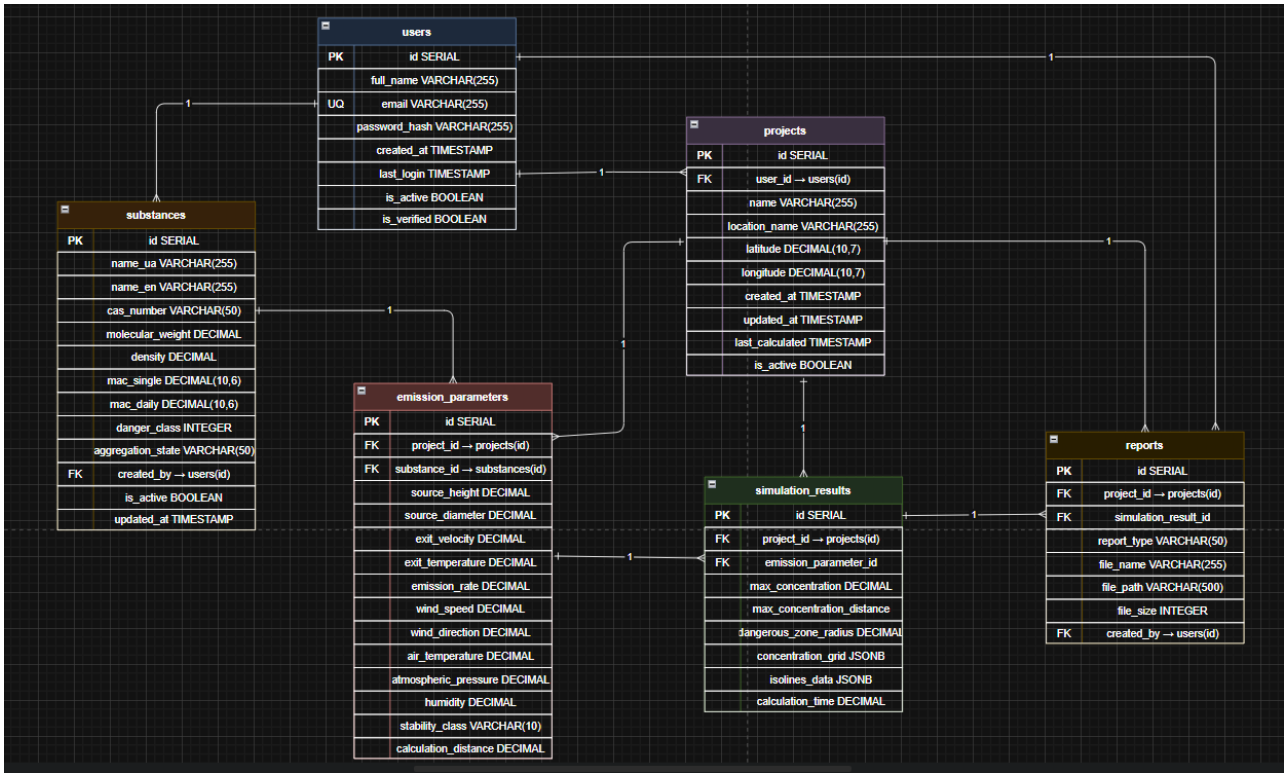


Рис. 10 Діаграма класів із асоціаціями

Діаграма класів охоплює шість основних сутностей: User, Project, EmissionParameters, Substance, SimulationResult та Report. Кожен клас має власний набір атрибутів що зберігають ключову інформацію, а також методи що відповідають за функціональну поведінку системи.

Особливу увагу приділено асоціаціям між класами. Клас User пов'язаний з Project відношенням один до багатьох — один користувач може мати необмежену кількість проєктів моделювання, але кожен проєкт належить лише одному користувачу. Це відображає реальну логіку роботи: кожен зареєстрований фахівець веде власну історію розрахунків незалежно від інших.

Клас Project є центральним вузлом діаграми і пов'язаний одразу з кількома класами. З EmissionParameters встановлено відношення один до багатьох — в межах одного проєкту можна виконувати кілька розрахунків з різними параметрами викиду. EmissionParameters, в свою чергу, асоційований з класом Substance у відношенні багато до одного — одна й та сама речовина може використовуватись у різних розрахунках, тоді як кожен розрахунок прив'язаний до конкретного забруднювача.

Клас SimulationResult пов'язаний з EmissionParameters відношенням один до одного — кожен набір параметрів породжує рівно один результат розрахунку. Водночас SimulationResult має асоціацію з Report у відношенні один до багатьох, оскільки на основі одного результату можна сформувати кілька звітів у різних форматах.

Таким чином, діаграма класів із визначеними асоціаціями формує структурну основу для подальшої програмної реалізації системи. Вона поєднує аналітичний і проєктний підходи, забезпечуючи логічний перехід від моделі даних до конкретних програмних об'єктів і зв'язків між ними.

3.3 Організаційна структура програмного забезпечення

Організаційна структура AeroSpread Pro представлена у вигляді діаграми пакетів, яка відображає логічний поділ системи на окремі функціональні модулі та зв'язки між ними. Діаграма пакетів зображена на мал.11

Вся система представлена єдиним кореневим пакетом aerospread-pro, який охоплює п'ять підпакетів. Кожен з них відповідає за конкретну частину функціоналу і має чітко визначену зону відповідальності. Така структура забезпечує зручність супроводу, незалежність окремих модулів та можливість масштабування без порушення цілісності системи.

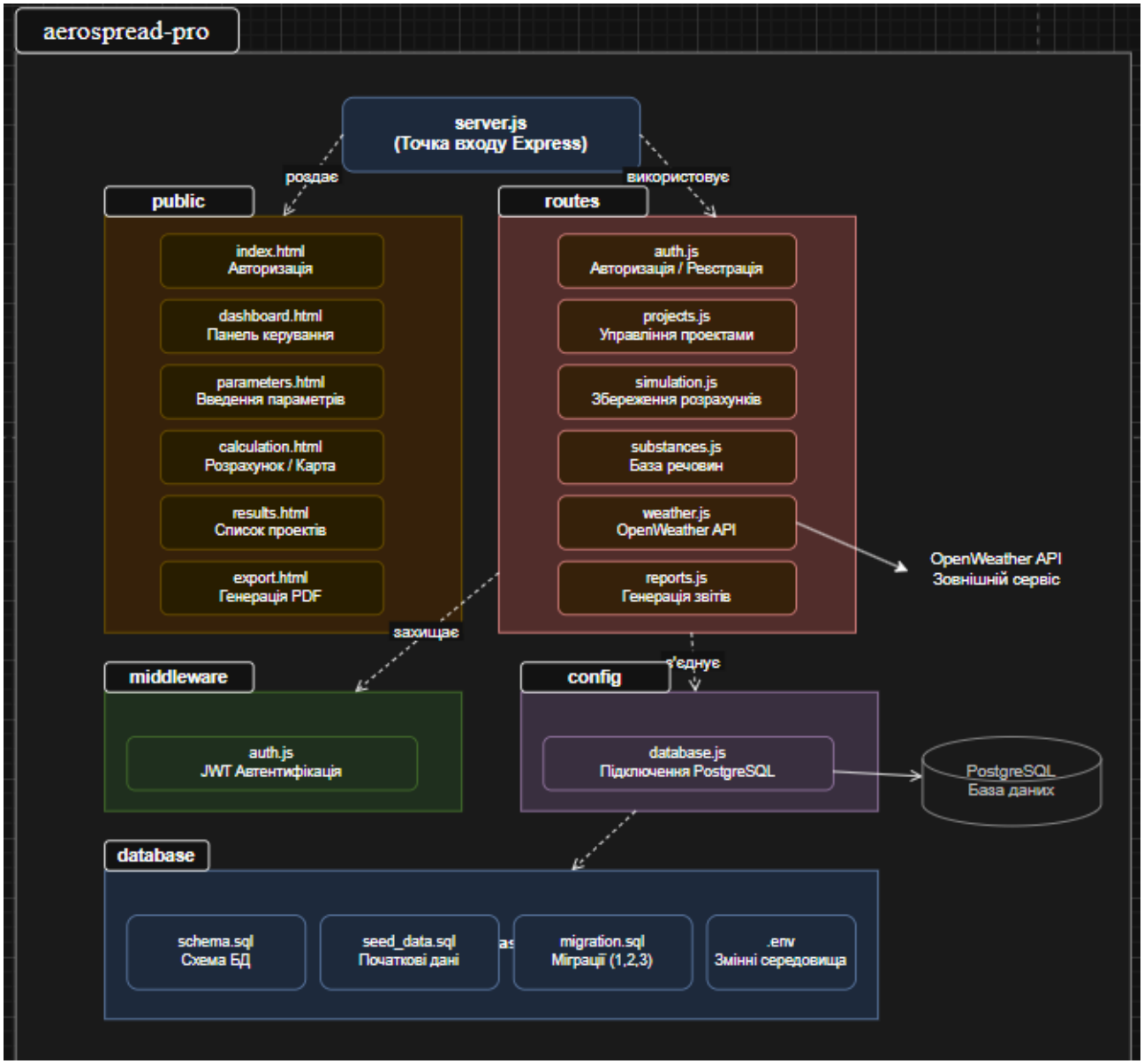


Рис. 11 Діаграма пакетів

Опис пакетів

public — клієнтська частина системи, що містить усі HTML-сторінки інтерфейсу: авторизацію, панель керування, форму введення параметрів викиду, сторінку розрахунку з інтерактивною картою, список збережених проектів та сторінку експорту звітів. Цей пакет є точкою взаємодії користувача з системою і не залежить від інших серверних модулів — він лише надсилає запити до routes.

routes — серверна частина системи, що реалізує REST API. Містить шість маршрутних модулів: авторизацію та реєстрацію користувачів, управління проектами, виконання і збереження розрахунків, роботу з базою забруднюючих

речовин, інтеграцію з OpenWeather API та генерацію звітів. Цей пакет є центральним — через нього проходять усі запити від клієнтської частини.

middleware — допоміжний пакет, що реалізує JWT-автентифікацію. Він підключається до захищених маршрутів пакету routes і перевіряє наявність та коректність токена перед виконанням будь-якого запиту. Пакет залежить від config, оскільки для перевірки токена використовується секретний ключ зі змінних середовища.

config — відповідає за підключення до бази даних PostgreSQL. Містить єдиний модуль database.js, який ініціалізує пул з'єднань і надає його іншим модулям системи. Від цього пакету залежать routes, оскільки всі операції з даними виконуються через нього.

database — містить SQL-файли для розгортування та підтримки бази даних: схему таблиць, початкове наповнення даними про забруднюючі речовини та три файли міграцій, що відображають еволюцію структури бази в процесі розробки. Цей пакет є незалежним і використовується лише при первинному розгортуванні або оновленні системи.

Між пакетами визначено залежності, що відображають реальну логіку взаємодії компонентів. public залежить від routes, оскільки всі дії користувача реалізуються через API. routes залежать від middleware для захисту ендпоінтів і від config для доступу до бази даних. config, в свою чергу, взаємодіє з database для ініціалізації структури. Такий розподіл забезпечує чітку межу між шарами системи і дозволяє вносити зміни в окремі модулі без впливу на решту.

3.4 Компонентна структура системи

Компонентна структура AeroSpread Pro представлена у вигляді діаграми компонентів, яка моделює логічну архітектуру застосунку розподілену на три основні рівні: інтерфейс користувача, бізнес-логіку та рівень доступу до даних. Такий підхід забезпечує високий рівень модульності та зручність супроводу системи. Діаграма компонентів представлена на мал. 12

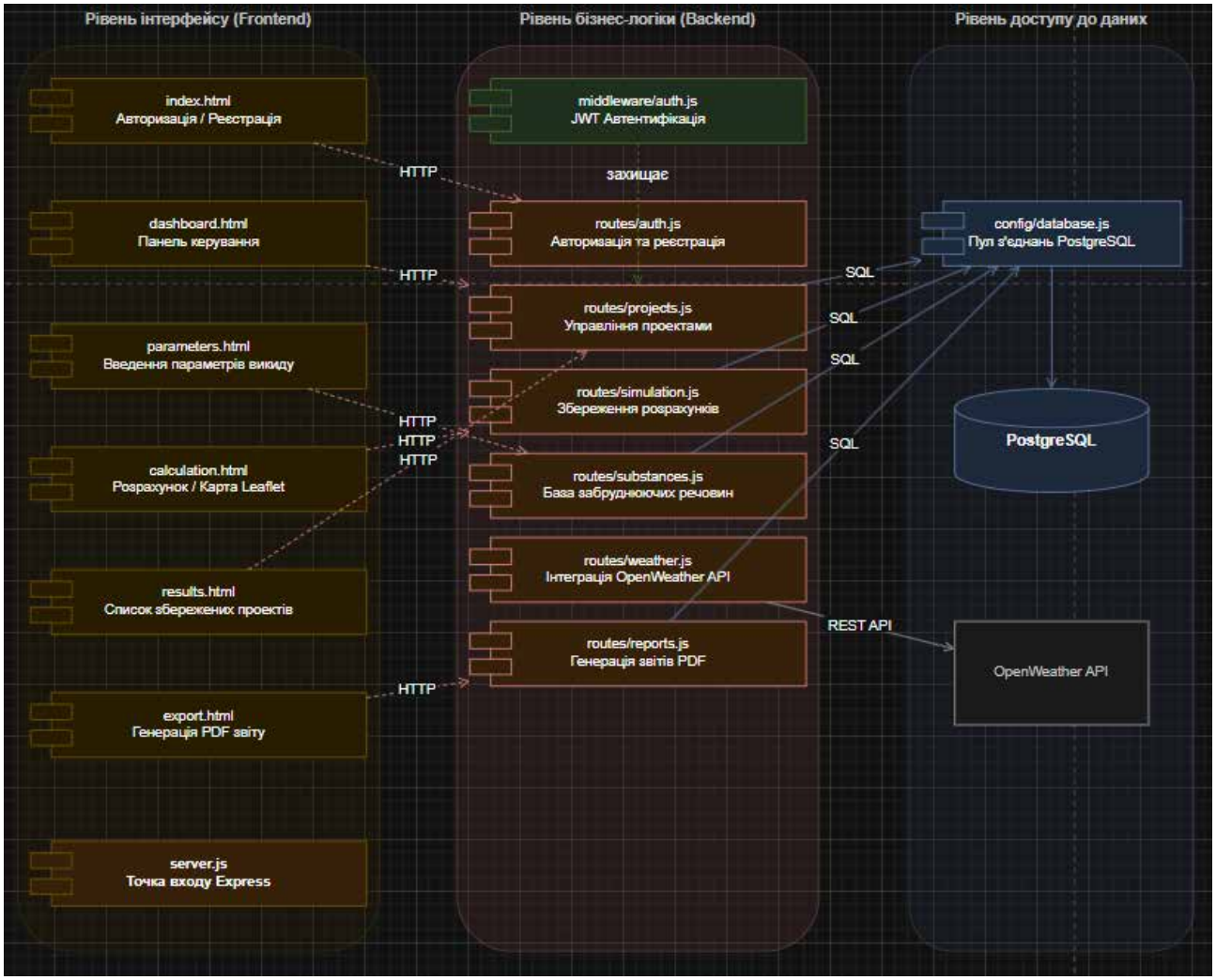


Рис. 12 Діаграма компонентів

Інтерфейс користувача

Клієнтська частина системи реалізована у вигляді HTML-сторінок, кожна з яких відповідає за окремий функціональний екран. index.html забезпечує авторизацію та реєстрацію користувача в системі. dashboard.html є головною панеллю керування після входу — звідси користувач переходить до решти функцій. parameters.html надає форму для введення параметрів джерела викиду та вибору забруднюючої речовини. calculation.html є центральним екраном системи — тут запускається розрахунок і відображаються результати на інтерактивній карті Leaflet. results.html показує список збережених проектів з можливістю їх перегляду та фільтрації. export.html відповідає за формування та завантаження звіту у форматі PDF.

Усі сторінки взаємодіють з серверною частиною через HTTP-запити до відповідних маршрутів API.

Сервісні компоненти (бізнес-логіка)

Серверна частина побудована на основі Express.js і реалізована у вигляді окремих маршрутних модулів. routes/auth.js обробляє реєстрацію та авторизацію, видає JWT-токени і перевіряє облікові дані. routes/projects.js реалізує CRUD-операції над проектами моделювання. routes/simulation.js відповідає за запуск розрахунку та збереження його результатів у базі даних. routes/substances.js надає доступ до довідника забруднюючих речовин з можливістю пошуку. routes/weather.js здійснює запит до OpenWeather API і повертає актуальні метеорологічні дані за координатами. routes/reports.js генерує PDF-звіт на основі збережених результатів розрахунку.

Захист маршрутів забезпечує middleware/auth.js — він перевіряє JWT-токен перед виконанням кожного захищеного запиту.

Рівень доступу до даних

Взаємодія з базою даних централізована через модуль config/database.js, який ініціалізує пул з'єднань з PostgreSQL і надає його всім маршрутним модулям. Це дозволяє уникнути дублювання коду підключення та забезпечує єдину точку управління з'єднаннями. Безпосередньо в базі зберігаються всі сутності системи — користувачі, проекти, параметри викиду, результати розрахунків та звіти.

Окремим зовнішнім компонентом виступає OpenWeather API, до якого звертається routes/weather.js через REST-запит.

Зв'язки між компонентами

Зв'язки між компонентами поділяються на два типи. Залежності вказують на використання одного компонента іншим — HTML-сторінки залежать від відповідних маршрутів API, маршрути залежать від middleware для автентифікації та від config для доступу до бази даних. Реалізації демонструють пряме впровадження функціональності — маршрутні модулі безпосередньо реалізують бізнес-логіку, а config/database.js є прямим посередником між серверною логікою і фізичним зберіганням даних у PostgreSQL.

3.5 Вибір інструментарію для створення ПЗ

У процесі розробки AeroSpread Pro було підібрано інструментарій, який з одного боку відповідає технічним вимогам проєкту, а з іншого — добре знайомий з попередніх курсів і дозволяє зосередитись на логіці системи, а не на освоєнні нових технологій з нуля.

Основним середовищем розробки став Visual Studio Code — легкий, але функціональний редактор з підтримкою JavaScript, вбудованим терміналом, розширеннями для роботи з Node.js та зручною інтеграцією з Git. Він добре підходить для веб-проєктів саме завдяки своїй простоті та швидкості.

Серверна частина написана на Node.js з використанням фреймворку Express.js. Node.js обрано через його асинхронну модель виконання, яка добре справляється з одночасними HTTP-запитами, а Express.js дозволяє швидко організувати маршрутизацію та middleware без зайвого коду. Разом вони утворюють мінімалістичний, але достатній стек для побудови REST API.

Клієнтська частина реалізована на чистому HTML5, CSS3 та JavaScript ES6+ без використання фронтенд-фреймворків. Таке рішення було свідомим — для застосунку з фіксованим набором сторінок і чіткою логікою додаткові залежності лише ускладнили б підтримку. Для відображення інтерактивної карти використано бібліотеку Leaflet з тайлами OpenStreetMap, яка дозволяє візуалізувати результати моделювання безпосередньо на географічній підкладці.

Зберігання даних реалізовано засобами PostgreSQL. Для адміністрування та розробки структури бази використовувався pgAdmin — офіційний клієнт PostgreSQL, що дозволяє керувати таблицями, виконувати запити та відстежувати стан бази. Підключення до PostgreSQL з серверної частини здійснюється через бібліотеку pg для Node.js.

Для реалізації функціоналу експорту звітів підключено бібліотеку генерації PDF, що дозволяє формувати документи з таблицями та результатами розрахунків безпосередньо на сервері без потреби у зовнішніх сервісах.

Отримання актуальних метеорологічних даних реалізовано через OpenWeather API. Інтеграція здійснюється через стандартний HTTP-запит з

серверної частини — це дозволяє уникнути проблем з CORS та приховати API-ключ від клієнта.

Контроль версій вівся засобами Git з використанням репозиторію на GitHub, що забезпечувало збереження історії змін та можливість відкату до попередніх версій у разі потреби.

Таким чином, обраний стек — Node.js, Express, PostgreSQL, Leaflet — є збалансованим рішенням для веб-застосунку такого типу. Він забезпечує достатню продуктивність, просту розгортку на хостингу та зручність подальшого розширення функціональності.

3.6 Алгоритмізація та програмування програмних модулів

Алгоритмізація функціональних процесів системи є важливою складовою програмної реалізації, що забезпечує послідовність, точність та ефективність виконання основних операцій. На цьому етапі визначаються ключові дії, що виконуються під час взаємодії користувача з інтерфейсом системи, обробки введених даних, їх перевірки та збереження у базу даних. Для ілюстрації буде наведено ключовий алгоритм системи.

Центральним алгоритмом системи є розрахунок концентрацій забруднюючої речовини в атмосфері. Після завантаження сторінки calculation.html система зчитує параметри з localStorage або завантажує збережений проєкт з сервера. За відсутності параметрів користувач перенаправляється назад на форму введення.

Першим кроком є виклик функції getSigmaFunctions(), яка на основі класу стійкості атмосфери за Паскіль-Гіффордом (A–F) визначає коефіцієнти Брідоса для розрахунку стандартних відхилень σy та σz. Далі запускається цикл по відстанях від 10 до 5000 м з кроком 10 м. На кожній ітерації викликається функція gaussianPlume(), яка обчислює концентрацію за формулою:

$$C = Q / (2\pi \cdot u \cdot \sigma y \cdot \sigma z) \cdot \exp(-0.5 \cdot (H/\sigma z)^2) \cdot 1000 \quad (1)$$

де Q — потужність викиду, H — висота джерела, u — швидкість вітру. В процесі циклу фіксуються максимальна концентрація, відстань до неї та радіус небезпечної зони де концентрація перевищує ГДК.

Після завершення циклу результати відображаються на сторінці, викликається drawIsolines() для побудови зон забруднення на карті у вигляді шести кольорових рівнів відносно ГДК, та fillTable() для заповнення таблиці значень на восьми контрольних відстанях. Блок-схему алгоритму наведено на рис.13

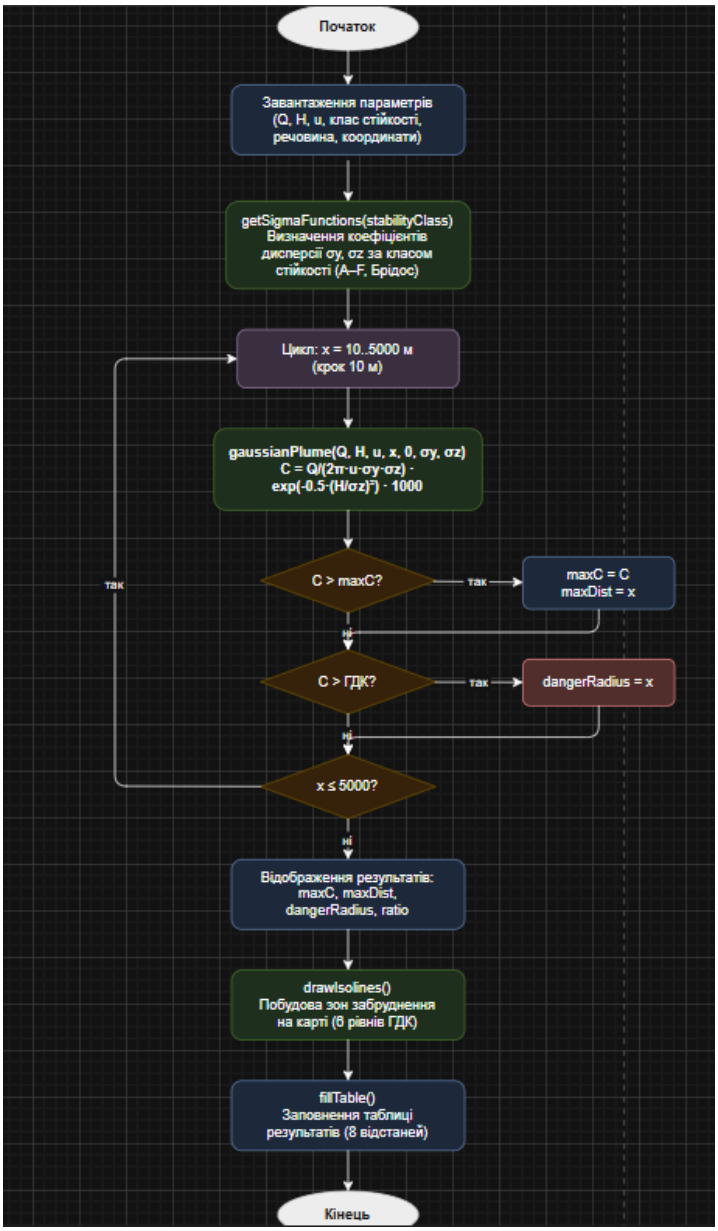


Рис. 13 Блок-схема алгоритму розрахунку розповсюдження за моделлю Гаусівського шлейфу

Далі розглянемо основні фрагменти коду, які реалізують цей алгоритм.

Функція getSigmaFunctions(), наведена на рис. 14, визначає коефіцієнти дисперсії σy та σz залежно від класу стійкості атмосфери за класифікацією Паскіля–Гіффорда. Для кожного з шести класів (A–F) задано пару коефіцієнтів за методом Брідоса. Функція повертає два лямбда-вирази, які обчислюють σy(x) та σz(x) для довільної відстані x за степеневим законом.

```
function getSigmaFunctions(sc) {
    const c = {
        A: {ya:0.22, yb:0.894, za:0.20, zb:0.894},
        B: {ya:0.16, yb:0.894, za:0.12, zb:0.894},
        C: {ya:0.11, yb:0.894, za:0.08, zb:0.894},
        D: {ya:0.08, yb:0.894, za:0.06, zb:0.894},
        E: {ya:0.06, yb:0.894, za:0.03, zb:0.894},
        F: {ya:0.04, yb:0.894, za:0.016, zb:0.894}
    }[sc] || c.D;
    return {
        y: x => c.ya * x**c.yb,
        z: x => c.za * x**c.zb
    };
}
```

Рис. 14 Функція getSigmaFunctions()

Функція gaussianPlume() фрагмент якої наведено на рис. 15, реалізує безпосередній математичний розрахунок концентрації забруднюючої речовини в заданій точці простору. Вона приймає потужність викиду Q, висоту джерела H, швидкість вітру u, координати точки x та y відносно джерела, а також значення стандартних відхилень σy і σz. За умови $x \leq 0$ функція повертає нуль, оскільки розповсюдження відбувається лише за напрямком вітру.

```
function gaussianPlume(Q, H, u, x, y, sy, sz) {
    if (x <= 0) return 0;
    return (Q / (2 * Math.PI * u * sy * sz)) *
           Math.exp(-0.5 * (y/sy)**2) *
           Math.exp(-0.5 * (H/sz)**2) * 1000;
}
```

Рис. 15 Функція gaussianPlume()

Функція calculate(), фрагмент якої наведено на рис. 16, запускає основний цикл обчислень. В циклі від 10 до 5000 м з кроком 10 м на кожній ітерації викликається gaussianPlume() для точки на осі шлейфу (y=0). Паралельно відстежується максимальна концентрація та відповідна їй відстань, а також визначається радіус небезпечної зони — остання відстань де концентрація перевищує значення ГДК.

```
function calculate() {
    setTimeout(() => {
        const Q = parameters.emissionRate;
        const H = parameters.sourceHeight;
        const u = parameters.windSpeed;
        const mac = parseFloat(parameters.substance.mac_single) || 0.1;

        const sigma = getSigmaFunctions(parameters.stabilityClass);
        let maxC = 0, maxDist = 0, dangerRadius = 0;

        for (let x = 10; x <= 5000; x += 10) {
            const C = gaussianPlume(Q, H, u, x, 0, sigma.y(x), sigma.z(x));
            if (C > maxC) { maxC = C; maxDist = x; }
            if (C > mac && x > dangerRadius) dangerRadius = x;
        }

        results = {maxC, maxDist, dangerRadius, ratio: maxC / mac};
        display();
        drawIsolines(Q, H, u, sigma, mac);
        fillTable(Q, H, u, sigma, mac);
    }, 1500);
}
```

Рис. 16 Функція calculate()

Функція drawIsolines(), фрагмент якої наведено на рис. 17, відповідає за побудову зон забруднення на інтерактивній карті. Для кожного з шести порогових рівнів відносно ГДК система обходить сітку точок і знаходить контур зони де концентрація перевищує відповідний поріг. Контур повертається відповідно до напрямку вітру через тригонометричне перетворення координат, після чого відображається на карті Leaflet у вигляді кольорового полігону.

```
function drawIsolines(Q, H, u, sigma, mac) {
    const windDirectionDegrees = getWindDirectionDegrees(parameters.windDirection);
    const windAngle = (windDirectionDegrees * Math.PI) / 180;

    const levels = [
        {threshold: 10*mac, color: '#ff0000', opacity: 0.6, label: '> 10 ГДК'},
        {threshold: 5*mac,  color: '#ff6b00', opacity: 0.5, label: '5-10 ГДК'},
        {threshold: mac,    color: '#ffab00', opacity: 0.45, label: '1-5 ГДК'},
        {threshold: 0.5*mac,color: '#ffd700', opacity: 0.4, label: '0.5-1 ГДК'},
        {threshold: 0.1*mac,color: '#00d4ff', opacity: 0.35, label: '0.1-0.5 ГДК'},
        {threshold: 0.01*mac,color: '#64ffda', opacity: 0.3, label: '< 0.1 ГДК'}
    ];

    reachableLevels.reverse().forEach(level => {
        const points = [];
        for (let y = -maxDistance; y <= maxDistance; y += 30) {
            let maxX = 0;
            for (let x = 10; x <= maxDistance; x += 20) {
                const C = gaussianPlume(Q, H, u, x, y, sigma.y(x), sigma.z(x));
                if (C >= level.threshold) maxX = x;
            }
            if (maxX > 0) {
                const rotatedX = maxX * Math.cos(windAngle) - y * Math.sin(windAngle
                const rotatedY = maxX * Math.sin(windAngle) + y * Math.cos(windAngle
                const coord = offset(parameters.latitude, parameters.longitude, rota
                points.push([coord.lat, coord.lng]);
            }
        }
        if (points.length >= 3) {
            L.polygon(points, {
                color: level.color,
                fillColor: level.color,
                fillOpacity: level.opacity,
                weight: 2
            }).addTo(map).bindPopup(`<b>${level.label}</b>`);
        }
    });
}
```

Рис. 17 Функція drawIsolines()

4 РЕКОМЕНДАЦІЇ ЩОДО ВПРОВАДЖЕННЯ ТА ЕКСПЛУАТАЦІЇ СИСТЕМИ

4.1 Тестування системи

Ретельне тестування є невід'ємною частиною розробки будь-якої програмної системи — саме воно дозволяє виявити помилки до того, як система потрапить до кінцевого користувача, і переконатись що реалізований функціонал відповідає поставленим вимогам. В ході тестування AeroSpread Pro перевірялись як окремі компоненти, так і система в цілому, включаючи взаємодію між фронтендом, бекендом і базою даних.

Було проведено кілька видів тестування.

Модульне тестування охоплювало перевірку окремих функцій системи — валідацію форм введення параметрів, коректність математичних обчислень у функціях gaussianPlume() та getSigmaFunctions(), а також окремі ендпоінти REST API. Основна мета — виявити логічні помилки на рівні конкретних програмних блоків ще до їх інтеграції.

Інтеграційне тестування перевіряло взаємодію між компонентами системи: передачу параметрів з форми на сервер, збереження результатів у PostgreSQL, коректність отримання метеоданих з OpenWeather API та їх підстановку в розрахунок.

Тестування зручності використання проводилось шляхом імітації типових сценаріїв роботи — від реєстрації до отримання готового PDF-звіту. Оцінювалась логічність навігації між сторінками, зрозумілість форм введення та читабельність результатів на карті.

Фінальне тестування проводилось без звернення до внутрішнього коду — перевірялась поведінка системи виключно з точки зору кінцевого користувача: задавались вхідні дані, фіксувались фактичні результати і порівнювались з очікуваними.

Наведений нижче приклад імітує типову послідовність дій користувача в системі без знання її внутрішньої реалізації.

Реєстрація у системі .Користувач відкриває сторінку авторизації та вводить облікові дані. На рис. 18 показано відповідну форму реєстрації. Після успішної перевірки токен зберігається в localStorage і система автоматично перенаправляє користувача на dashboard.html.

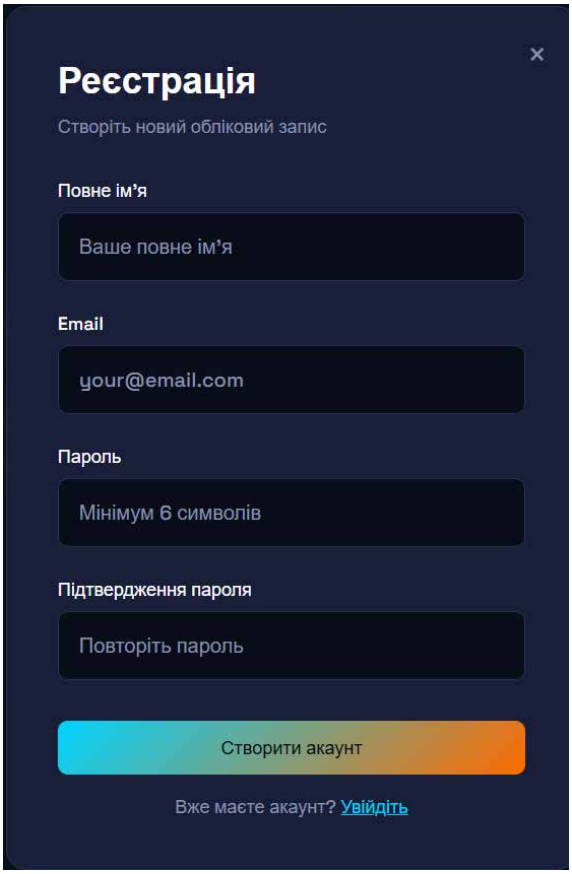


Рис. 18 Форма реєстрації

Головне панель. Після входу користувач потрапляє на головну панель керування. На екрані відображаються доступні модулі та можливість перейти до інших сторінок.

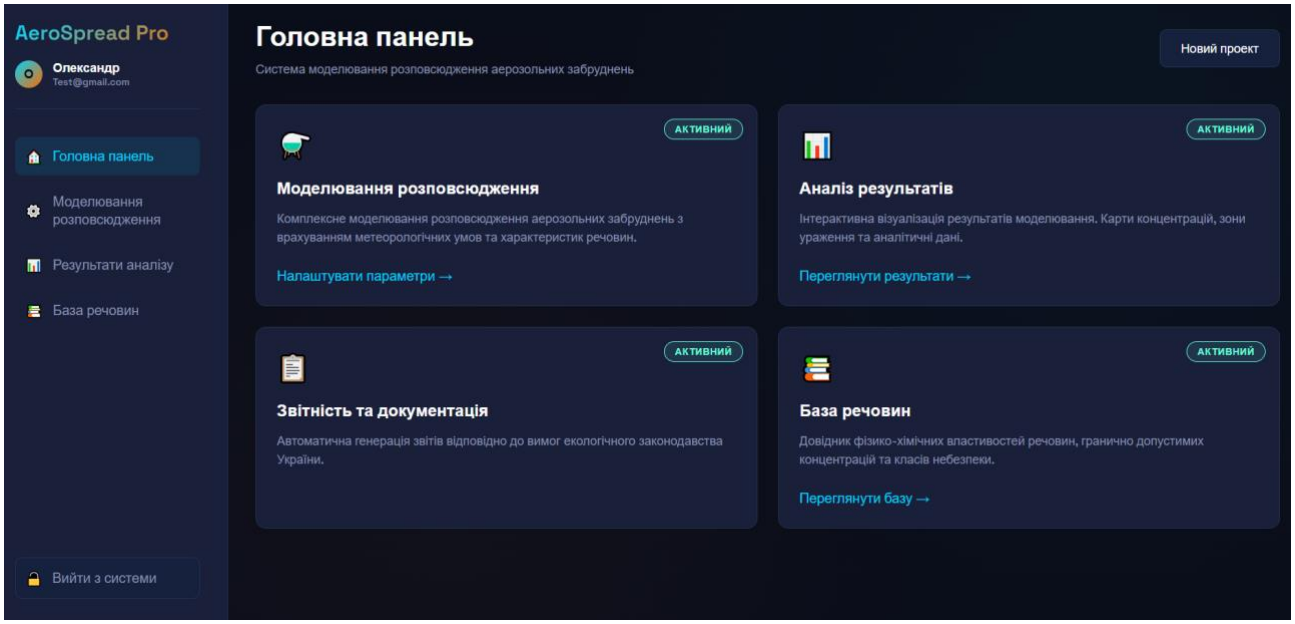


Рис. 19 Головна панель

Для запуску розрахунку користувач переходить на сторінку моделювання розповсюдження, де заповнює форму з параметрами джерела викиду — висотою джерела, Інтенсивність викиду, та обирає забруднюючу речовину з бази даних. Метеорологічні дані можна ввести вручну або завантажити автоматично з OpenWeather API за координатами об'єкта.

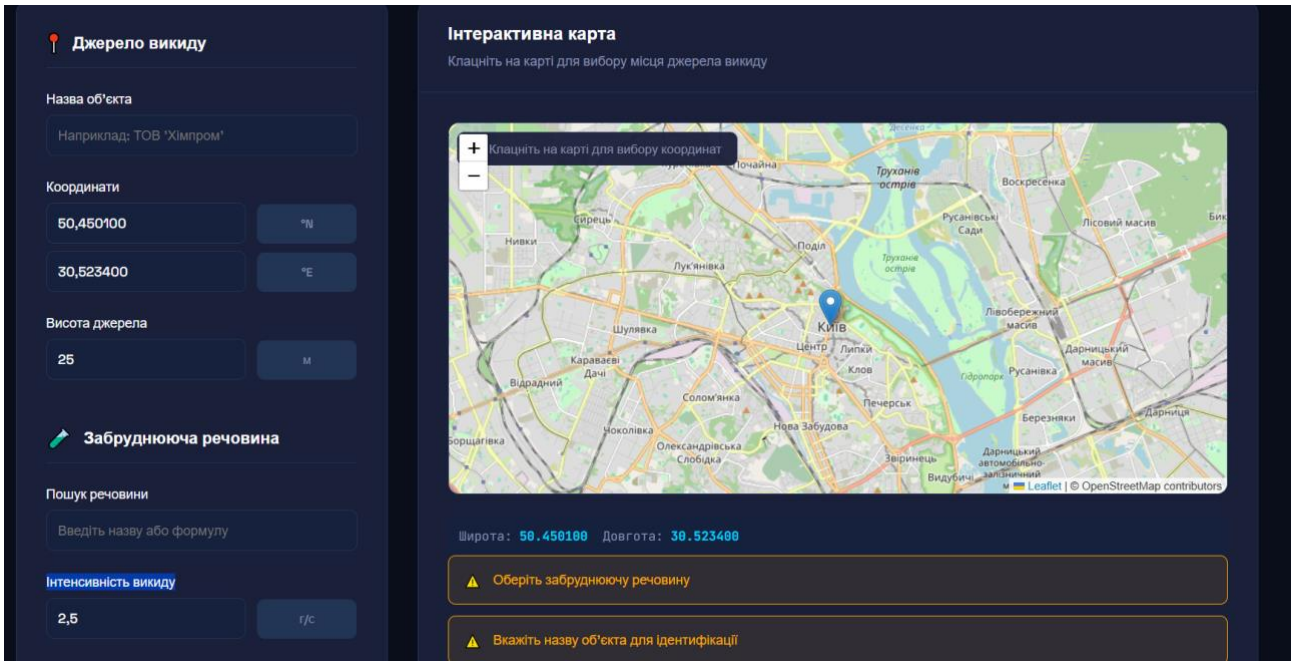


Рис. 20 Моделювання розповсюдження

Після заповнення всіх полів користувач натискає кнопку розрахунку і система перенаправляє його на сторінку розрахунку, де виконується моделювання та відображаються результати на інтерактивній карті з кольоровими зонами концентрацій.

Рис. 21 Результат моделювання на інтерактивній карті

Детальні результати

Відстань, м	Концентрація, мг/м³	% від ГДК	Оцінка
100	0.034640	17.3%	Безпечно
250	0.697974	349.0%	Небезпечно
500	0.447672	223.8%	Небезпечно
1000	0.163202	81.6%	Безпечно
1500	0.082965	41.5%	Безпечно
2000	0.050518	25.3%	Безпечно
3000	0.024813	12.4%	Безпечно
5000	0.010033	5.0%	Безпечно

Рис. 22 Таблиця концентрації

Після завершення розрахунку користувач може, зберегти проект та сформувати звіт, натиснувши кнопку «Звіт» на сторінці результатів. Система перенаправляє на сторінку звітів, де відображаються всі параметри розрахунку, результати моделювання та таблиця концентрацій на різних відстанях від

джерела. Звіт можна завантажити у форматі PDF натисканням відповідної кнопки. Згенерований документ містить назву проєкту, координати об'єкта, характеристики забруднюючої речовини, метеорологічні умови та основні показники розрахунку — максимальну концентрацію, відстань до неї і радіус небезпечної зони.

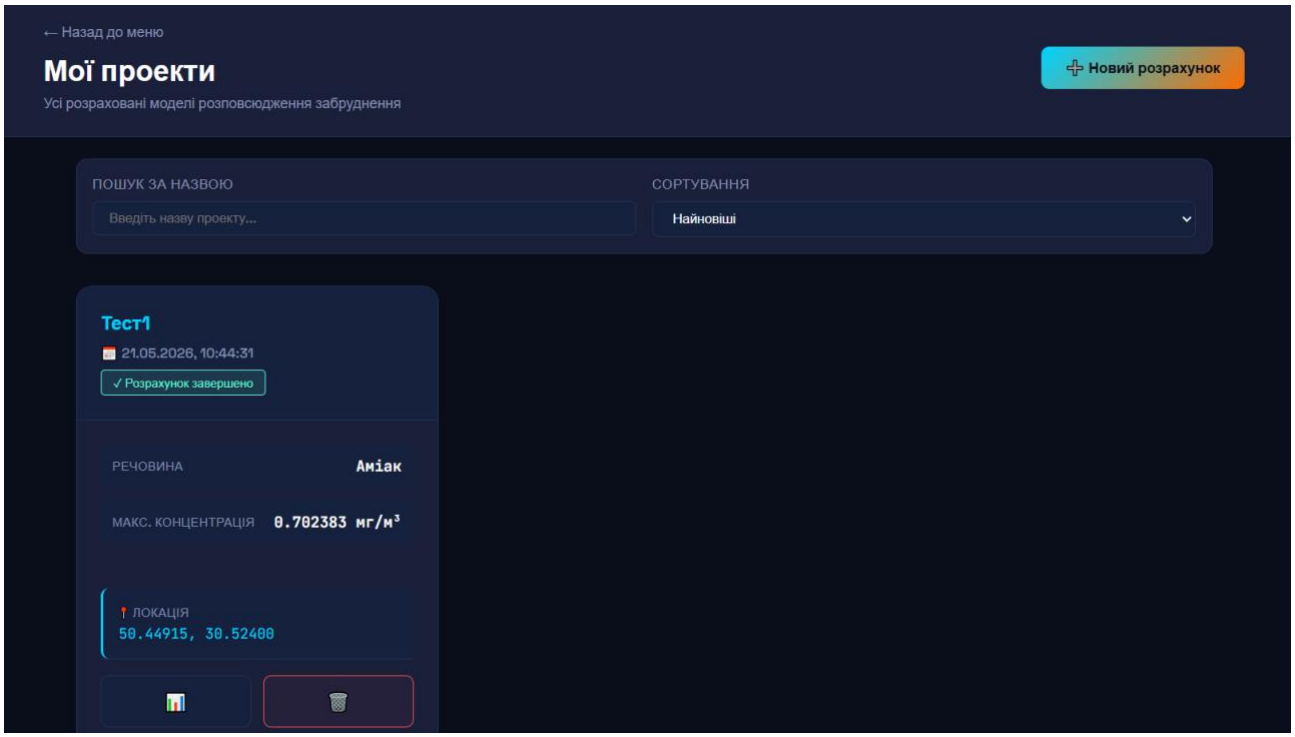


Рис. 23 Збережені проєкти

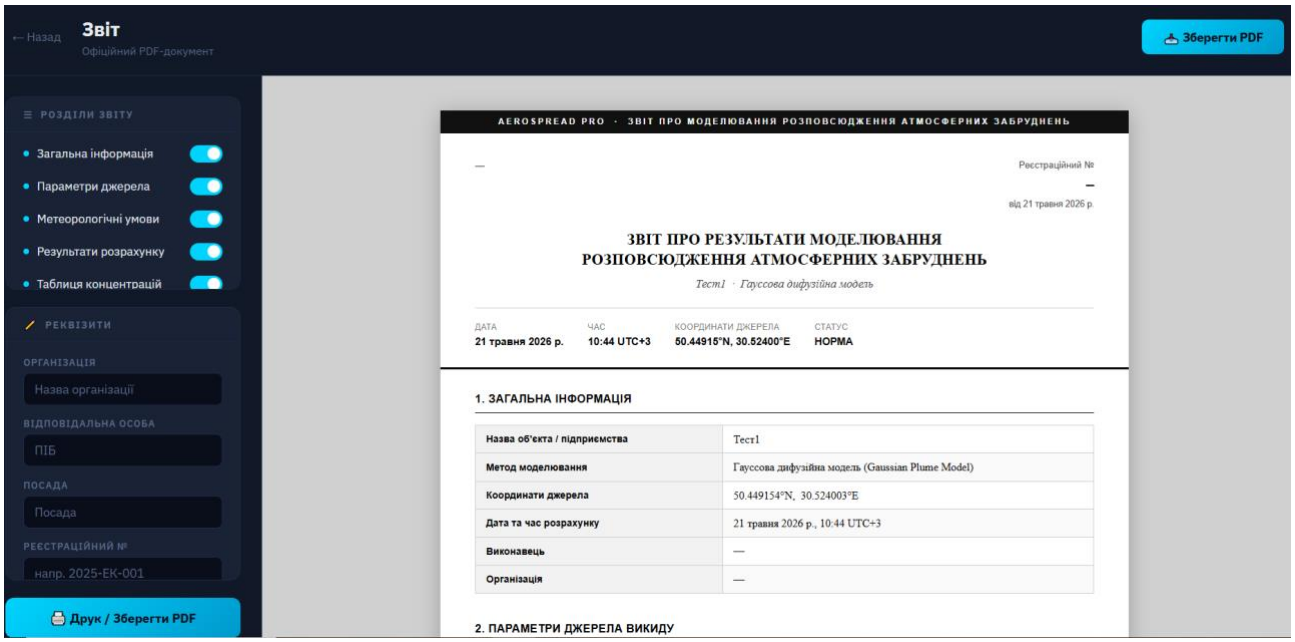
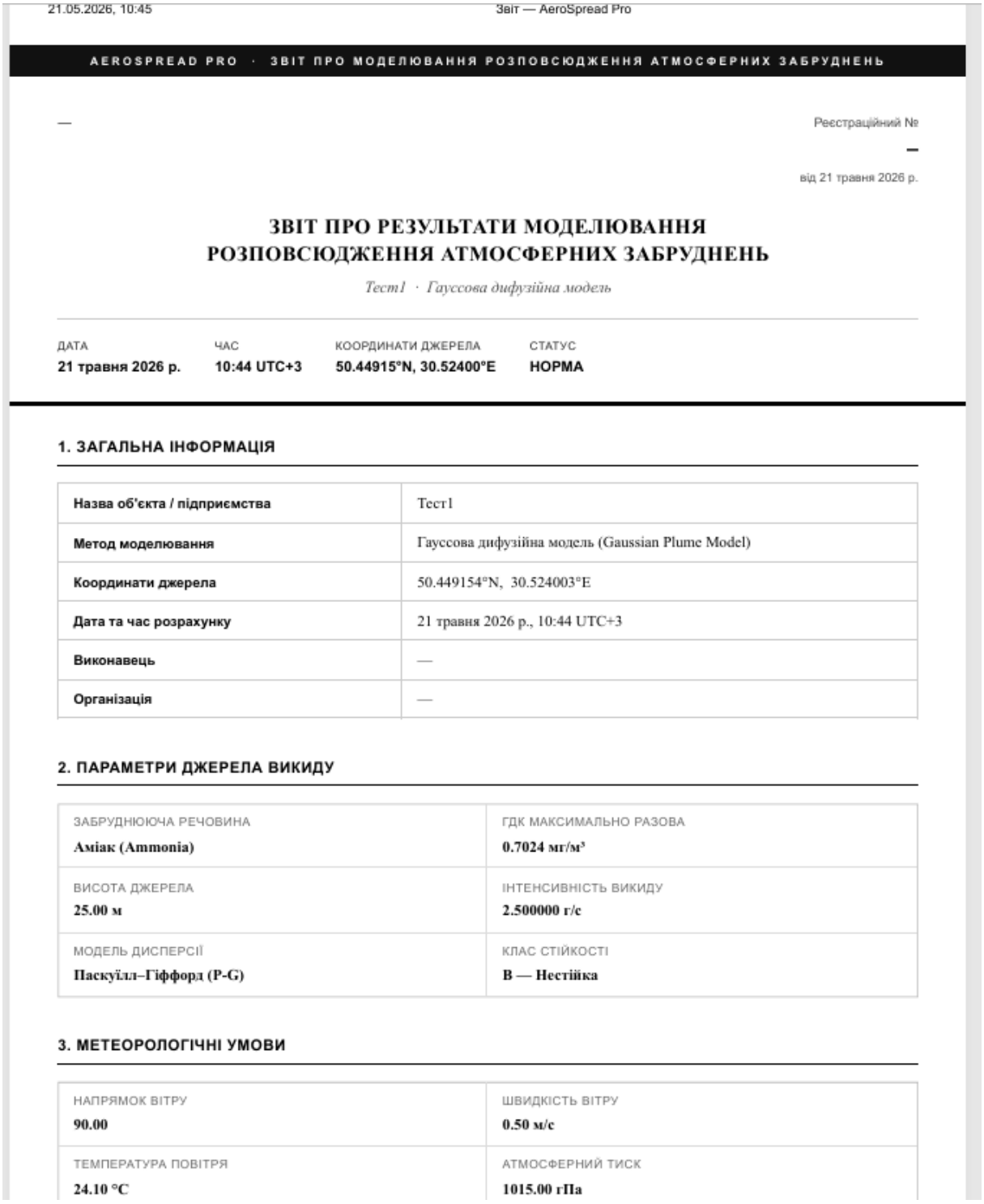


Рис. 24 Сторінка створення звіту



21.05.2026, 10:45 Звіт — AeroSpread Pro

AEROSPREAD PRO · ЗВІТ ПРО МОДЕЛЮВАННЯ РОЗПОВСЮДЖЕННЯ АТМОСФЕРНИХ ЗАБРУДНЕНЬ

—

Реєстраційний №

–

від 21 травня 2026 р.

ЗВІТ ПРО РЕЗУЛЬТАТИ МОДЕЛЮВАННЯ
РОЗПОВСЮДЖЕННЯ АТМОСФЕРНИХ ЗАБРУДНЕНЬ

Тест1 · Гауссова дифузійна модель

ДАТА	ЧАС	КООРДИНАТИ ДЖЕРЕЛА	СТАТУС
21 травня 2026 р.	10:44 UTC+3	50.44915°N, 30.52400°E	НОРМА

1. ЗАГАЛЬНА ІНФОРМАЦІЯ

Назва об'єкта / підприємства	Тест1
Метод моделювання	Гауссова дифузійна модель (Gaussian Plume Model)
Координати джерела	50.449154°N, 30.524003°E
Дата та час розрахунку	21 травня 2026 р., 10:44 UTC+3
Виконавець	—
Організація	—

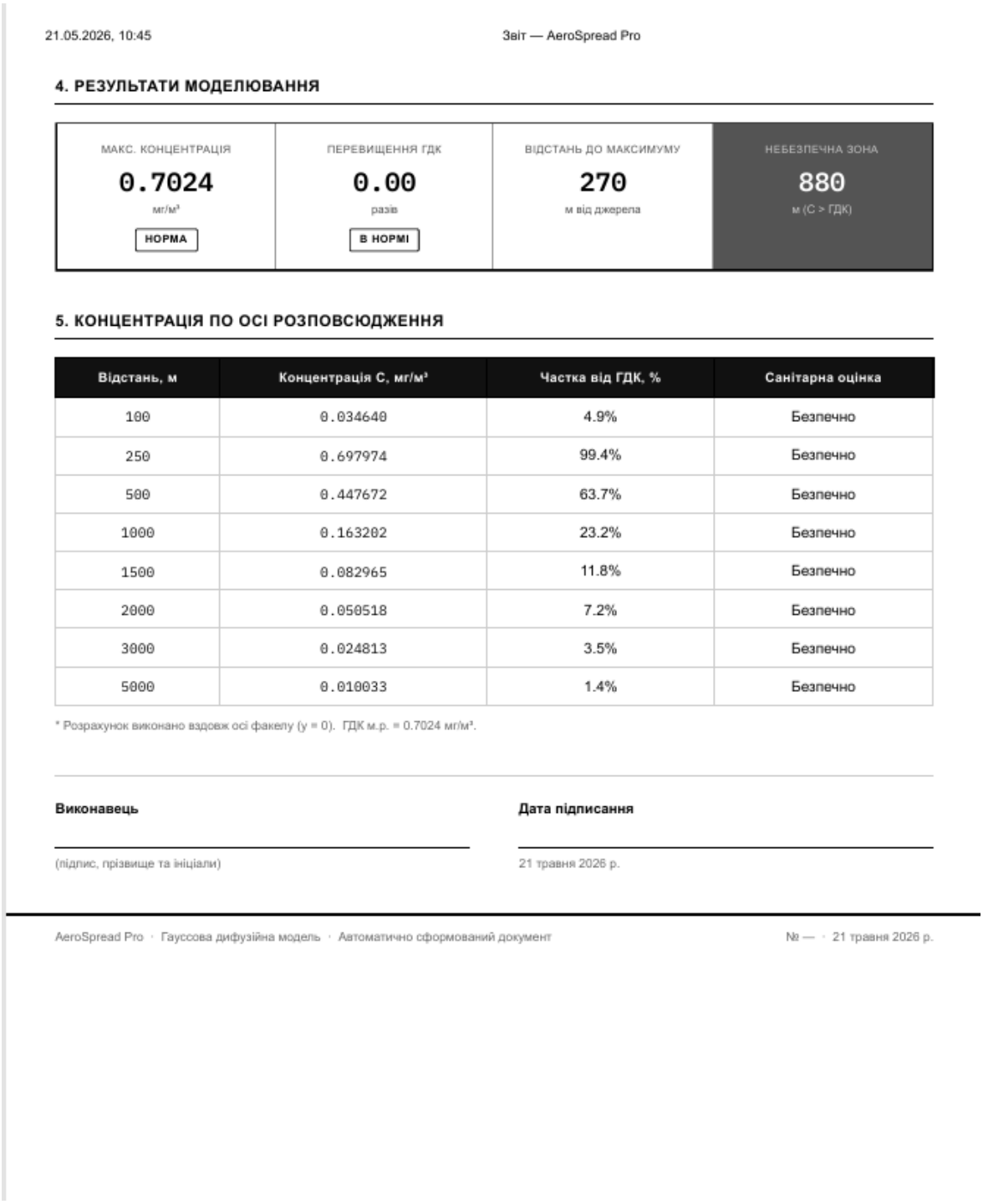
2. ПАРАМЕТРИ ДЖЕРЕЛА ВИКИДУ

ЗАБРУДНЮЮЧА РЕЧОВИНА **Аміак (Ammonia)**	ГДК МАКСИМАЛЬНО РАЗОВА **0.7024 мг/м³**
ВИСОТА ДЖЕРЕЛА **25.00 м**	ІНТЕНСИВНІСТЬ ВИКИДУ **2.500000 г/с**
МОДЕЛЬ ДИСПЕРСІЇ **Паскуїлл–Гіффорд (P-G)**	КЛАС СТІЙКОСТІ **B — Нестійка**

3. МЕТЕОРОЛОГІЧНІ УМОВИ

НАПРЯМОК ВІТРУ **90.00**	ШВИДКІСТЬ ВІТРУ **0.50 м/с**
ТЕМПЕРАТУРА ПОВІТРЯ **24.10 °C**	АТМОСФЕРНИЙ ТИСК **1015.00 гПа**

Рис. 25 Сформований звіт, перший аркуш



21.05.2026, 10:45 Звіт — AeroSpread Pro

4. РЕЗУЛЬТАТИ МОДЕЛЮВАННЯ

МАКС. КОНЦЕНТРАЦІЯ	ПЕРЕВИЩЕННЯ ГДК	ВІДСТАНЬ ДО МАКСИМУМУ	НЕБЕЗПЕЧНА ЗОНА
0.7024 мг/м³ НОРМА	0.00 разів В НОРМІ	270 м від джерела	880 м (C > ГДК)

5. КОНЦЕНТРАЦІЯ ПО ОСІ РОЗПОВСЮДЖЕННЯ

Відстань, м	Концентрація C, мг/м³	Частка від ГДК, %	Санітарна оцінка
100	0.034640	4.9%	Безпечно
250	0.697974	99.4%	Безпечно
500	0.447672	63.7%	Безпечно
1000	0.163202	23.2%	Безпечно
1500	0.082965	11.8%	Безпечно
2000	0.050518	7.2%	Безпечно
3000	0.024813	3.5%	Безпечно
5000	0.010033	1.4%	Безпечно

* Розрахунок виконано вздовж осі факелу (y = 0). ГДК м.р. = 0.7024 мг/м³.

Виконавець _____ (підпис, прізвище та ініціали)

Дата підписання _____ 21 травня 2026 р.

AeroSpread Pro · Гауссова дифузійна модель · Автоматично сформований документ № — · 21 травня 2026 р.

Рис. 26 Сформований звіт, другий аркуш

Результат тестування:

За результатами проведеного тестування система відпрацювала всі кроки сценарію без зауважень. Всі функції виконались коректно на кожному етапі. Введені дані оброблялись правильно, а отримані результати відповідали

очікуваним значенням, що підтверджує відповідність реалізованого функціоналу поставленим вимогам.

4.2 Вимоги до апаратного та програмного забезпечення

Для коректного функціонування AeroSpread Pro необхідно дотриматись певних вимог до апаратного та програмного забезпечення як на стороні сервера, так і на стороні користувача. Оскільки система реалізована як веб-застосунок, клієнтська частина не потребує встановлення — достатньо браузера та підключення до інтернету.

Система побудована за архітектурою клієнт-сервер: браузер користувача взаємодіє з Express.js сервером через HTTP-запити, сервер у свою чергу звертається до PostgreSQL для зберігання даних та до OpenWeather API для отримання метеорологічних даних.

Вимоги до апаратного забезпечення

Серверна частина:

- процесор не нижче двоядерного з тактовою частотою від 1.5 ГГц;
- оперативна пам'ять — не менше 1 ГБ;
- вільне місце на диску — не менше 500 МБ для застосунку та бази даних;
- стабільне підключення до інтернету для інтеграції з OpenWeather API.

Клієнтська частина:

- будь-який пристрій з доступом до інтернету та сучасним браузером;
- мінімальна роздільна здатність екрану 1280×720 для коректного відображення карти.

Вимоги до програмного забезпечення

Серверна частина:

- операційна система Linux або Windows Server;
- Node.js версії 14 або новіша;
- PostgreSQL версії 12 або новіша;
- менеджер процесів PM2 для стабільної роботи сервера у продакшені;
- Nginx як reverse proxy для обробки вхідних запитів.

Клієнтська частина:

- будь-який сучасний браузер з підтримкою JavaScript ES6+ — Chrome, Firefox, Safari, Edge;
- підключення до інтернету для завантаження тайлів карти OpenStreetMap та метеоданих.

4.3 Склад інсталяційного пакету

Інсталяційний пакет містить усі необхідні компоненти для розгортання системи на сервері.

До складу пакету входять:

- вихідний код застосунку з усіма залежностями, визначеними у файлі package.json;
- файл .env з параметрами підключення до бази даних, секретним ключем JWT та ключем OpenWeather API;
- SQL-скрипти для ініціалізації структури бази даних — schema.sql, seed_data.sql та три файли міграцій;
- документація з інструкціями розгортання та налаштування системи.

Розгортання системи виконується у кілька кроків: встановлення залежностей командою npm install, налаштування змінних середовища у файлі .env, ініціалізація бази даних виконанням SQL-скриптів та запуск сервера через PM2. Після цього система готова до роботи і доступна через браузер за адресою сервера.

ВИСНОВКИ

У результаті виконання бакалаврської кваліфікаційної роботи на тему «Програмне забезпечення оцінки розповсюдження забруднюючої аерозольної хмари в антропогенному середовищі» пройдено повний цикл розробки — від аналізу предметної області до розгортання готової системи на хостингу.

На етапі аналізу досліджено процеси розповсюдження забруднюючих речовин в атмосфері, розглянуто існуючі програмні рішення та визначено їхні обмеження. На основі цього сформульовано технічне завдання і обрано математичну модель Гаусівського шлейфу як основу для розрахунків — вона забезпечує достатню точність для практичних задач і водночас реалізується без надмірних обчислювальних ресурсів.

Спроєктовано базу даних PostgreSQL з шістьма основними таблицями, реалізовано тригери автоматичного оновлення та індекси для оптимізації запитів. Логічна модель приведена до третьої нормальної форми, що виключає надлишковість і забезпечує цілісність даних.

Серверна частина побудована на Node.js з Express.js і реалізує REST API з шістьма маршрутними модулями. Клієнтська частина виконана на чистому HTML5/CSS3/JavaScript без фреймворків — для відображення результатів на карті використано Leaflet з тайлами OpenStreetMap. Система інтегрована з OpenWeather API для автоматичного отримання метеоданих.

Реалізований функціонал охоплює повний робочий цикл: реєстрацію та авторизацію з JWT-токенами, введення параметрів викиду, розрахунок концентрацій на відстанях до 5000 м, візуалізацію шести зон забруднення на інтерактивній карті, збереження проектів у базі даних та експорт звітів у форматі PDF.

Система пройшла комплексне тестування — модульне, інтеграційне та за методом чорної скриньки. Всі перевірені сценарії відпрацювали коректно. Застосунок розгорнуто на віддаленому сервері і доступний для використання.

Таким чином, усі поставлені завдання виконано в повному обсязі. AeroSpread Pro є практичним інструментом для оцінки атмосферного забруднення, доступним без встановлення і придатним для використання в освітніх установах, дослідницьких центрах та екологічних службах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Pasquill F., Smith F. B. Atmospheric Diffusion: Studies of Dispersion and
2. Wark C., Warner C. F., Davis W. T. Air Pollution: Its Origin and Control. Pearson, 2020. 576 p.
3. Баженов В. А., Гурєєва І. В. Охорона навколишнього середовища. Київ: Каравела, 2018. 352 с.
4. Запольський А. К., Салюк А. І. Основи екології. Київ: Вища школа, 2019. 399 с.
5. Node.js Foundation. Node.js Documentation. URL: https://nodejs.org/en/docs/ (дата звернення: 10.04.2026).
6. The Express.js Team. Express.js API Reference. URL: https://expressjs.com/en/api.html (дата звернення: 10.04.2026).
7. PostgreSQL Global Development Group. PostgreSQL 15 Documentation. URL: https://www.postgresql.org/docs/ (дата звернення: 10.04.2026).
8. Leaflet.js Team. Leaflet Documentation. URL: https://leafletjs.com/reference.html (дата звернення: 10.04.2026).
9. OpenWeather Inc. OpenWeatherMap API Documentation. URL: https://openweathermap.org/api (дата звернення: 10.04.2026).
10. Flanagan D. JavaScript: The Definitive Guide. 7th ed. O'Reilly Media, 2020. 706 p.
11. Fowler M. Patterns of Enterprise Application Architecture. Addison-Wesley, 2002. 533 p.
12. ДСТУ 8302:2015. Бібліографічне посилання. Загальні положення та правила складання. Київ: ДП «УкрНДНЦ», 2016. 17 с.
13. ДСТУ 3.2-92:1993. Забруднення атмосфери. Граничне допустиме стоси шкідливих речовин у повітрі робочої зони. Київ, 1993. 12 с.
14. Борисенко В. В., Салащенко М. М. Моделювання розповсюдження забруднюючих речовин в атмосфері. Вісник КНУ імені Тараса Шевченка. 2019. № 1(76). С. 28–35.

15. Лекція: Нормальні форми бази даних – [Електронний ресурс] – режим доступу: https://javarush.com/ua/quests/lectures/ua.questhibernate.level17.lecture02 (дата звернення: 19.04.2025)
16. Нормалізація СУБД – [Електронний ресурс] – режим доступу: https://www.guru99.com/uk/database-normalization.html (дата звернення: 19.04.2025)
17. Файл-Серверні – [Електронний ресурс] – режим доступу: http://ni.biz.ua/14/14_2/14_28675_fayl-servernie.html#google_vignette (дата звернення: 20.04.2025)
18. 8 Top Database Management Systems (DBMS) – [Електронний ресурс] – режим доступу: https://www.pipedrive.com/en/blog/database-management-systems (дата звернення: 20.04.2025)
19. What is SQL Server – [Електронний ресурс] – режим доступу: https://learn.microsoft.com/uk-ua/sql/sql-server/what-is-sql-server?view=sql-server-ver16 (дата звернення: 20.04.2025)
20. Основні об’єкти структури бази даних sql-серверу – [Електронний ресурс] – режим доступу: https://studfile.net/preview/14501229/page:5/ (дата звернення: 21.04.2025)
21. Фізичне проектування структури бази даних – [Електронний ресурс] – режим доступу: https://rdb.dp.ua/uk/chapter_04 (дата звернення: 21.04.2025)
22. Anthropic. Claude AI Assistant (claude.ai). URL: https://claude.ai (дата звернення: 10.04.2026).
23. https://chatgpt.com/

ДОДАТКИ

ДОДАТОК А

Створення таблиць

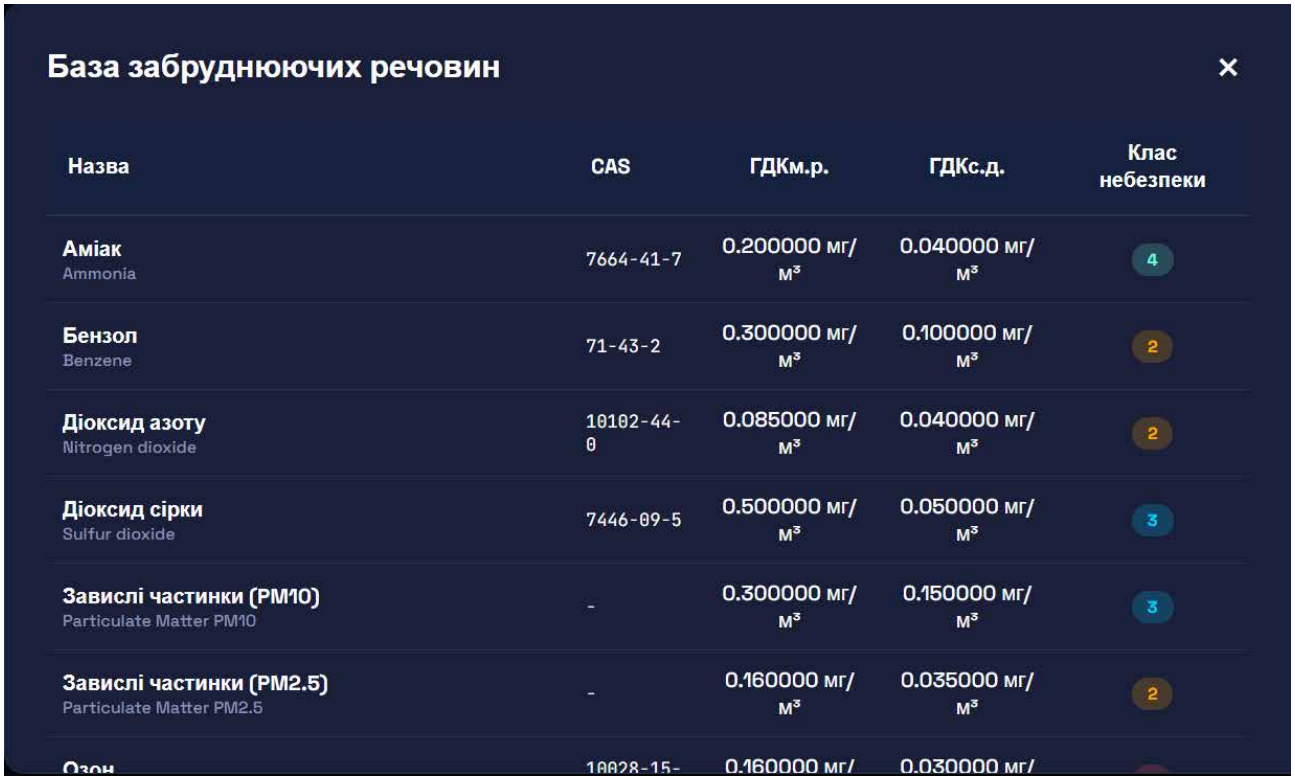
```
-- Таблиця користувачів
CREATE TABLE IF NOT EXISTS users (
    id SERIAL PRIMARY KEY,
    full_name VARCHAR(255) NOT NULL,
    email VARCHAR(255) UNIQUE NOT NULL,
    password_hash VARCHAR(255) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    last_login TIMESTAMP,
    is_active BOOLEAN DEFAULT true
);
-- Таблиця речовин (забруднювачів)
CREATE TABLE IF NOT EXISTS substances (
    id SERIAL PRIMARY KEY,
    name_ua VARCHAR(255) NOT NULL,
    name_en VARCHAR(255),
    cas_number VARCHAR(50),
    molecular_weight DECIMAL(10, 4), -- молекулярна маса, г/моль
    density DECIMAL(10, 4), -- щільність, г/см³
    mac_single DECIMAL(10, 6), -- ГДКм.р. (мг/м³) - максимальна разова
    mac_daily DECIMAL(10, 6), -- ГДКс.д. (мг/м³) - середньодобова
    danger_class INTEGER, -- клас небезпеки (1-4)
    aggregation_state VARCHAR(50), -- агрегатний стан (газ, рідина, тверда речовина, аерозоль)
    description TEXT,
    created_by INTEGER REFERENCES users(id),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    is_active BOOLEAN DEFAULT true
);
-- Таблиця проектів моделювання
CREATE TABLE IF NOT EXISTS projects (
    id SERIAL PRIMARY KEY,
    user_id INTEGER NOT NULL REFERENCES users(id) ON DELETE CASCADE,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    location_name VARCHAR(255), -- назва місця
    latitude DECIMAL(10, 7) NOT NULL, -- широта джерела викиду
    longitude DECIMAL(10, 7) NOT NULL, -- довгота джерела викиду
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    last_calculated TIMESTAMP,
```

```
    is_active BOOLEAN DEFAULT true
);

-- Таблиця параметрів викиду для проектів
CREATE TABLE IF NOT EXISTS emission_parameters (
    id SERIAL PRIMARY KEY,
    project_id INTEGER NOT NULL REFERENCES projects(id) ON DELETE CASCADE,
    substance_id INTEGER NOT NULL REFERENCES substances(id),

    -- Параметри джерела викиду
    source_height DECIMAL(10, 2) NOT NULL, -- висота джерела викиду, м
    source_diameter DECIMAL(10, 2), -- діаметр устя джерела, м
    exit_velocity DECIMAL(10, 2), -- швидкість виходу газоповітряної суміші, м/с
    exit_temperature DECIMAL(10, 2), -- температура виходу, °C
    emission_rate DECIMAL(15, 6) NOT NULL, -- потужність викиду, г/с

    -- Метеорологічні умови
    wind_speed DECIMAL(10, 2), -- швидкість вітру, м/с
    wind_direction DECIMAL(10, 2), -- напрямок вітру, градуси
    air_temperature DECIMAL(10, 2), -- температура повітря, °C
    atmospheric_pressure DECIMAL(10, 2), -- атмосферний тиск, гПа
    humidity DECIMAL(10, 2), -- вологість, %
    stability_class VARCHAR(10), -- клас стійкості атмосфери (A, B, C, D, E, F)

    -- Додаткові параметри
    roughness_length DECIMAL(10, 4), -- параметр шорсткості поверхні, м
    calculation_distance DECIMAL(10, 2) DEFAULT 5000, -- відстань розрахунку, м

    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Таблиця результатів моделювання
CREATE TABLE IF NOT EXISTS simulation_results (
    id SERIAL PRIMARY KEY,
    project_id INTEGER NOT NULL REFERENCES projects(id) ON DELETE CASCADE,
    emission_parameter_id INTEGER NOT NULL REFERENCES emission_parameters(id),

    -- Результати розрахунків
    max_concentration DECIMAL(15, 8), -- максимальна концентрація, мг/м³
    max_concentration_distance DECIMAL(10, 2), -- відстань до максимуму, м
    dangerous_zone_radius DECIMAL(10, 2), -- радіус небезпечної зони (>ГДК), м

    -- JSON дані для візуалізації
    concentration_grid JSONB, -- сітка концентрацій для побудови ізоліній
    isolines_data JSONB, -- дані ізоліній
```

```
    calculation_time DECIMAL(10, 3), -- час розрахунку, секунди
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Таблиця звітів
CREATE TABLE IF NOT EXISTS reports (
    id SERIAL PRIMARY KEY,
    project_id INTEGER NOT NULL REFERENCES projects(id) ON DELETE CASCADE,
    simulation_result_id INTEGER REFERENCES simulation_results(id),

    report_type VARCHAR(50) NOT NULL, -- тип звіту (PDF, DOCX)
    file_name VARCHAR(255) NOT NULL,
    file_path VARCHAR(500),
    file_size INTEGER, -- розмір файлу в байтах

    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    created_by INTEGER REFERENCES users(id)
);

-- Індекси для оптимізації запитів
CREATE INDEX idx_projects_user_id ON projects(user_id);
CREATE INDEX idx_emission_parameters_project_id ON emission_parameters(project_id);
CREATE INDEX idx_simulation_results_project_id ON simulation_results(project_id);
CREATE INDEX idx_reports_project_id ON reports(project_id);
CREATE INDEX idx_substances_name ON substances(name_ua);

-- Тригери для автоматичного оновлення updated_at
CREATE OR REPLACE FUNCTION update_updated_at_column()
RETURNS TRIGGER AS $$
BEGIN
    NEW.updated_at = CURRENT_TIMESTAMP;
    RETURN NEW;
END;
$$ language 'plpgsql';

CREATE TRIGGER update_users_updated_at BEFORE UPDATE ON users
    FOR EACH ROW EXECUTE FUNCTION update_updated_at_column();

CREATE TRIGGER update_substances_updated_at BEFORE UPDATE ON substances
    FOR EACH ROW EXECUTE FUNCTION update_updated_at_column();

CREATE TRIGGER update_projects_updated_at BEFORE UPDATE ON projects
    FOR EACH ROW EXECUTE FUNCTION update_updated_at_column();

CREATE TRIGGER update_emission_parameters_updated_at BEFORE UPDATE ON emission_parameters
                                                    FOR EACH ROW EXECUTE FUNCTION update_updated_at_column();
```

ДОДАТОК Б

База речовин



База забруднюючих речовин

Назва	CAS	ГДКм.р.	ГДКс.д.	Клас небезпеки
Аміак Ammonia	7664-41-7	0.200000 мг/м³	0.040000 мг/м³	4
Бензол Benzene	71-43-2	0.300000 мг/м³	0.100000 мг/м³	2
Діоксид азоту Nitrogen dioxide	10102-44-0	0.085000 мг/м³	0.040000 мг/м³	2
Діоксид сірки Sulfur dioxide	7446-09-5	0.500000 мг/м³	0.050000 мг/м³	3
Завислі частинки (PM10) Particulate Matter PM10	-	0.300000 мг/м³	0.150000 мг/м³	3
Завислі частинки (PM2.5) Particulate Matter PM2.5	-	0.160000 мг/м³	0.035000 мг/м³	2
Озон	10028-15-	0.160000 мг/	0.030000 мг/	

ДОДАТОК В

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ
І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

Декларація про академічну доброчесність та використання ШІ

ДЕКЛАРАЦІЯ ЗДОБУВАЧА

Я, Федосенко Олександр Леонідович, здобувач НУБіП України, усвідомлюючи відповідальність за порушення академічної доброчесності, цією заявою підтверджую, що:

1. Моя бакалаврська кваліфікаційна робота (проєкт) на тему «Програмне забезпечення оцінки розповсюдження забруднюючої аерозольної хмари в антропогенному середовищі» є результатом моєї особистої праці.
2. Усі запозичення з текстів інших авторів мають відповідні посилання згідно з чинними стандартами.
3. Щодо використання інструментів ШІ зазначаю, що (обрати необхідне):
    - [ ] інструменти генеративного ШІ не використовувалися.
    - [+] інструменти ШІ (вказати назву: ChatGPT, Gemini, Claude, DeepL, ChatGPT, Claude інші (вказати назву)) були використані для:
        - [+] перекладу іншомовних джерел;
        - [+] стилістичного редагування та перевірки граматики;
        - [+] допомоги у написанні/відлагодженні програмного коду;
        - [ ] інше: _____.

Я розумію, що надання недостовірної інформації може призвести до скасування результатів захисту та відрахування з числа здобувачів НУБіП України.

«___» _____ 2026 р. _____ **Ім’я ПРІЗВИЩЕ**
(підпис)