НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ

кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ

На правах рукопису"
УДК_____

“До захисту допущено”
Завідувач кафедри ЦТЕ
_____ **Наталія АУШЕВА**
“___”_____2023 р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою
“Цифрові технології в енергетиці”
зі спеціальності 122 “*Комп'ютерні науки*”

на тему: Методи візуалізації на інтерактивній карті територій, потерпілих від наслідків_____

Виконав: студент 2 курсу, групи ТР-21мп
__Островоой Денис Олегович_____ _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник _Доцент к.в.н. Онисько Андрій Ілліч _____
(посада, вчене звання, науковий ступінь, ім’я ПРІЗВИЩЕ) (підпис)

Рецензент_____ _____
(посада, вчене звання, науковий ступінь, ім’я ПРІЗВИЩЕ) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ - 2023

Національний технічний університет України
“ Київський політехнічний інститут ім. Ігоря Сікорського”

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ

Кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ
Рівень вищої освіти другий (магістерський)
За освітньою програмою ""“Цифрові технології в енергетиці”
Спеціальності 122 Комп’ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Наталія АУШЕВА
(підпис)
«____» _____ 2023р.

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ ДИСЕРТАЦІЮ СТУДЕНТУ

Островому Денису Олеговичу
(прізвище, ім’я, по батькові)

1. Тема дисертації Методи візуалізації на інтерактивній карті територій, потерпілих від наслідків

Науковий керівник к.в.н., доцент Онисько Андрій Ілліч
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від “06” листопада 2023 року №5152-с
2. Строк подання студентом дисертації 18 грудня 2023р
3. Вихідні дані до роботи готова система для візуалізації на інтерактивній карті територій, потерпілих від наслідків
4. Перелік питань, які потрібно розробити аналіз подібних систем, реалізувати збір даних, аналіз даних, прогнозування подій, інтерфейс системи
5. Орієнтований перелік ілюстративного матеріалу
Постановка задачі, Засоби розробки, Збір даних, База даних, Аналіз і навчання моделі, Архітектура системи, Діаграма прецедентів, Інтерфейс, Висновки.

6. Орієнтований перелік публікацій

7. Дата видачі завдання «22» лютого 2023р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	Отримати завдання	22.02.2023	
2.	Практика	01.09.2023 – 26.10.2023	
3.	Збір інформації	15.09.2023 – 10.10.2023	
4.	Аналіз вимог завдання, розробка методів і засобів розв’язання оставленої задачі	10.10.2023	
5.	Розробка та тестування програмного продукту	11.10.2023- 26.10.2023	
6.	Виконання розділів дисертації (практична частина, загальні висновки, список джерел)	26.10.2023	
7.	Написання основних розділів автореферату	01.11.2023- 04.12.2023	
8.	Перевірка дисертації науковим керівником	14.11.2023	
9.	Подання в електронному вигляді роботи та анотації до неї на перевірку нормоконтролера та плагіат (UNICHECK)		
10.	Надання документів на засідання кафедри		
11.	Предзахист магістерської дисертації та допуск до захисту дисертації	06.12.2023	
12.	Подання магістерської дисертації рецензенту. Отримання рецензії.		
13.	Подання пакету документів по магістерської дисертаціїта супровідних до неї документів до захисту в ЕК		
14.	Захист магістерської дисертації		

Студент _____ Островой Д. О.
(підпис) (прізвище та ініціали)

Науковий керівник _____ Онисько А. І.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка складається зі вступу, п’яти розділів, висновку, списку використаних джерел; містить 84 сторінок, 17 рисунки, 16 таблиць Список використаних джерел включає 24 бібліографічних найменувань.

Мета роботи:

Метою даної роботи є розробка та оптимізація методів візуалізації на інтерактивних картах, територій, що постраждали від різноманітних наслідків, для підвищення ефективності прийняття рішень у кризових ситуаціях.

Завдання дослідження:

Проаналізувати існуючі методи візуалізації на інтерактивних картах.

Розробити метод візуалізаційних методик відображення.

Створити прототип системи візуалізації з використанням сучасних програмних інструментів.

Тестування системи на наявність помилок.

Визначити напрямки для подальшого вдосконалення системи.

Об’єктом дослідження виступають інтерактивні карти та системи візуалізації, що використовуються для відображення наслідків природних катастроф, надзвичайних ситуацій багатьох видів.

Предметом є методи візуалізації, які дозволяють представити дані про наслідки катастроф на інтерактивних картах, їхня чіткість, точність.

Ключові слова: *надзвичайні події, Python, Leaflet, метеоінформація, інтерактивна мапа, прогнозування, інтерполяція*

ABSTRACT

The explanatory note consists of an introduction, five chapters, a conclusion, a list of used sources; contains 84 pages, 17 figures, 16 tables. The list of used sources includes 24 bibliographic names.

The goal of the work:

The purpose of this work is to develop and optimize methods of visualization on interactive maps of territories affected by various consequences to increase the effectiveness of decision-making in crisis situations.

Objectives of the study:

Analyze existing methods of visualization on interactive maps.

Develop a method of visualization mapping technologies.

Create a prototype visualization system using modern software tools.

Testing the system for errors.

Determine directions for further improvement of the system.

The object of research is interactive maps and visualization systems used to display the consequences of natural disasters and emergency situations of many kinds.

The subject is visualization methods that allow presenting data on the consequences of disasters on interactive maps, their clarity, accuracy.

Keywords: *emergencies, Python, Leaflet, weather information, interactive map, forecasting, interpolation*

ЗМІСТ

ВСТУП.....9
1 ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ.....11
1.1 Розробка серверної частини додатка візуалізації на інтерактивній карті територій, потерпілих від наслідків.....11
1.2 Створення бази даних територій постраждалих від наслідків.....12
1.3 Створення інтерфейсу користувача системи.....12
1.4 Методи візуалізації та топологічного з’єднання.....13
1.5 Огляд існуючих рішень.....14
1.5.1 Платформа Google Disaster Map.....15
1.5.2 Платформа Global Disaster Alert and Coordination System.....16
1.5.3 Платформа European Forest Fire Information System.....18
1.6 Засоби розробки.....19
1.6.1 Мова програмування Python.....20
1.6.2 Мова програмування JavaScript.....21
1.6.3 Каскадні таблиці стилів CSS.....21
1.6.4 Фреймворк Flask.....24
1.6.5 База даних MySQL.....25
1.6.6 Бібліотека SQLAlchemy.....26
1.6.7 Фреймворк AdminLTE.....26
1.6.8 Бібліотека Leaflet.....27
1.6.9 Бібліотека BCrypt.....28
1.6.10 Середа розробки Visual Studio Code.....28

Висновки до розділу 1.....30
2 ОПИС МЕТОДІВ ВИРІШЕННЯ ЗАДАЧІ.....31
2.1 Математична модель.....31
2.1.1 Афінні перетворення.....32
2.1.2 Інтерполяція.....37
2.1.3 Алгоритм обробки геоданих.....40
2.2 Побудова інтерфейсу користувача.....41
Висновки до розділу 2.....42
3 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ.....43
3.1 Структура Бази даних.....43
3.2 Система міграцій.....45
3.3 Використання криптографії для покращення безпеки.....46
3.4 Протоколи доступу до веб додатку.....48
3.5 Рівні доступу користувачів та їхні ролі.....49
3.6 Архітектура програмної системи.....50
3.6.1 Діаграма Класів.....50
3.6.2 Діаграма прецедентів.....54
3.6.3 BPMN діаграма.....56
Висновки до розділу 3.....58
4 РОБОТА КОРИСТУВАЧА З ПРОГРАМОЮ.....59
4.1 Загальні характеристики застосунку.....59
4.2 Вимоги до обчислювальної техніки.....60
4.3 Інтерфейс користувача.....62
Висновки до розділу 4.....66
5 РОЗРОБКА СТАРТАП ПРОЕКТУ.....67
5.1 Опис ідеї стартап-проекту.....67

5.2 Технологічний аудит ідеї проекту..... 69
5.3. Аналіз ринкових можливостей запуску стартап-проекту..... 70
5.4. Розроблення ринкової стратегії проекту.....77
Висновки до розділу 5.....81
ВИСНОВКИ..... 82
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ..... 83

ВСТУП

У сучасному світі, який називають інформаційною ерою, людство оточене величезною кількістю інформації. Враховуючи значення своєчасного доступу до даних в критичних ситуаціях, розроблено ряд інтерактивних платформ, які допомагають стежити за розвитком та наслідками надзвичайних подій. Наприклад, "Google Disaster Map" є інструментом, який інтегрує інформацію про катастрофи з надійних джерел на інтерактивній карті, дозволяє користувачам участь у зборі цінних даних. Погода, будучи часто непередбачуваним фактором, має величезний вплив на розвиток та наслідки надзвичайних ситуацій. Швидкі зміни погодних умов можуть не лише спровокувати природні катастрофи, але й суттєво ускладнити відповідь на них. Інструменти, такі як "MapBox", не лише надають можливості для створення детальних карт, але й дозволяють інтегрувати погодні дані, що може бути життєво важливим для планування евакуаційних заходів або розгортання допомоги. Форуми та спільноти зібрали навколо себе професіоналів та ентузіастів, які можуть обмінюватися важливою інформацією та стратегіями реагування на швидкі зміни умов. Однак, доступ до цих ресурсів може бути обмежений або вимагати оплати, тому користувачам слід уважно вибирати платформу, керуючись актуальністю, надійністю та доступністю інформації, яка їм потрібна. Також розподіл доступу до даних є фундаментальною складовою інформаційної безпеки. В ідеалі, кожен користувач системи отримує доступ лише до тієї інформації, яка необхідна для виконання його професійних завдань. Це відокремлення прав доступу допомагає захистити чутливу інформацію та дані від несанкціонованого перегляду або змін. Наприклад, звичайні користувачі можуть мати дозвіл тільки на перегляд даних, тоді як адміністратори мають права на редагування та видалення. Чутливість інформації визначається її важливістю та потенційним впливом, який її розголошення чи зміна може мати. Інформація, що містить персональні дані,

фінансові звіти, інтелектуальну власність, або національну безпеку, вимагає особливих мір обережності та захисту. Такі дані часто підпадають під дію регуляторних вимог, таких як GDPR у Європейському Союзі, що вимагає від організацій забезпечувати належний захист персональних даних. Для запобігання зміні інформації з поганими намірами, такими як фальсифікація даних або шкідливе втручання, необхідні комплексні системи безпеки. Це включає в себе шифрування даних, постійний моніторинг системи на предмет підозрілої поведінки та регулярні аудити безпеки. Важливим є також наявність чіткої політики інформаційної безпеки та програм навчання для співробітників, які підвищують обізнаність щодо потенційних кіберзагроз та методів їх запобігання. Кінцевою метою цих заходів є створення надійного середовища, де інформація захищена від неавторизованого доступу та зловмисного використання, а права та обов'язки користувачів чітко регламентовані та зрозумілі для всіх учасників процесу.

1 ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

Розробка системи візуалізації на інтерактивній карті територій, потерпілих від наслідків, включає наступні кроки: створення серверної частини з реалізацією методів для візуалізації даних на карті, алгоритмів розрахунку наслідків для територій та розробку бази даних для збереження інформації про території та їх стан, створення інтерфейсу для взаємодії користувача з цією базою даних, а також інтерфейсу для оператора, щоб дозволити їм працювати з системою візуалізації та додавати або змінювати дані і отримувати інформацію про потерпілі території.

1.1 Розробка серверної частини додатка візуалізації на інтерактивній карті територій, потерпілих від наслідків

Програмний продукт для візуалізації на інтерактивній карті територій, потерпілих від наслідків, повинен включати повний набір інструментів, необхідних для програмної реалізації ефективного алгоритму візуалізації даних на карті, алгоритму прогнозування наслідків в залежності від погодних умов та в залежності від стану погоди візуалізація подальших переміщень або змін площі ураження територій. Реалізація цього програмного продукту повинна включати всі необхідні властивості для представлення територій та їх стану, а також мапу для графічного відображення цих даних. Програмний продукт повинен бути розроблений з використанням мов програмування Python та JavaScript для забезпечення взаємодії з веб-середовищем.

1.2 Створення бази даних територій постраждалих від наслідків

Для візуалізації на інтерактивній карті територій, що постраждали від наслідків, потрібно створити базу даних, яка буде зв'язувати географічні об'єкти на карті з інформацією про їх стан. Ця база даних повинна надавати можливість виконувати такі завдання:

– Додавання та редагування інформації про різні географічні об'єкти на карті.

– Додавання та редагування інформації про стан та характеристики територій,
що постраждали від наслідків.

1.3 Створення інтерфейсу користувача системи

Інтерфейс користувача для роботи з візуалізацією територій, що постраждали від наслідків, повинен дозволяти користувачеві створювати графічні представлення цих територій, завантажувати їх з бази даних, і проводити аналіз цих візуалізацій за допомогою спеціальних методів. Також, інтерфейс повинен надавати можливість відображати результати аналізу. Вся система і інтерфейс користувача повинні бути розроблені з використанням мов програмування Python та JavaScript для взаємодії з веб-середовищем.

У контексті системи для візуалізації на інтерактивній карті територій, що постраждали від наслідків, можна визначити два основних аспекти:

— Географічні дані та топології територій, які стали об'єктом візуалізації.

— Методи та алгоритми для ефективної візуалізації цих географічних даних.

Наряду з цим, слід відзначити, що існують різноманітні програмні засоби та бібліотеки для візуалізації географічних даних, такі як Leaflet, Mapbox, Google Maps, які можуть бути використані для цієї мети. На жаль, деякі з існуючих систем можуть вимагати від користувача додаткових знань та навичок у роботі з географічними інструментами.

1.4 Методи візуалізації та топологічного з’єднання

Система візуалізації територій, що постраждали від наслідків, має на меті надати користувачеві зручний інтерфейс для роботи з географічними даними та ефективні методи візуалізації цих даних на інтерактивній карті.

В випадку методів візуалізації на інтерактивній карті територій, які страждають від наслідків певних подій, можна розглядати структуру з'єднання об'єктів на цій карті. Ці об'єкти можуть бути пов'язані різними способами, такими як повне з'єднання, функціональна зв'язаність, неповне з'єднання та інші. Ці зв'язки визначають, як об'єкти пов'язані один з одним в контексті подій на карті. Важливо також враховувати фізичну та логічну організацію об'єктів на карті. Фізична топологія відображає фактичні з'єднання та розташування об'єктів, тоді як логічна топологія визначає логічні зв'язки та ієрархію між ними.

При роботі з методами візуалізації на карті територій можна розглядати різні типи топологічних взаємозв'язків для об'єктів. Наприклад, об'єкти можуть мати повну зв'язаність, коли кожен об'єкт пов'язаний з кожним іншим, або неповну зв'язаність, де існують як прямі, так і непрямі зв'язки. В випадку методів візуалізації на інтерактивній карті територій, потрібно розглядати різні можливості організації об'єктів, включаючи ієрархічну структуру. Кожна з цих структур може мати важливе значення при відображенні подій на карті та взаємодії з ними.

1.5 Огляд існуючих рішень

Аналіз існуючих рішень щодо візуалізації наслідків надзвичайних ситуацій на інтерактивних картах вказує на широкий спектр доступних інструментів та підходів, кожен з яких використовує різні методи для представлення даних. Наприклад, Google Disaster Map використовує супутникові знімки та накладення даних для ілюстрації розмаху стихійних лих, тоді як Crisis Hub забезпечує платформу для користувачів, щоб ділитися свіжою інформацією про поточні події. Важливим методом візуалізації є використання шарів даних, які можуть бути нашаровані один на одного для показу різних аспектів кризи. Це може включати демографічні дані, інформацію про транспортну інфраструктуру, розташування медичних установ, а також метеорологічні умови. Ще один метод — це використання "гарячих карт" (heat maps), які показують концентрацію певних подій або статистики, таких як кількість пошкоджених будівель або рівень забруднення. Ці карти використовують кольорове кодування для легкого розуміння масштабів проблеми. Інтерактивність також є ключовою, дозволяючи користувачам клікати на різні об'єкти та отримувати додаткову інформацію або переглядати зміни в часі через слайдери чи хронологічні шкали. Це може допомогти відслідковувати прогрес відновлення територій, що постраждали від катастроф, або аналізувати тренди розвитку подій. Розроблення інтерактивних карт, які візуалізують території, що постраждали від наслідків, вимагає врахування різноманітних джерел даних та використання комплексних геопросторових аналітичних інструментів. Ці інструменти повинні бути інтуїтивно зрозумілими для користувачів, дозволяючи їм легко інтерпретувати складні дані та адаптувати їх для конкретних потреб і ситуацій.

1.5.1 Платформа Google Disaster Map

Google Disaster Map - це важливий ініціативний проект, спрямований на забезпечення громадськості та владам швидкого та доступного доступу до інформації про надзвичайні ситуації та кризові події, які виникають по всьому світу як показано на рисунку 1.5.1. Проект став особливо актуальним під час пандемії COVID-19 та інших кризових ситуацій. Велика кількість інформації та зручний інтерфейс дозволили людям отримувати необхідні знання та рекомендації для вжиття заходів у разі надзвичайних обставин.

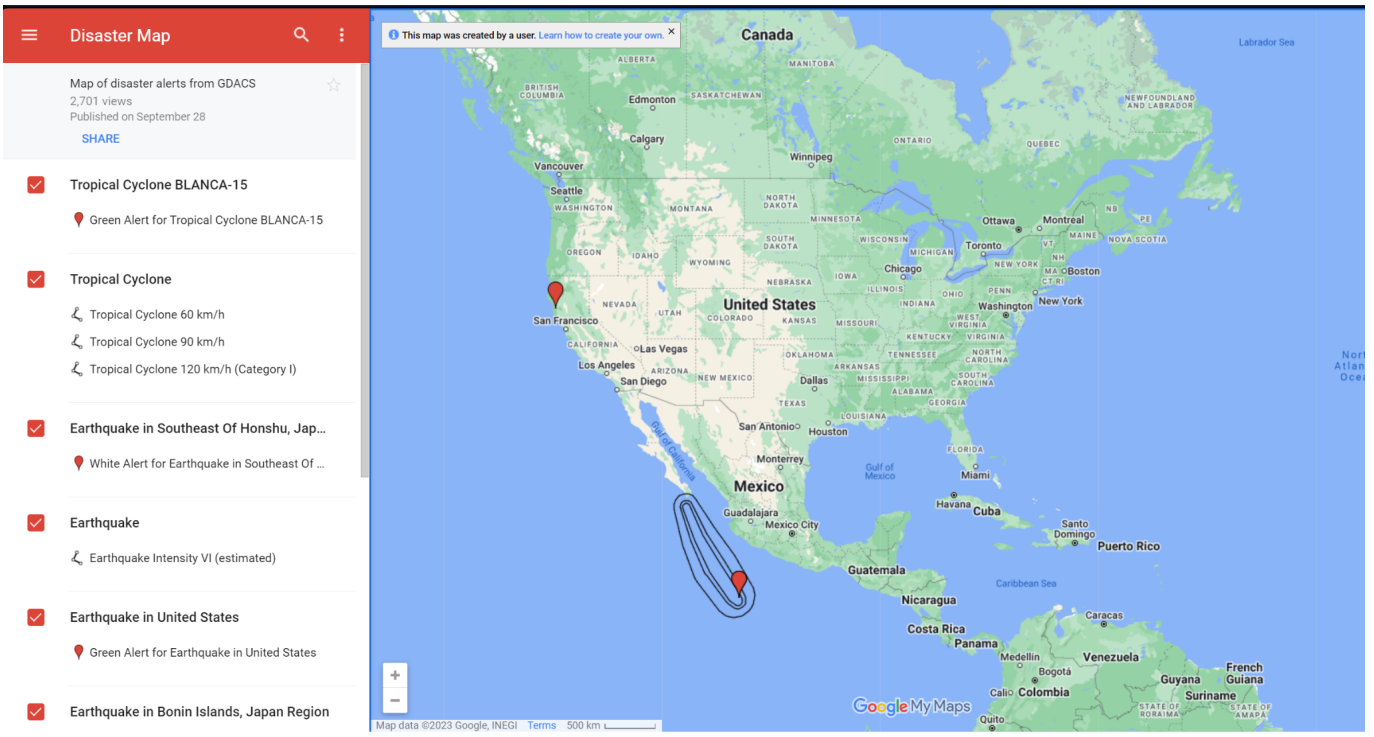


Рисунок 1.1 – Інтерфейс Google disaster map

Цей інструмент розроблений з метою надання актуальних та достовірних даних про різноманітні катастрофи, природні лиха, техногенні аварії та інші небезпечні ситуації. Google Disaster Map надає користувачам можливість в

реальному часі слідкувати за розвитком подій, перевіряти стан справ на практиці, а також отримувати оновлення та рекомендації стосовно безпеки.

Google Disaster Map відіграє важливу роль у забезпеченні громадськості і фахівців актуальною інформацією, що є вирішальним чинником для збереження безпеки та зменшення негативних наслідків надзвичайних ситуацій.

1.5.2 Платформа Global Disaster Alert and Coordination System

Global Disaster Alert and Coordination System (GDACS) - це міжнародна ініціатива, створена для поліпшення реагування на великі катастрофи шляхом забезпечення своєчасних та точних попереджень та аналізу потенційних наслідків. GDACS є спільним проектом Організації Об'єднаних Націй, Європейського Союзу та ряду інших партнерів приклад зовнішнього вигляду наведено на рисунку 1.5.2.

– Раннє попередження: GDACS використовує різні технології для моніторингу природних катастроф, таких як землетруси, цунамі, тропічні циклони, повені та вулканічні виверження. Система аналізує дані в реальному часі та видає попередження, коли катастрофа може мати серйозні наслідки.

– Оцінка наслідків: Після виявлення потенційної катастрофи GDACS проводить швидку оцінку її можливих наслідків. Це включає аналіз таких факторів, як чисельність населення у зоні ризику, вразливість інфраструктури та історичні дані про попередні катастрофи.

– Координація реагування: GDACS сприяє координації міжнародних зусиль з надання допомоги. Це включає обмін інформацією між урядами, гуманітарними організаціями та іншими ключовими стейкхолдерами.

– Комунікаційна платформа: GDACS служить як платформа для обміну інформацією та ресурсами між різними організаціями, що займаються реагуванням на катастрофи. Це допомагає уникнути дублювання зусиль та забезпечує більш ефективне використання ресурсів.

– Інновації та розвиток: GDACS постійно працює над удосконаленням своїх технологій та методів аналізу. Це включає використання супутникових даних, геопросторових технологій, штучного інтелекту та інших інноваційних підходів для покращення якості своїх попереджень та аналізів.

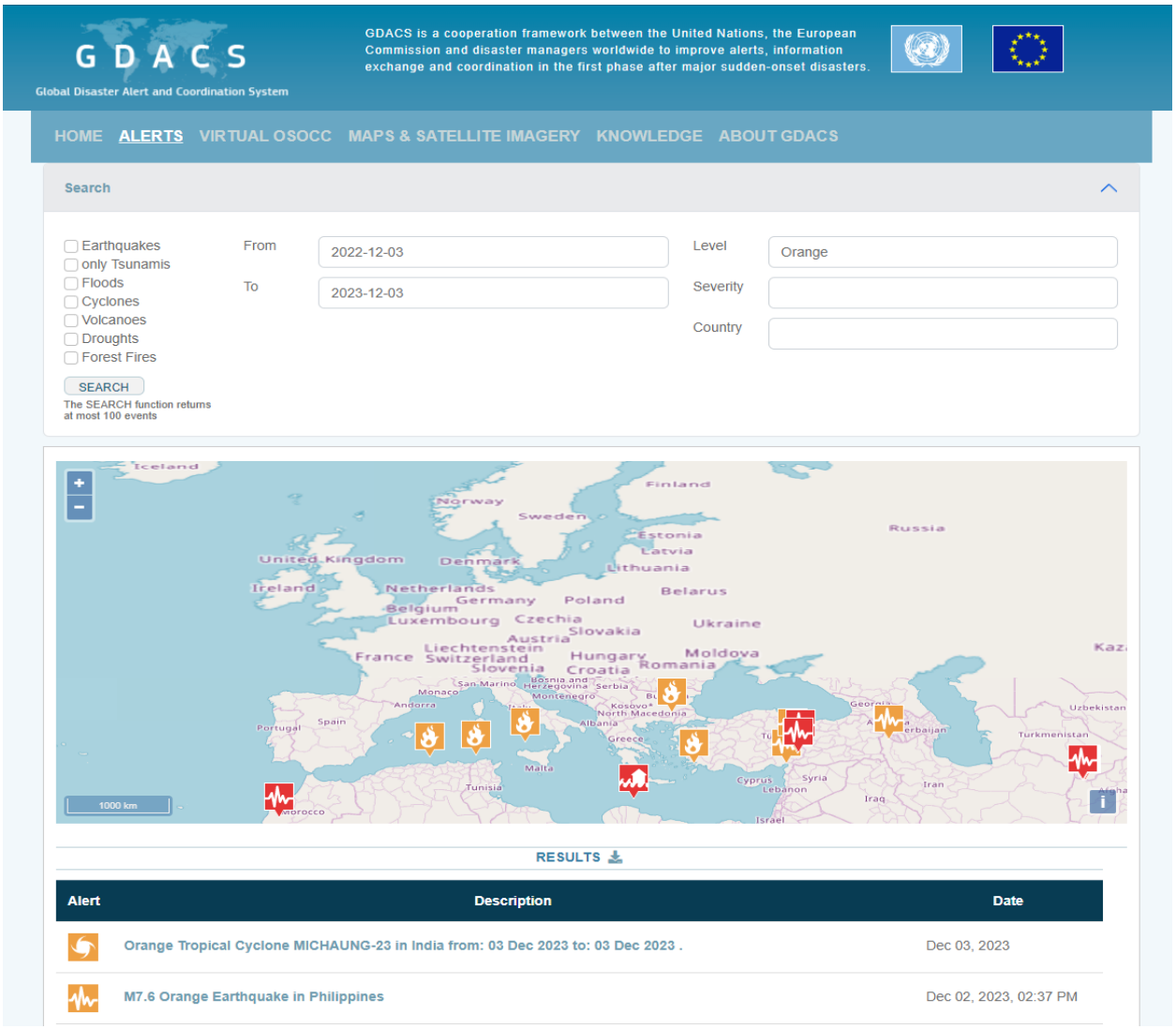


Рисунок 1.2 – Інтерфейс Global Disaster Alert and Coordination System

GDACS є важливим інструментом для міжнародної спільноти у відповіді на природні катастрофи, допомагаючи зменшити їх вплив та сприяючи більш ефективному та координованому реагуванню на надзвичайні ситуації.

1.5.3 Платформа European Forest Fire Information System

European Forest Fire Information System (EFFIS) – важливий інструмент для Європи у боротьбі з лісовими пожежами включає аналіз погодних умов, рослинності, та інших факторів, які можуть сприяти виникненню та поширенню пожеж. Він не тільки допомагає в оперативному реагуванні на пожежі, але й сприяє розумінню довгострокових тенденцій та впливу зміни клімату на лісові пожежі. Це дозволяє краще планувати заходи з запобігання пожежам та адаптації до змін клімату. Надає дані про лісові пожежі в Європі, включаючи їх розташування, інтенсивність та потенційні наслідки приклад інтерфейсу наведено на рисунку 1.5.3.

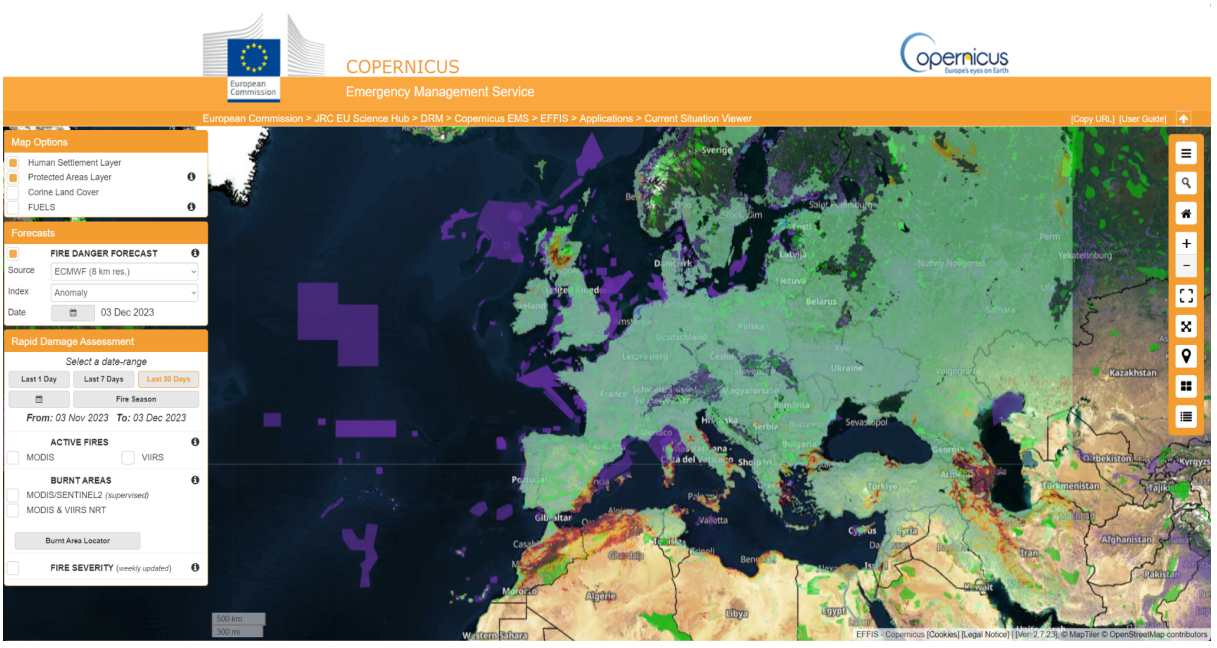


Рисунок 1.3 – Головна сторінка European Forest Fire Information System

Картографування пожеж: EFFIS створює детальні карти пожеж, які допомагають у плануванні заходів з гасіння та оцінці збитків. Ці карти включають інформацію про розмір, інтенсивність та напрямок поширення

пожеж. Аналіз наслідків пожеж: Після гасіння пожеж EFFIS допомагає оцінити їх вплив на лісові екосистеми, економіку та суспільство. Це включає оцінку збитків, втрату біорізноманіття та вплив на якість повітря.

Підтримка прийняття рішень: Інформація та аналіз, які надає EFFIS, є важливими для урядів та організацій, які займаються управлінням лісовими ресурсами та реагуванням на пожежі. EFFIS інтенсивно використовує дані з різних супутників, включаючи програму Copernicus Європейського Союзу, для моніторингу лісових пожеж та оцінки їх впливу. Геопросторовий аналіз: Система використовує геопросторові технології для створення детальних карт та аналітичних звітів. Інтеграція даних: EFFIS інтегрує дані з різних джерел, включаючи метеорологічні станції, лісові інвентаризації та інші джерела інформації.

1.6 Засоби розробки

Для розробки програмного продукту було використане велике різноманіття сучасних інструментів. Більшість з них можна побачити на рисунку 1.2. Для забезпечення роботи користувачів на різних девайсах, було прийнято рішення розробити саме веб застосунок.

Для розробки додатку було використано мови програмування Python та JavaScript також були використані бібліотеки та фреймворки:

–Фреймворк Flask

–База даних MySQL

–Бібліотека SQLAlchemy

–Фреймворк AdminLTE

–Бібліотека Leaflet

–Бібліотека BCrypt

1.6.1 Мова програмування Python

Python був розроблений у 1991 році, як легка для розуміння і зручна у використанні мова. Інтерпретатор Python та його стандартні бібліотеки доступні як у вихідній так і у скомпільованій формі на всіх основних платформах. Python є мультипарадигменою мовою та підтримує окрім об'єктно орієнтованого підходу також декілька інших парадигм, зокрема: процедурну, функціональну та аспектно-орієнтовану. Перевагами мови можна назвати простий та зрозумілий синтаксис та швидку розробку будь-яких додатків. Сьогодні її використовують для розробки веб- застосунків (зокрема бек-енд частини), програмування математики та написання системних скриптів.Python також широко використовується у наукових дослідженнях та аналізі даних, завдяки великій кількості наукових бібліотек, таких як NumPy, SciPy, Pandas, та Matplotlib. Ці бібліотеки надають потужні інструменти для обробки даних, статистичного аналізу та візуалізації, роблячи Python ідеальним вибором для обробки великих наборів даних та проведення складних наукових обчислень. Крім того, Python є популярною мовою у сфері веб-скрапінгу завдяки таким бібліотекам, як Beautiful Soup та Scrapy. Це дозволяє розробникам легко збирати та обробляти дані з веб-сторінок. Python також вирізняється своїм великим співтовариством, яке надає обширну підтримку та численні ресурси для навчання та вирішення проблем. Спільнота розробників активно підтримує офіційні форуми, блоги, конференції та мітапи, що сприяє швидкому навчанню та обміну досвідом. Враховуючи його універсальність та легкість інтеграції з іншими технологіями та API, Python став основним вибором для багатьох стартапів та технологічних компаній, що шукають швидкого впровадження нових ідей та продуктів. Однак, крім згаданої повільності, іншими недоліками Python є високий споживання пам'яті та менша ефективність для певних типів низькорівневих обчислень порівняно з мовами, як C або C++. Втім, ці недоліки частково компенсуються за допомогою інтеграції з оптимізованими на C/C++ бібліотеками або використання

інших оптимізаційних інструментів. Недоліком мови є її повільність виконання в порівнянні з компільованими аналогами.

1.6.2 Мова програмування JavaScript

JavaScript — це високорівнева, інтерпретована мова програмування, яка об'єднує елементи об'єктно-орієнтованого та функціонального програмування. Спочатку вона була створена для додавання інтерактивності веб-сторінкам у межах веб-браузерів. З часом JavaScript став невід'ємною частиною веб-розробки і знайшов застосування не лише у браузерах, але і на серверах, в мобільних додатках та в різноманітних IoT-пристроях.

Однією з ключових особливостей JavaScript є його інтерпретована природа, завдяки якій код може виконуватися без попередньої компіляції. Ця мова також відзначається підтримкою як об'єктно-орієнтованого, так і функціонального програмування, що надає розробникам гнучкість у виборі підходів до коду. Асинхронність в JavaScript, реалізована за допомогою промісів та async/await, дозволяє ефективно обробляти операції, що можуть тривати невизначений час, такі як запити до сервера.

1.6.3 Каскадні таблиці стилів CSS

CSS - мова графічного дизайну визначення та створення уявлення структурованого документа, написаного мовою розмітки. Він широко використовується для створення візуального оформлення веб-документів та інтерфейсів користувача, написаних на HTML або XHTML; мова може бути застосована до будь-якого XML-документу, включаючи XHTML, SVG, XUL, RSS тощо. CSS в першу чергу призначений для розділення змісту документа та

способу його представлення, таких як макети, кольори та шрифти. Цей поділ спрямований на покращення доступності документа, забезпечення більшої гнучкості та контролю у визначенні презентаційних функцій, дозволяє декільком HTML-документам використовувати один і той самий стиль за допомогою однієї окремої таблиці стилів у файлі .css, а також зменшує складність та повторення коду у структурі документа . Розділення форматування та змісту дозволяє представити один і той же розмічений документ у різних стилях для різних методів візуалізації, наприклад, на екрані, у друкованому вигляді, у мові (за допомогою мовного браузера або програми читання з екрану) та тактильних пристроїв на основі шрифту Брайля. Також можна по-різному відображати веб-сторінку залежно від розміру екрана або типу пристрою. Читачі можуть вказати іншу таблицю стилів, наприклад, таблицю стилів CSS, збережену на комп'ютері, для перезапису таблиці стилів дизайнера.

Розділення форматування та вмісту також дає можливість представити одну й ту саму сторінку розмітки в різних стилях для різних методів відтворення, наприклад, на екрані, у друкованому вигляді, голосом (через мовленнєвий браузер або програму зчитування з екрана). CSS також має правила альтернативного форматування, якщо доступ до вмісту здійснюється на мобільному пристрої.

Каскадність назв походить від заданої схеми пріоритету, щоб визначити, яке правило стилю застосовується, якщо більше одного правила відповідає певному елементу. Ця каскадна схема пріоритетів є передбачуваною.

CSS має простий синтаксис і використовує ряд англійських ключових слів для визначення назв різних властивостей стилю таблиця стилів складається зі списку правил. Кожне правило або набір правил складається з одного або декількох селекторів і блоку оголошення.

Інформація в CSS може бути надана з різних джерел. Такими джерелами можуть бути веб-браузер або користувач. Інформація може бути додатково класифікована на вбудовану, тип медіа, важливість, специфіку селектора, порядок

правил, успадкування та визначення властивостей. Інформація про стилі може бути в окремому документі або вбудовуватися в HTML файлі. Можна імпортувати декілька таблиць стилів. Залежно від використовуваного вихідного пристрою можна застосовувати різні стилі; наприклад, екранна версія може значно відрізнятися від друкованої, щоб автори могли адаптувати презентацію відповідно до кожного засобу.

Таблиця стилів з найвищим пріоритетом керує відображенням вмісту. Оголошення, не встановлені в джерелі з найвищим пріоритетом, передаються джерелу з нижчим пріоритетом, наприклад, стиль агента користувача. Процес називається каскадним. Однією з цілей CSS є надання користувачам більшого контролю над презентацією. Хтось вважає заголовки, виділені червоним курсивом, важко читаними, може застосувати іншу таблицю стилів. Залежно від браузера та веб-сайту, користувач може обрати одну з різних таблиць стилів, наданих дизайнерами, або може видалити всі додані стилі та переглянути сайт, використовуючи стиль браузера за замовчуванням, або може замінити лише стиль заголовка червоним курсивом, не змінюючи інші атрибути.

Спадкування є ключовою особливістю CSS; він покладається на відносини предок-нащадок для функціонування. Спадкування — це механізм, за допомогою якого властивості застосовуються не тільки до певного елемента, а й до його нащадків. Спадкування покладається на дерево документів, яке є ієрархією елементів XHTML на сторінці на основі вкладеності. Елементи-нащадки можуть успадковувати значення властивостей CSS від будь-якого елемента-предка, що їх охоплює. Загалом, елементи-нащадки успадковують властивості, пов’язані з текстом, але їх властивості, пов’язані з блоком, не успадковуються. Властивості, які можна успадкувати, — це колір, шрифт, інтервал між літерами, висота рядка, стиль списку, вирівнювання тексту, відступ тексту, перетворення тексту, видимість, пробіл і інтервал між словами. Властивості, які не можуть бути успадковані, — це фон, межа, відображення, плаваюча та чиста, висота та ширина, поля, мінімальна та максимальна висота та ширина, контур,

переповнення, відступ, положення, прикраса тексту, вертикальне вирівнювання та z-індекс. Спадкування можна використовувати, щоб уникнути повторного оголошення певних властивостей у таблиці стилів, дозволяючи коротший CSS. Спадкування в CSS не те саме, що успадкування в мовах програмування на основі класів, де можна визначити клас B як «як клас A, але з модифікаціями». За допомогою CSS можна стилізувати елемент із «класом A, але з модифікаціями». Однак неможливо визначити такий клас CSS B, який потім можна використовувати для стилізації кількох елементів без повторення змін.

1.6.4 Фреймворк Flask

Flask – це фреймворк Python, який був створений для спрощення розробки веб додатків

Flask вважається міні-фреймворком, оскільки він не потребує спеціальних сторонніх бібліотек чи засобів для розробки програмного забезпечення. У ньому присутні всі необхідні рівні абстракцій для роботи з файлами, роботи з базами даних та іншими додатками, які надають необхідні функції для розробників. Flask має також підтримку великої кількості розширень, які забезпечують додаткові можливості розробнику. Серед них можна виділити розширення для перевірки форм, контролю процесу завантаження, розгирення для маніпуляцій об'єктно-реляційних зв'язків, розширення для підтримки відкритих технологій аутентифікації.

Ключовими властивостями Flask є його простота в розробці програмного забезпечення, система керування запитами REST, підтримка шаблонів Jinja2, підтримка cookie на стороні веб-браузера, документація. Також варто відмітити розширення від сторонніх розробників, функціональну підтримку інших мов та підтримку Google App Engine.

Перевагами цього фреймворку можна назвати швидкість розробки на ньому, RESTful API, докладну документацію та підтримку Unicode.

Flask має також декілька недоліків, наприклад, складність в створенні складних багаторівневих веб-додатків та організація їх архітектури.

1.6.5 База даних MySQL

MySQL - це система управління базами даних (СУБД), яка використовує мову запитів SQL (Structured Query Language) для управління даними та налаштування її згідно з власними потребами. Це відкрите програмне забезпечення, що означає, що воно доступне для використання та модифікації будь-яким користувачем

Однією з основних переваг MySQL є її висока надійність та продуктивність, особливо в контексті веб-додатків та інших застосувань, що обробляють значні обсяги даних. Система відзначається своєю масштабованістю, здатністю ефективно працювати як з малими, так і з великими проектами. У сфері безпеки MySQL пропонує різноманітні можливості, включаючи шифрування даних та розширені функції аутентифікації та управління доступом. Підтримка транзакцій забезпечує цілісність та надійність даних. MySQL широко використовується для різних цілей, включаючи веб-сайти та додатки, де вона служить для зберігання інформації про користувачів, контент сайту та трансакційні дані. Вона також є популярним вибором для електронних комерційних платформ, систем управління контентом, таких як WordPress, Joomla та Drupal, а також для форумів та соціальних мереж, де використовується для управління користувацькими даними, публікаціями та коментарями.

1.6.6 Бібліотека SQLAlchemy

SQLAlchemy — це бібліотека на Python для роботи з реляційними базами даних. Вона надає набір високорівневих API для з'єднання з базами даних, виконання SQL-запитів та роботи з результатами. SQLAlchemy можна розглядати як набір інструментів для роботи з базами даних, який об'єднує в собі можливості ORM (Object-Relational Mapping) та традиційного SQL.

Однією з головних особливостей є наявність двох рівнів API. SQLAlchemy надає SQL Expression Language, що є низькорівневим інтерфейсом, близьким до SQL, і дозволяє виконувати запити. З іншого боку, є також ORM, який дозволяє працювати з даними на більш високому рівні, представляючи їх у вигляді об'єктів Python. SQLAlchemy підтримує багато різних СУБД, включаючи PostgreSQL, MySQL, SQLite та інші. Ця гнучкість робить його відмінним вибором для різних проектів. Крім того, завдяки своїй модульності і гнучкості, SQLAlchemy дозволяє розробникам легко адаптувати роботу з базами даних під специфічні потреби проекту.

1.6.7 Фреймворк AdminLTE

AdminLTE є одним із популярних рішень для розробників, коли йдеться про створення адміністративних панелей або контрольних панелей для веб-додатків. Цей темплейт, розроблений на основі Bootstrap, відомого frontend фреймворку, став вибором багатьох завдяки своїм готовим компонентам та елементам інтерфейсу.

Оскільки Bootstrap створений для реагування на різні розміри екрану, AdminLTE автоматично адаптує свій інтерфейс, що робить його дружнім до користувачів на будь-якому пристрої, від настільних комп'ютерів до мобільних телефонів. Це забезпечує універсальність та зручність користування в різних умовах.

Окрім основних компонентів, AdminLTE також інтегрується з численними JavaScript-бібліотеками, що дозволяє розширювати функціонал панелей. Наприклад, інтеграція з Chart.js дозволяє легко вбудовувати інтерактивні графіки, а DataTables спрощує роботу з табличними даними.

Також важливо відзначити, що AdminLTE не просто виглядає гарно, але і має добре структурований код. Це робить його ідеальним рішенням для розробників, які прагнуть не тільки швидко запустити проект, але й мати змогу легко його адаптувати та розширювати в майбутньому.

1.6.8 Бібліотека Leaflet

Leaflet – бібліотека для мови програмування JavaScript, що дозволяє легко створювати візуалізовані карти на основі векторних графіків. Він базується на Raphael.js для відображення векторних карти та дозволяє вам створювати інтерактивні карти з підтримкою подій, кольорових областей, маркерів та інших налаштувань, також з його допомогою можна візуалізовувати набори даних, таких як: дані про погоду та геолокаційні дані.

Основні переваги Leaflet включають можливість легкого налаштування, гнучкість у відображенні даних та інтеграції з іншими бібліотеками та інструментами. Також важливо відзначити, що плагін підтримує ряд готових

карти, таких як світова карта, карта США, карта Європи тощо, але користувачі також можуть додавати свої власні SVG-карти.

Крім основних функцій візуалізації, Leaflet також дозволяє анімувати елементи карти, що робить його ідеальним для створення інтерактивних демонстрацій або додатків з великою кількістю даних.

1.6.9 Бібліотека BCrypt

BCrypt - це криптографічний метод хешування паролів, розроблений для надання високого ступеня захисту. Його специфіка полягає в додаванні випадкової послідовності символів до пароля перед його перетворенням в хеш. Цей підхід ускладнює використання таблиць райдужних цін для злому та гарантує, що навіть ідентичні паролі матимуть різні хеші.

Відзначимо також адаптивний характер BCrypt. Враховуючи постійний розвиток обчислювальних технологій, алгоритм може змінювати ступінь своєї складності, аби залишатися стійким до спроб злому. Така особливість робить BCrypt особливо привабливим для захисту паролів в сучасних системах, надаючи надійний захист без втрати продуктивності.

1.6.10 Середа розробки Visual Studio Code

Visual Studio Code, відомий як VSCode, є безкоштовним і відкритим інтегрованим середовищем розробки (IDE) від Microsoft. Запущений у 2015 році, цей редактор швидко завоював популярність серед програмістів завдяки своїй багатофункціональності, адаптивності та широкому спектру можливостей.

Ключові особливості Visual Studio Code охоплюють:

1. Підтримка багатьох мов програмування VS Code підтримує численні мови, включно з JavaScript, TypeScript, Python, PHP, C++ та Java, завдяки різноманітним доступним розширенням.

2. Доступність Розширень та Плагінів: Це середовище дозволяє користувачам налаштовувати свій робочий процес з допомогою широкого вибору плагінів та розширень, доступних через вбудовану систему розширень.

3. Вбудований Термінал та Підтримка Систем Контролю Версій: VS Code включає інтегрований термінал та підтримує системи контролю версій як Git, дозволяючи керувати версіями коду та роботою з репозиторіями безпосередньо у середовищі.

4. Розширене Автозавершення Коду: Завдяки IntelliSense, VS Code пропонує продвинуте автозавершення коду, надаючи підказки щодо змінних, функцій, та інших елементів коду.

5. Функції Відладки: IDE містить розширені інструменти відладки, що дозволяють встановлювати точки зупинки, переглядати стек викликів і аналізувати змінні під час виконання програм.

6. Легкість Інтеграції з Хмарними Сервісами: Програма легко інтегрується з хмарними сервісами, як наприклад Microsoft Azure, сприяючи розробці та розгортанню додатків.

7. Кросплатформність: VS Code доступний на різних операційних системах, включаючи Windows, MacOS та Linux, роблячи його популярним серед широкого кола розробників.

8. Активна Спільнота та Підтримка: Завдяки високій популярності, середовище підтримується активною спільнотою, яка розробляє нові розширення та покращення, а також надає підтримку іншим користувачам.

Загалом, Visual Studio Code відзначається своєю легкістю, адаптивністю та широким набором можливостей, що робить його ідеальним вибором для розробників на всіх рівнях досвіду.

Висновки до розділу 1

У цьому розділі поставлено задачі до системи, розглянуто основні засоби розробки програмного продукту. Наведені їх основні переваги у застосуванні у даній нам предметній області. Оглянуті інструменти розраховані на розробку системи модульним підходом. Оглянуто аналогічні продукти для візуалізації на карті територій постраждалих від наслідків.

2 ОПИС МЕТОДІВ ВИРІШЕННЯ ЗАДАЧІ

В цьому розділі описано опис методів виконання поставленої задачі, які складають новизну системи та вирізняють її на тлі інших рішень. Будуть наведені дизайнерські рішення для побудови зручного інтерфейсу користувача, а також деталі імплементації сторонніх сервісів у додаток.

2.1 Математична модель

У математичній моделі для розрахунку наслідків надзвичайних ситуацій, погодні умови, зокрема вітер, відіграють значну роль. Використання алгоритмів інтерполяції дозволяє прогнозувати параметри вітру в тих точках, де безпосередні вимірювання відсутні, виходячи з наявних даних з файлів, як-от grib.json. Ці алгоритми беруть вектори вітру, які включають напрямки та швидкості вітру, і використовують їх для створення безперервного поля вітру на карті.

Сервіси, такі як NOMADS (NOAA Operational Model Archive and Distribution System), надають доступ до великих наборів метеорологічних даних, які можуть бути використані для вищезгаданої інтерполяції. Перетворення векторних даних із формату, що містить компоненти напрямку (x та y), до формату полярних координат (кут руху та швидкість) є критичним для точного моделювання та візуалізації поведінки вітру. Для візуалізації змін і перетворень вітрових векторів можуть бути застосовані афінні перетворення. Це дає можливість масштабувати, обертати та переносити дані таким чином, щоб вони були узгоджені з географічною моделлю території. Афінні перетворення вимагають визначити новий базис координатної системи, тобто перевизначити одиничні відрізки кожної з осей. Для аналізу впливу вітру на зону надзвичайної

події використовуються математичні операції, як-от множення вектору швидкості вітру на матрицю точок, що представляє зону події. У випадку кола як моделі зони надзвичайної події, це означає створення розподілу векторів вітру всередині кола для визначення, як вітер може вплинути на поширення диму, вогню або інших речовин, що розносяться повітрям. Ці розрахунки допомагають розуміти можливий вплив вітру на розповсюдження наслідків надзвичайної події та важливі для планування рятувальних операцій і евакуаційних заходів.

2.1.1 Афінні перетворення

Афінні перетворення можуть бути використані для розрахунку площі територій на інтерактивній карті у контексті територій, що постраждали від наслідків надзвичайних ситуацій. Це особливо актуально, коли потрібно враховувати зміни, що відбуваються внаслідок погодних умов, таких як затоплення або пожежі, викидів шкідливих речовин де площа постраждалої території може змінюватися.

За замовчуванням уражена територія представлена колом з радіусом, який позначає ступінь шкоди та дає розуміння в яких місцях потрібно проводити роботи з утрисування ускладнень та ліквідації наслідків, в залежності від погодних умов форма та розмір площі може змінюватись задля розуміння поточної ситуації. Афінні перетворення можуть бути використані для масштабування розміру кола відповідно до змін у погодних умовах. Наприклад, збільшення швидкості вітру може призвести до розширення зони пожежі , що може бути відображено на карті шляхом масштабування або зміни вигляду площі.

Для виконання змін було використано матрицю афінних перетворень, яка має вигляд матриці

$$T = \begin{bmatrix} X_x & Y_x & 0 \\ X_y & Y_y & 0 \\ X_0 & Y_0 & 1 \end{bmatrix}$$

де, O(x,y) – точка початку координат, X(x,y) – точка одиничного відрізку на осі абсцис, Y(x,y) – точка одиничного відрізку на осі ординат.

В програмній реалізації матриця має вигляд який представлено на рисунку 2.1

```
transform_matrix = [
    1, 0, 0,
    0, 1, 0,
    0, 0, 1
]
```

Рисунок 2.1 – Матриця афінних перетворень

Для для обчислення найкоротшої відстані між двома точками на поверхні еліпсоїда і визначення азимуту між ними для побудови оновленої площі ураження було використано формулу Вінсенті, яка має початкові значення

$$U_1 = arctan((1 - f) * tan(lat_1))$$

$$U_2 = arctan((1 - f) * tan(lat_2))$$

$$L = lon_2 - lon_1$$

$$\Lambda = L$$

Використовується ітераційний процес для знаходження Λ де Λ це різниця довготи двох точок на еліпсоїді.

Після знаходження Λ необхідно розрахувати кут σ між двома точками.

$$sin\,\sigma = \sqrt{(cos\,U_2 * sin\,\Lambda)^2 + (cos\,U_1 * sin\,U_2 - sin\,U_1 * cos\,U_2 * cos\,\Lambda)^2}$$

$$cos\,\sigma = sin\,U_1 * sin\,U_2 + cos\,U_1 * cos\,U_2 * cos\,\Lambda)$$

$$\sigma = arctan\,2(sin\,\sigma, cos\,\sigma)$$

Для розрахунку азимуту скористаємось формулою

$$sin\,\alpha = cos\,U_1 * cos\,U_2 * \frac{sin\,\Lambda}{sin\,\sigma}$$

$$cos^2\,\alpha = 1 - sin^2\alpha$$

$$cos\;\; 2\sigma_m = cos\;\; \sigma\;\; - 2 * sin\;\; U_1 * \frac{sin\,U_2}{cos^2\alpha}$$

Розрахунок відстані

$$C = f/16 * cos^2\alpha * (4 + f * (4 - 3 * cos^2\alpha))$$

$$L = \Lambda - (1 - C) * f * sin\,\alpha * (\sigma + C * sin\,\sigma * (cos\;\; 2\sigma_m + C * cos\,\sigma *$$

$$* (-\,1 + 2 * cos\;\; 2\sigma_m^{\,2})))$$

$$s = b * A * (\sigma - \Delta\sigma)$$

Де f – Сплюснення еліпсоїда.

a,b – Великий та малий радіуси еліпсоїда.

lat_1, lat_2 – Широти початкової та кінцевої точок.

lon_1 , lon_2 – Довготи початкової та кінцевої точок.

s – Відстань між двома точками.

Ця формула враховує кривизну земної поверхні і є дуже точною для розрахунку відстаней, азимутів та геодезичних ліній на еліпсоїді. На рисунку 2.2 представлено програмну реалізацію формули Вінсенті

```
while (Math.abs(sigma - sigmaP) > 1e-12) {
    cos2SigmaM = Math.cos(2 * sigma1 + sigma);
    sinSigma = Math.sin(sigma);
    cosSigma = Math.cos(sigma);
    const deltaSigma =
        B *
        sinSigma *
        (cos2SigmaM +
            (B / 4) *
            (cosSigma * (-1 + 2 * cos2SigmaM * cos2SigmaM) -
                (B / 6) *
                cos2SigmaM *
                (-3 + 4 * sinSigma * sinSigma) *
                (-3 + 4 * cos2SigmaM * cos2SigmaM)));
    sigmaP = sigma;
    sigma = s / (b * A) + deltaSigma;
}
const tmp = sinU1 * sinSigma - cosU1 * cosSigma * cosAlpha1;
const lat2 = Math.atan2(
    sinU1 * cosSigma + cosU1 * sinSigma * cosAlpha1,
    (1 - f) * Math.sqrt(sinAlpha * sinAlpha + tmp * tmp)
);
const lambda = Math.atan2(
    sinSigma * sinAlpha1,
    cosU1 * cosSigma - sinU1 * sinSigma * cosAlpha1
);
const C = (f / 16) * cosSqAlpha * (4 + f * (4 - 3 * cosSqAlpha));
const lam =
    lambda -
    (1 - C) *
    f *
    sinAlpha *
    (sigma +
        C *
        sinSigma *
        (cos2SigmaM + C * cosSigma * (-1 + 2 * cos2SigmaM * cos2SigmaM)));
const lamFunc = lon1 + (lam * 180) / pi;
const lat2a = (lat2 * 180) / pi;
```

Рисунок 2.2 – Програмна реалізація алгоритму Вінсенті

Для виконання афінних перетворень на масиві географічних точок, тобто зміна положення точок на основі заданих параметрів, таких як зсув і трансформація (зміна масштабу чи обертання) було використано функцію aphine_update. Спочатку функція встановлює параметри трансформації, які можуть включати масштабування, обертання та інші операції. Потім вона приступає до обробки кожної точки в масиві, віднімаючи визначені значення зсуву з координат кожної точки, застосовуючи афінні перетворення до зміщених координат, і нарешті повертаючи зсув назад до координат кожної точки. Оновлені координати зберігаються назад у масив, який функція потім повертає. Цей

процес включає зміну місця розташування точок на карті, що може бути корисним для візуалізації даних, зміни масштабу або орієнтації об'єктів .

Для виконання афінних перетворень була використана формула зсуву

$$x' = x + \Delta x$$

$$y' = y + \Delta y$$

Де Δx та Δy коефіцієнт зсуву у двовимірному просторі точок x та y

Масштабування розрахуємо по формулі

$$x' = S_x * x$$

$$y' = S_y * y$$

При обертанні точки на певний кут навколо початку координат, нові координати визначаються за формулою

$$x' = x * cos(\theta) + y * sin(\theta)$$

$$y' = x * sin(\theta) + y * cos(\theta)$$

Кінцева позиція кожної точки після афінного перетворення розраховується за допомогою комбінації операцій масштабування, переносу, повороту та зсуву, що дозволяє здійснити точне позиціонування та орієнтацію елементів у просторі. Ці трансформації базуються на матричних обчисленнях, які використовують координати початкової точки та визначають її нове положення в результаті перетворень приклад використання цих формул представлено на рисунку 2.3.

```
aphine_update(latlngs, transform_x, transform_y, dx = 0, dy = 0) {
    this.set_transform_angel(transform_x, transform_y);
    console.log('matrix:');
    console.log(this.transform_matrix);

    let cur_points = latlngs[0];
    for (let i = 0, length = cur_points.length; i < length; i++) {
        let cur_point = cur_points[i];
        let center_lat = cur_point.lat - dx;
        let center_lng = cur_point.lng - dy;
        let point = this.update_point(center_lat, center_lng);

        cur_points[i].lat = point[0] + dx;
        cur_points[i].lng = point[1] + dy;
    }

    return cur_points;
}
```

Рисунок 2.3 – Функція афінних перетворень

На інтерактивній карті ці зміни можуть бути відображені в реальному часі, де користувачі можуть бачити, як змінюється площа постраждалої території в залежності від оновлень даних про погодні умови, які можуть бути інтегровані з системами моніторингу погоди, щоб автоматично отримувати дані про погодні умови та відповідно адаптувати візуалізацію постраждалих територій.

2.1.2 Інтерполяція

Інтерполяція точок відносно наявного масиву векторів вітру, отриманих з файлу grib.json, є методом, який можна використовувати для прогнозування умов

вітру в тих місцях, де безпосередньо інформація про вітер відсутня. Цей метод є особливо корисним у контексті створення інтерактивних карт, які відображають потенційні зони ризику від надзвичайних ситуацій, викликаних погодними умовами, такими як сильні вітри. Інтерполяція точок вітру відіграє ключову роль у цьому процесі, дозволяючи прогнозувати умови вітру в місцях, де безпосередні вимірювання відсутні.

Інтерполяція являє собою процес використання відомих даних про вітер (наприклад, швидкість та напрямок) з різних точок для оцінки умов вітру в інших місцях. Це робиться шляхом математичного аналізу існуючих даних та створення моделі, яка може передбачити умови вітру в точках, де безпосередні вимірювання недоступні. Для розуміння процесу інтерполяції необхідно пройтись по етапах процесу таких, як.

Збір даних: Спочатку збираються дані про вітер з файлу grib.json, який містить інформацію про вітрові умови у різних точках.

Вибір методу інтерполяції: Існує кілька методів інтерполяції, таких як лінійна, білінійна, сплайн-інтерполяція тощо. Вибір методу залежить від характеру даних та потрібної точності.

Створення прогнозної моделі: На основі методу білінійної інтерполяції створюється модель, яка може передбачити умови вітру в невиміряних точках. Для розрахунку інтерполяції було використано формулу

$$f(x, y) = g_{00} * (1 - x) * (1 - y) + g_{10} * x * (1 - y) + g_{01} * (1 - x) * * y + g_{11} * x * y$$

Де

$f(x, y)$ – Інтерпольоване значення в точці (x, y).

$g_{00}, g_{10}, g_{01}, g_{11}$ – Значення в чотирьох найближчих точках сітки.

x, y – Відносні координати точки інтерполяції в межах від 0 до 1.

Інтеграція з інтерактивною картою: Отримані прогнози інтегруються з інтерактивною картою, дозволяючи візуалізувати потенційні зони ризику від вітру.

Функція інтерполяції представлена на рисунку 2.4

```
var interpolate = function(λ, ϕ) {

    if(!grid) return null;

    var i = floorMod(λ - λ0, 360) / Δλ;
    var j = (ϕ0 - ϕ) / Δϕ;

    var fi = Math.floor(i), ci = fi + 1;
    var fj = Math.floor(j), cj = fj + 1;

    var row;
    if ((row = grid[fj])) {
        var g00 = row[fi];
        var g10 = row[ci];
        if (isValue(g00) && isValue(g10) && (row = grid[cj])) {
            var g01 = row[fi];
            var g11 = row[ci];
            if (isValue(g01) && isValue(g11)) {
                // All four points found, so interpolate the value.
                return builder.interpolate(i - fi, j - fj, g00, g10, g01, g11);
            }
        }
    }
    return null;
```

Рисунок 2.4 – Функція інтерполяції

Цей метод, застосовується для створення більш точних та інформативних інтерактивних карт, зокрема, пов'язані з сильними вітрами. Це дозволяє детально візуалізувати зони потенційного ризику, показувати шляхи переміщення вітрових фронтів та зони, що можуть постраждати найбільше. Такі карти можуть використовуватися для підготовки населення, організації евакуаційних заходів та

розподілу ресурсів рятувальних служб. Крім того, завдяки точній візуалізації, можна здійснювати аналіз даних у реальному часі, що допомагає приймати оперативні рішення та знижує ризики для життя та здоров'я людей у разі сильних вітрів.

2.1.3 Алгоритм обробки геоданих

GRIB-файли є форматом файлів, який використовується для зберігання метеорологічних даних. Вони складаються з декількох наборів даних, кожен з яких містить інформацію про певну метео-величину.

Набір даних складається з двох компонентів: заголовка та мапи значень. Заголовок містить інформацію про тип даних, дату та час, область, яку покривають дані, і інші. Карта значень містить фактичні значення даних для конкретної точки на мапі.

Для обчислення погоди достатньо трьох наборів даних: напрямку вітру за віссю Ox, напрямку вітру за віссю Oy та швидкості вітру.

Для побудови векторного масиву використовується наступний алгоритм:

```
for (var j = 0; j < nj; j++) {
  var row = [];
  for (var i = 0; i < ni; i++, p++) {
    row[i] = [uData[i], vData[i], temp.data[i]];
  }
  grid[j] = row;
}
```

Де:

j - координата широти

i - координата довготи

nj - довжина тривимірної мапи даних

ni - висота двовимірного масива даних

uData - масив значень напрямку за віссю Ox

vData - масив значень напрямку за віссю Oy

temp.data - масив значень (в нашому випадку - швидкість вітру)

Після чого, отримана матриця накладається на світову мапу.

2.2 Побудова інтерфейсу користувача

Інтерфейс користувача (UI) - це сукупність засобів, за допомогою яких користувач взаємодіє з веб-застосунком. Він є важливим компонентом будь-якої програми, оскільки від нього залежить, наскільки легко і зручно користувачеві буде ним користуватися.

UI повинен бути інтуїтивно зрозумілим для користувача. Необхідно реалізувати просту систему навігації, де можна буде швидко знайти всі необхідні функції, без необхідності читати довгі інструкції. Відвідувачі веб застосунку повинні легко виконувати необхідні дії, не витрачаючи зайвого часу. Щоб зробити застосунок конкурентоспроможним також важливо розробити сучасний та не перевантажений дизайн

Бібліотека AdminLTE - це відкритий фреймворк для створення UI веб-застосунків. Вона надає широкий спектр готових елементів інтерфейсу, таких як кнопки, текстові поля, меню, таблиці, тощо. Ці елементи мають сучасний дизайн і відповідають принципам інтуїтивної навігації.

Фреймворк також включає в себе набори стилів для різних типів пристроїв, що дозволяє забезпечити однаково якісний досвід користування незалежно від розміру екрана чи типу пристрою.

Бібліотека містить в собі набір мінімалістичних та чітко зрозумілих навігаційних елементів, має велику кількість шаблонів, що дозволяють швидко реалізувати

базову структуру веб порталу, без необхідності продумувати і оптимізовувати розміщення інформаційних блоків під різні типи пристроїв. Впроваджує великий набір бібліотек, які розширюють можливості базових HTML елементів, що допомагає спростити процес розробки та зробити інтерфейс користувача більш якісним

Висновки до розділу 2

У даному розділі були розглянуті методи розв'язання задачі. Окрім аналізу існуючих методів, була зосереджена увага на використанні афінних перетворень та білінійної інтерполяції. Ці техніки є ключовими для виконання розрахунків наслідків на інтерактивній карті, інтегрованій із розробленою системою. Це дозволяє ефективно обробляти та аналізувати дані. Також розглянуто метод створення користувацького інтерфейсу і обробки метеоданих.

3 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

В програмній реалізації методу візуалізації на інтерактивній карті територій потерпілих від наслідків представлено користувацький інтерфейс та серверна частина з базою даних.

3.1 Структура Бази даних

База даних має п’ять таблиць: ua, events та admins, emergency_types, emergencies, в яких зберігається інформація для роботи з системою. Далі наведено пояснення про структуру та дані, що зберігаються структуру представлено на рисунку 3.1.

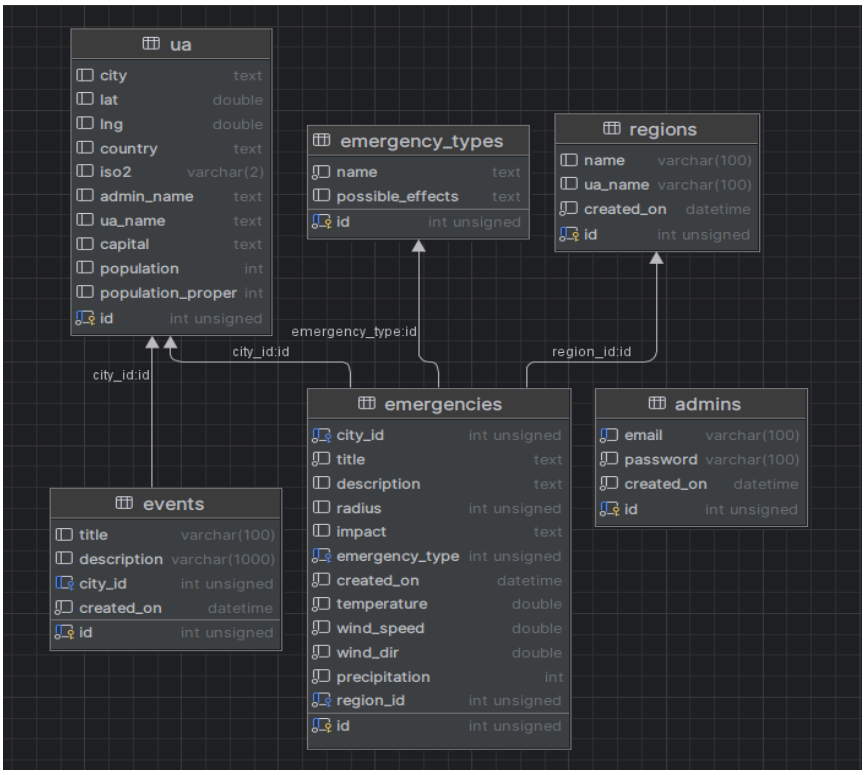


Рисунок 3.1 структура бази даних

Надалі наведено опис бази даних, та пояснено які дані зберігаються в кожній таблиці:

1. Таблиця emergency_types містить типи надзвичайних ситуацій за їхнім походженням та наслідками, такими як: техногенні, природні, тощо

Наявні поля: id, name та possible_effects.

Поле id виступає в якості первинного ключа.

Існує зовнішній ключ (показаний стрілкою) з поля emergency_type таблиці emergencies до цього поля id.

2. Таблиця ua містить назви населених пунктів, наявні події, координати, також міститься інформація про населення в регіоні.

Наявні поля: id, city, lat, lng, country, iso2, admin_name, ua_name, capital, population та population_proper.

Поле id виступає в якості первинного ключа.

Зображено два зовнішніх ключових відношення:

a. Стрілка веде від поля city_id таблиці events до поля id таблиці ua.

b. Стрілка веде від поля city_id таблиці emergencies до поля id таблиці ua.

3. Таблиця events містить інформацію про окрему подію, таку як: опис, заголовок, дата створення та місце події.

Містить поля: id, title, description, city_id та created_on.

Поле id виступає в якості первинного ключа.

Поле city_id має зовнішнє ключове відношення з таблицею ua.

4. Таблиця admins містить дані про операторів, задля безпеки, щоб тільки авторизовані користувачі могли вносити зміни до системи. Даними в цій таблиці є пароль, електронна адреса, та дата створення запису.

Містить поля: id, email, password та created_on.

Поле id виступає в якості первинного ключа.

Існує унікальне обмеження на поле email, що забезпечує відсутність дублікатів електронних адрес.

5. Таблиця emergencies містить інформацію про надзвичайну ситуацію, її тип, радіус ураження, наслідки, тип, також використовується для розрахунку подальших змін в площі та території ураження в подальшому.

Містить поля: id, city_id, title, description, radius, impact, emergency_type, temperature, wind_speed, wind_dir, precipitation та created_on.

Поле id виступає в якості первинного ключа.

Існують три зовнішніх ключових відношення:

a. Перше з'єднує поле emergency_type з полем id таблиці emergency_types.

b. Другез'єднує поле city_id з полем id таблиці ua.

c. Третє з’єднує поле region_id з полем id таблиці regions

6. Таблиця regions містить інформацію про області України, їх назву та транслітеровану назву. Використовується для коректної обробки подій, в областях

Поле id виступає в якості первинного ключа.

Поле city_id має зовнішнє ключове відношення з таблицею ua.

3.2 Система міграцій

Файли міграції бази даних - це важливий інструмент для розробників веб-проєктів, які використовують бази даних. Вони допомагають забезпечити, щоб база даних завжди перебувала в актуальному стані і щоб зміни до бази даних були контрольованими і відстежуваними.

Вони використовуються для для підтримки структури даних в актуальному стані. Наприклад, якщо необхідно додати нову функцію до свого веб-застосунку, яка потребує збереження чи отримання нової інформації від СУБД. Файл міграції дозволяє розробити та ініціалізувати ці зміни в контрольованому середовищі.

Загалом така система має наступні переваги:

1. Гарантує, що база даних завжди перебуває в актуальному стані. Це допомагає уникнути проблем, пов'язаних з несумісністю версій СУБД.

2. Зберігає історію змін до бази даних. Це дозволяє легко відстежувати, які зміни були внесені і коли.

3. Можна використовувати для повторення змін до бази даних на різних серверах. Це допомагає забезпечити однакову структуру СУБД для всіх наявних серверів чи при зміні хостинга.

3.3 Використання криптографії для покращення безпеки

Криптографія є фундаментальним інструментом для забезпечення безпеки в інформаційних системах, і її використання у процесі аутентифікації користувачів є критично важливим. Перевірка паролів користувачів — це не лише перший крок для встановлення довіри до особи, що намагається отримати доступ до системи, але й захист від несанкціонованого доступу. Сучасні системи захисту використовують хешування для перевірки паролів. Хешування — це односторонній криптографічний процес, який перетворює пароль у унікальний набір символів, відомий як хеш. Цей хеш зберігається в базі даних, а коли користувач вводить свій пароль, система хешує введення та порівнює його з збереженим хешем. Якщо обидва хеші співпадають, пароль вважається правильним. BCrypt є прикладом такого алгоритму хешування, який спеціально розроблений для захисту паролів. Він використовує сіль (випадково генеровані дані), щоб зробити кожен хеш унікальним навіть для однакових паролів, і включає механізм для уповільнення процесу хешування. Уповільнення є важливим, оскільки воно робить процес зламання паролів через вгадування чи за методом грубої сили (Brute Force) значно важчим та часозатратним. Важливим аспектом керування паролями є також блокування записів у базі даних під час їх оновлення, щоб запобігти одночасним змінам. Такий підхід гарантує, що кожен

пароль обробляється ізольовано, запобігаючи можливості перехресних атак або гонок за оновлення даних. Після успішного оновлення пароля, заблокований запис відпускається, дозволяючи системі знову взаємодіяти з ним процес обробки паролів за допомогою бібліотеки BCrypt показано на рисунку 3.2.

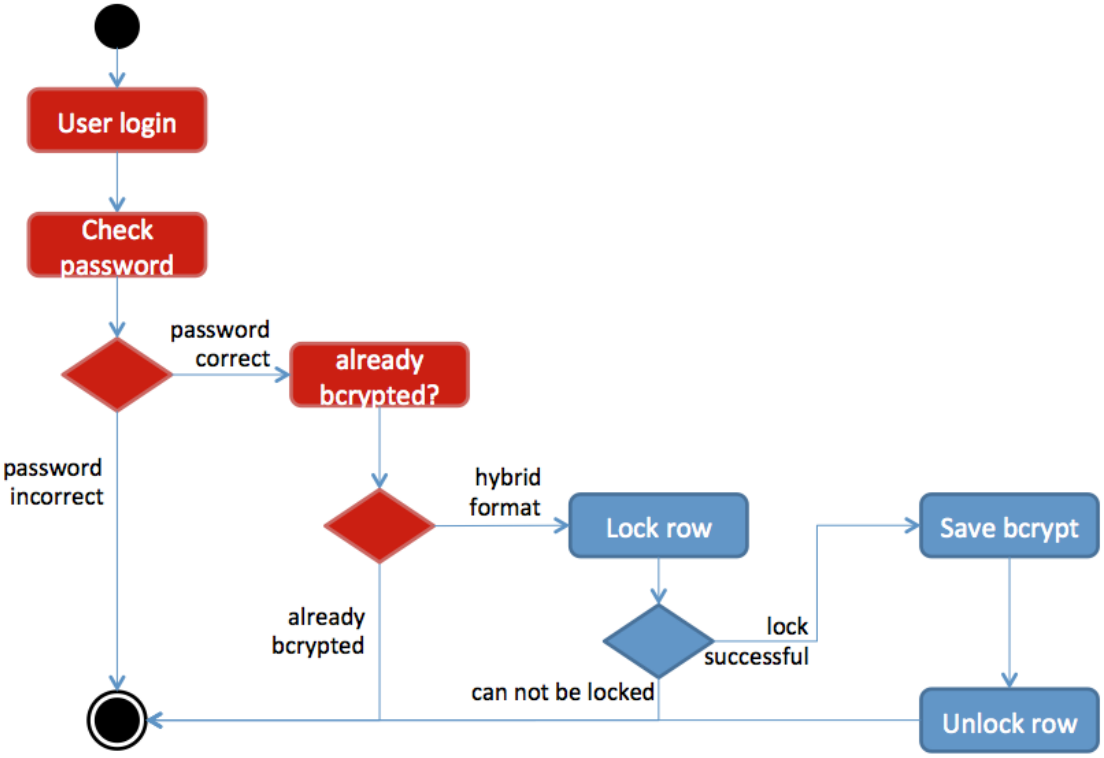


Рисунок 3.2 – Алгоритм роботи BCrypt

Застосування BCrypt у таких процесах є засобом превентивної безпеки. Воно не лише захищає від потенційних зламів. Замість того, щоб розглядати безпеку як однократну акцію, Необхідні постійні оновлення та адаптації безпекових механізмів для відповіді на зростаючі кіберзагрози та вдосконалення технологій.

3.4 Протоколи доступу до веб додатку

Протокол HTTP, розроблений для запитів гіпертекстових документів з серверів, є основою для зв'язку між веб-браузерами та серверами. Цей протокол, який працює на верхньому рівні стека TCP/IP, забезпечує передачу HTML-файлів, які браузер відображає разом з текстом, зображеннями, гіперпосиланнями та іншими ресурсами. В основі HTTP лежить модель взаємодії клієнт-сервер, де клієнти (наприклад, веб-браузери) ініціюють зв'язок із серверами та надсилають запити, на які сервери відповідають.

Під час веб-сеансу браузер встановлює TCP-з'єднання з HTTP-сервером, зазвичай використовуючи порт 80, хоча іноді можуть бути задіяні альтернативні порти, як-от 8080. Цей процес включає в себе обмін повідомленнями між клієнтом та сервером. Особливістю HTTP є його безстановий характер, що означає автоматичне розірвання з'єднання після завершення кожного запиту. Відмінно від протоколів, таких як FTP, HTTP не підтримує постійного з'єднання між запитами.

Коли виникають проблеми у процесі передачі даних через HTTP, такі як помилки користувача, неполадки в браузері чи сервері, помилки у коді веб-сторінок, або тимчасові збої в мережі, протокол відправляє відповідні коди стану помилок. Наприклад, помилки, що починаються з цифри 4, сигналізують про проблеми, пов'язані з обробкою запиту на стороні клієнта.

HTTPS стоїть окремо від свого незахищеного аналогу HTTP, надаючи додатковий шар безпеки завдяки використанню протоколів SSL або TLS. Ці протоколи створюють зашифрований канал (через порт 443), який захищає дані від перехоплення або несанкціонованого доступу під час передачі. Використовуючи шифрування з відкритим ключем, де сервер має пару ключів (відкритий та закритий) і обмінюється ними з клієнтом для створення зашифрованого з'єднання, HTTPS забезпечує значно більшу безпеку порівняно з HTTP. Коли користувач з'єднується з веб-сайтом, сервер демонструє володіння

приватним ключем, який відповідає публічному ключу в його SSL-сертифікат. Це підтвердження гарантує, що користувач з'єднується саме з тим сервером, який він очікує. Коли використовується HTTPS, всі HTTP-запити та відповіді шифруються за допомогою встановлених ключів сеансу. Це перетворює передані дані в безладний набір символів, що робить неможливим їх зчитування або інтерпретацію зловмисниками під час перехоплення.

Зашифрований протокол HTTPS має іншу важливу перевагу перед HTTP — швидкість завантаження веб-сторінок. Оскільки HTTPS вже вважається захищеним, веб-дані не потребують додаткового сканування або фільтрації. Це знижує обсяг переданих даних і сприяє швидшому завантаженню сторінок. Цей факт робить HTTPS не тільки безпечнішим, але й ефективнішим вибором для користувачів інтернету.

3.5 Рівні доступу користувачів та їхні ролі

В системі присутні дві сутності: звичайний користувач, та оператор іншими словами адміністратор, таке розподілення спрямоване на забезпечення цілісності та безпеки даних. Розподілення ролей між звичайними користувачами та адміністраторами (операторами) відіграє важливу роль у цьому процесі. Таке відокремлення розроблено для того, щоб запобігти несанкціонованому зміненню інформації з боку звичайних користувачів, обмежуючи їх можливості до перегляду або виконання обмеженого набору дій. Звичайні користувачі зазвичай мають доступ до базових функцій, таких як, перегляд існуючих подій та наслідків на інтерактивній карті, тоді як адміністратори наділені більш широкими повноваженнями. Адміністратори можуть налаштовувати системні параметри, оновлювати або видаляти записи в базах даних та виконувати інші критичні задачі, які вимагають вищого рівня довіри та відповідальності.

Встановлення чітких ролей і рівнів доступу також сприяє відповідальності та прозорості у веденні системних операцій. Реєстрація дій адміністраторів, включаючи зміни в базі даних, дозволяє проводити аудит та виявлення потенційних зловживань або помилок. Таким чином, стратегічне розмежування доступу та ролей в інформаційних системах не лише підвищує безпеку даних, але й ефективно розподіляє обов'язки між користувачами, забезпечуючи стабільне та контрольоване управління системою.

3.6 Архітектура програмної системи

В програмній реалізації системи методів відображення на інтерактивній карті територій постраждалих від наслідків можна виділити методи розрахунку наслідків та зміни зони ураження територій, базу даних територій, подій і наслідків та веб інтерфейс для роботи з системою.

3.6.1 Діаграма Класів

На рисунку 3.3 представлено діаграму класів системи, їхню взаємодію між собою.

Надалі буде описано кожну частину діаграми класів, розказано про її призначення та основні функції, будуть визначені атрибути, які класи використовують, їхні методи та відносини з іншими класами в системі. Окрім того, будуть розглянуті зв'язки між класами, що дозволяє зрозуміти структуру системи та механізми взаємодії компонентів.

Детальний опис кожного класу та його компонентів на діаграмі дозволяє глибше зрозуміти логіку системи і відіграють ключову роль у подальшій реалізації та тестуванні системи. Усі ці елементи разом формують комплексний

опис діаграми класів, який є критично важливим для розуміння та подальшої розробки системи.

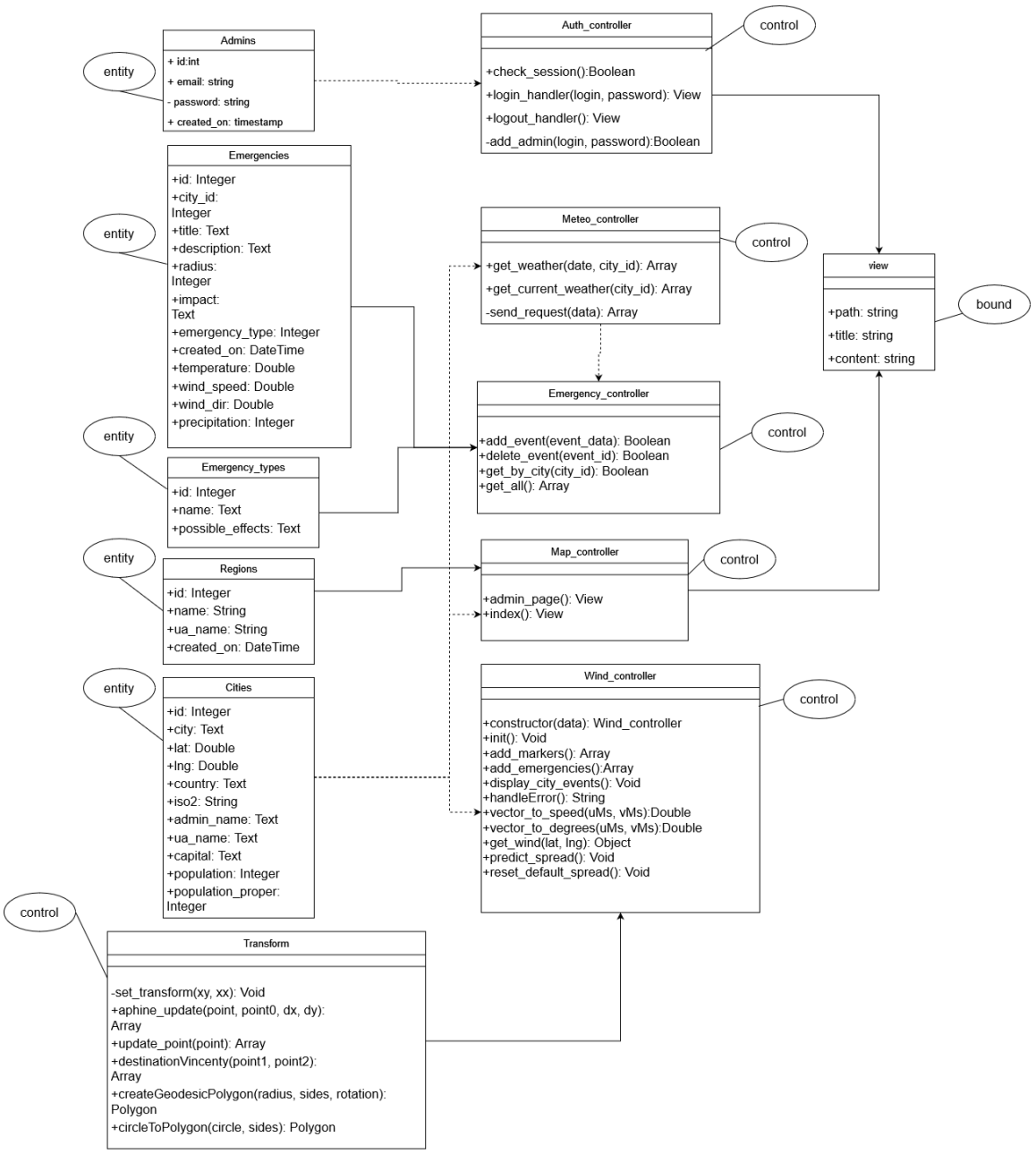


Рисунок 3.3 – Діаграма класів

Admins - сутність, яка представляє користувачів-адміністраторів системи. Клас має такі атрибути:

id - унікальний ідентифікатор користувача

email - адреса електронної пошти користувача

password - пароль користувача

created_on - дата створення облікового запису користувача

Emergencies - сутність, яка представляє надзвичайні події. Клас має такі атрибути:

id - унікальний ідентифікатор надзвичайної події

title - заголовок надзвичайної події

description - опис надзвичайної події

radius - радіус поширення надзвичайної події

impact - вплив надзвичайної події

emergency_type - тип надзвичайної події

city_id - ідентифікатор міста, в якому відбулася надзвичайна подія

created_on - дата створення надзвичайної події

EmergencyTypes - сутність, яка представляє типи надзвичайних подій. Клас має такі атрибути:

id - унікальний ідентифікатор типу надзвичайної події

name - назва типу надзвичайної події

possible_effects - можливі наслідки надзвичайної події

Regions - сутність, яка представляє області. Клас має такі атрибути:

id - унікальний ідентифікатор регіону

name - назва регіону

ua_name - назва регіону українською мовою

created_on - дата створення регіону

Cities - сутність, яка представляє міста. Клас має такі атрибути:

id - унікальний ідентифікатор міста

city - назва міста

lat - широта міста

lng - довгота міста

country - країна, в якій розташоване місто

iso2 - код ISO 2 для країни, в якій розташоване місто

admin_name - назва адміністративного центру міста

ua_name - назва міста українською мовою

population - населення міста

MeteoController - клас, який відповідає за взаємодію з API погодних служб. Клас має такі методи:

get_weather(date, city_id) - повертає дані про погоду для заданої дати та міста

get_current_weather(city_id) - повертає поточні дані про погоду для заданого міста

EmergencyController - клас, який відповідає за взаємодію з базою даних надзвичайних подій. Клас має такі методи:

add_event(event_data) - додає нову надзвичайну подію до бази даних

delete_event(event_id) - видаляє надзвичайну подію з бази даних

get_by_city(city_id) - повертає список надзвичайних подій для заданого міста

get_all() - повертає список усіх надзвичайних подій

WindController - контроллер, який відповідає за прогнозування поширення надзвичайних подій. Клас має такі методи:

constructor(data) - використовується для ініціалізації класа, та подальшої роботи з ним

int() - ініціалізує клас, підготовлює початкові дані та відображає їх на мапі

add_markers() - Додає всі необхідні маркери на мапу, повертає масив маркерів для подальшої роботи над ними

add_emergencies() - відображає на мапі всі наявні надзвичайні події, повертає масив подій для подальшої роботи над ними

display_city_events() - відображає всі події прив’язані до конкретного міста

handleError() - функція відображення помилок

vector_to_speed(uMs, vMs) - перетворює вектор вітру в швидкість.

vector_to_degrees(uMs, vMs) - перетворює вектор вітру в напрямок.

get_wind(lat, lng) - повертає дані про вітер для даної широти та довготи.

predict_spread() - прогнозує поширення надзвичайної події на основі даних про вітер

reset_default_spread() - видаляє спрогнозовані дані, вимальовує зонам вигляд за замовчуванням

Transform - контроллер, який реалізує операції трансформації точок. Клас має наступні методи та функції:

set_transform(xy, xx): встановлює матрицю трансформації.

aphine_update(point, point0, dx, dy): Обновлює точку point за допомогою афінної трансформації, заданої точками point0, dx і dy.

update_point(point): Оновлює точку point за допомогою поточної трансформації об'єкта.

destinationVincenty(point1, point2): Розраховує точку призначення на сфері за заданими точками point1 і point2.

createGeodesicPolygon(radius, sides, rotation): Створює геодезичний багатокутник з заданим радіусом, кількістю сторін і поворотом.

circleToPolygon(circle, sides): Створює багатокутник, що представляє круг з заданим радіусом і кількістю сторін

MapController - контроллер, який відповідає за відображення мапи надзвичайних подій. Клас має такі методи:

admin_page() - відображає сторінку адміністратора

index() - відображає основну сторінку системи

3.6.2 Діаграма прецедентів

На рисунку 3.4 зображено схему випадків використання, яка інтуїтивно зрозуміла і підходить для представлення функціональності системи як у словесній, так і у візуальній формі. Основна мета цього методу полягає в тому, щоб надати проектантам засіб для організації та планування системи, виходячи з

вимог та сподівань користувачів. Таке зображення сприяє чіткому уявленню про те, як система буде взаємодіяти з кінцевими користувачами, та демонструє всі можливі варіанти взаємодії з системою.

Використання ролей дозволяє диференціювати повноваження та обов'язки між різними типами користувачів. Наприклад, в системі ідентифіковані ролі звичайного користувача та адміністратора, кожна з яких має свій унікальний набір доступів та функцій. Звичайний користувач, будучи незареєстрованою особою, має обмежений доступ до системи, зазвичай до функцій перегляду, в той час як адміністратор, що має вищий рівень доступу, може займатися управлінням даними, включаючи додавання, видалення та редагування інформації про події.

Це розмежування доступу не тільки забезпечує безпеку та цілісність системи, але й сприяє ефективності її використання. Різні рівні доступу дозволяють адміністраторам системи здійснювати складні операції, такі як оновлення та модифікація даних, тоді як звичайні користувачі можуть переглядати інформацію та користуватися базовими функціями. Така ієрархія доступу також допомагає в підтримці порядку та організації в системі, водночас знижуючи ризик випадкових або зловмисних дій, які можуть негативно вплинути на функціонування системи. Крім того, ретельно сплановані сценарії використання сприяють тому, що система стає більш адаптивною до потреб різних груп користувачів, забезпечуючи гнучкість та масштабованість управління даними.

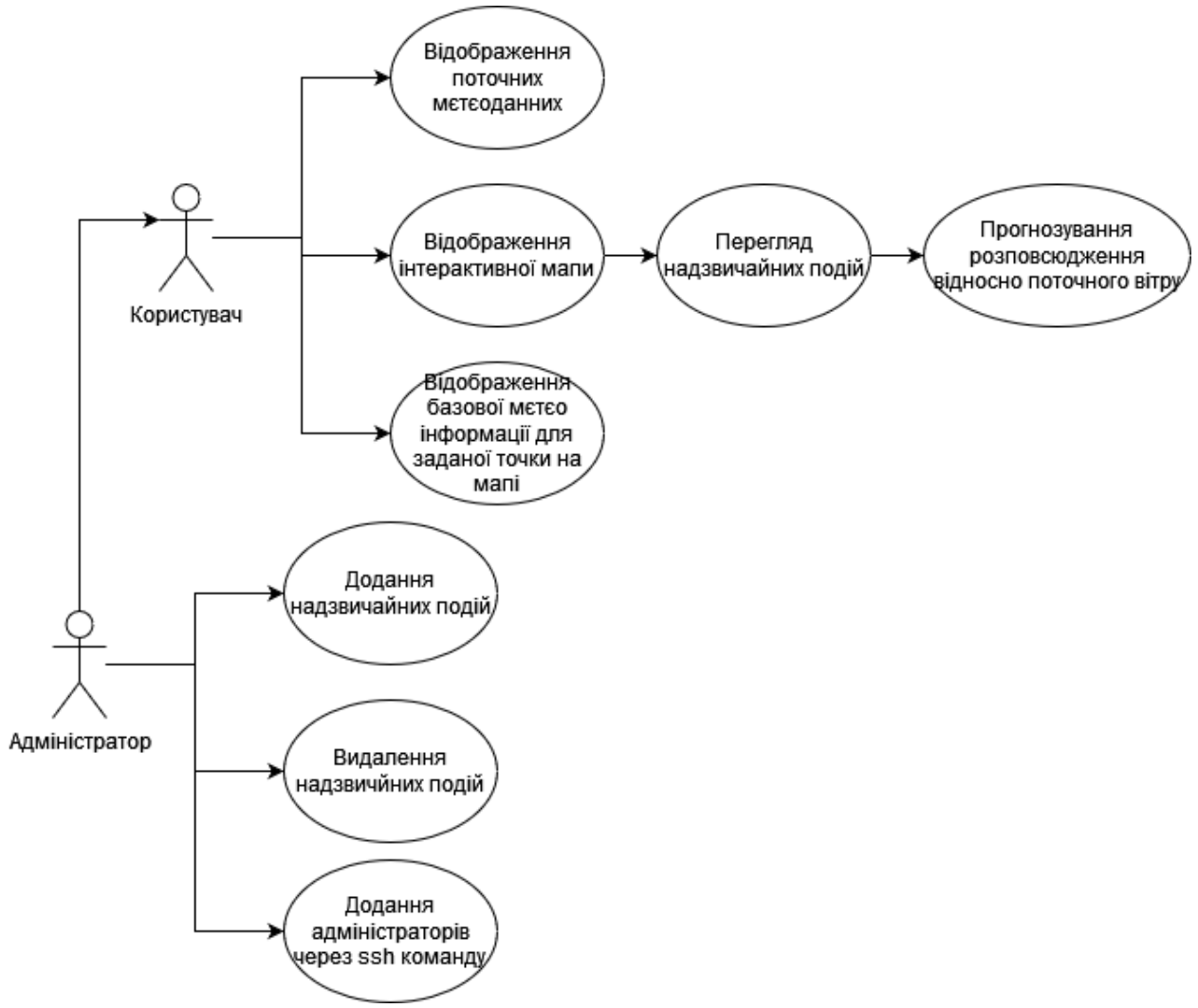


Рисунок 3.4 – Діаграма прецедентів

3.6.3 Діаграма BPMN

На даній BPMN діаграмі зображена базова логіка застосунку при вході на головну сторінку сайта рисунок 3.5.

Процес відображення головної сторінки починається з того, що користувач заходить на сайт системи. Користувач автоматично валідується системою або як Адміністратор, якщо збігаються зашифрований ключ сесії користувача, та оригінальна копія збережена для одного з профілів або як звичайний відвідувач.

При привілейованому доступі відображається сторінка адміністратора з можливістю додавання надзвичайних подій. Інакше - відображається Звичайна сторінка, де можливо тільки переглядати наявні події.

Паралельно з цим система надсилає два запити. Перший - до користувацького серверу, щоб отримати всі наявні надзвичайні події та інформацію про них. Другий - до метео серверу, щоб отримати інформацію про поточну погоду. Якщо дані так і не були знайдені - користувачу повинна відображатись помилка.

За бажанням користувача можна відобразити повну інформації про надзвичайну подію, клікнувши по ній на мапі

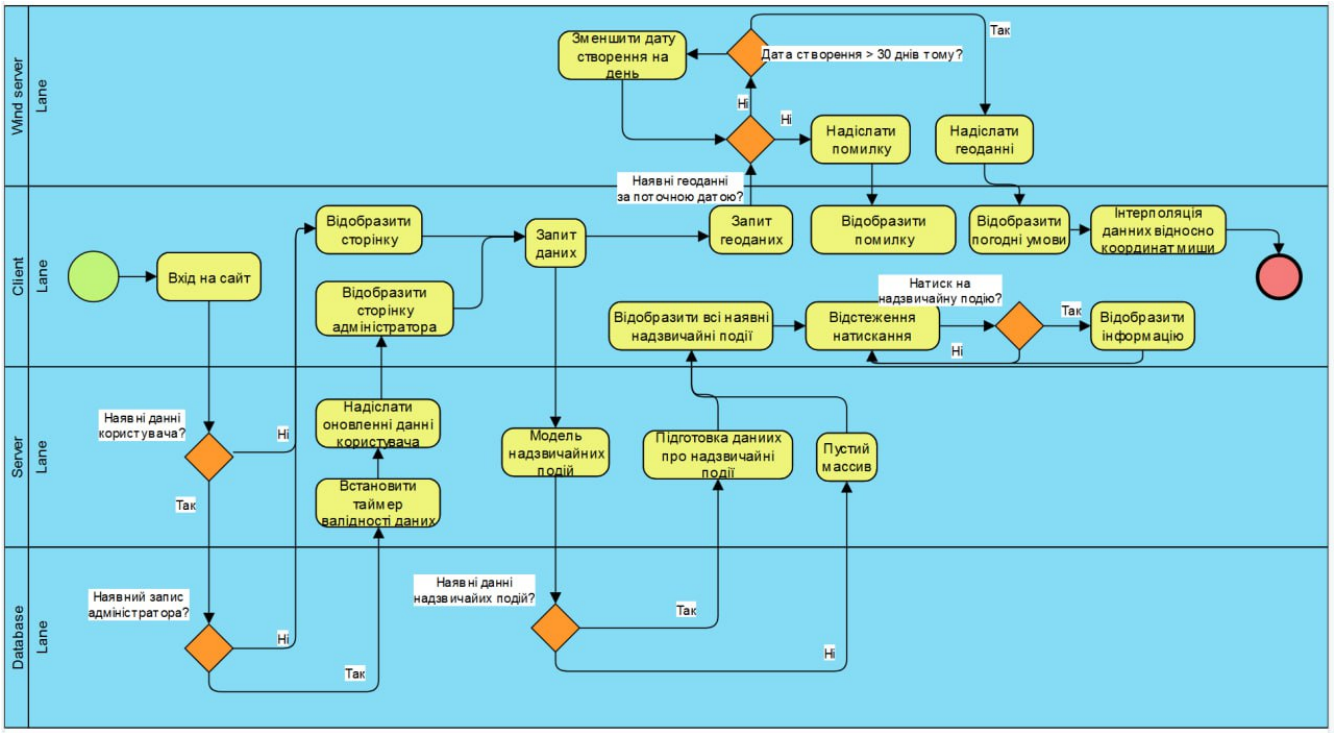


Рисунок 3.5 – Діаграма BPMN входу в систему

Висновки до розділу 3

В цьому розділі ознайомились з описом програмної реалізації, діаграмою прецедентів, діаграму класів, та діаграму послідовності дій при роботі з додатком. В системі присутні дві сутності: звичайний користувач, та оператор іншими словами адміністратор, таке розподілення спрямоване на забезпечення цілісності та безпеки даних. Розподілення ролей між звичайними користувачами та адміністраторами (операторами) відіграє важливу роль у цьому процесі. Таке відокремлення розроблено для того, щоб запобігти несанкціонованому зміненню інформації з боку звичайних користувачів

4 РОБОТА КОРИСТУВАЧА З ПРОГРАМОЮ

У цьому розділі буде розглянуто взаємодію кінцевих користувачів із програмою, а саме: як отримати доступ, процес реєстрації і безпосередню роботу всередині системи. Також буде наведено вимоги системи до середовища, а також необхідні засоби для запуску системи в цілому

4.1 Загальні характеристики застосунку

Програмний продукт складається з двох ключових сервісів, кожен з яких виконує специфічні функції:

–Користувацький портал: Цей сервіс має дві основні компоненти:

–Серверна частина: Відповідає за обробку запитів та зворотній зв'язок з клієнтською частиною.

–Клієнтська (браузерна) частина: Інтерфейс, через який користувачі взаємодіють з порталом.

–Технічний сервер: Призначений для автоматичного завантаження, обробки та збереження даних з відкритих геометричних джерел. Він може функціонувати незалежно від користувацького порталу. Обидва сервіси мають гнучку архітектуру, що дозволяє їх розгортання на будь-якій хмарній платформі. Проте, для цієї конкретної роботи сервіси були встановлені на локальній машині.

Що стосується зберігання даних:

Реляційна база даних MySQL використовується для зберігання інформації про надзвичайні події.

Погодні дані збираються двома способами:

Адміністрація додає їх вручну під час реєстрації надзвичайної події.

Технічний сервіс автоматично запитує ці дані щодня з бази даних NOMADS.

Бінарні дані погоди в форматі GRIB перетворюються в більш зручний формат JSON, після чого користувацький портал може їх отримати та відобразити.

Для проектування інформації на мапу використовуються наступні бібліотеки:

Leaflet: Популярна JavaScript бібліотека для мобільно-дружніх інтерактивних карт. WindJS-server: Сервіс для інтеграції вітрових і погодних даних на карту.

Ці бібліотеки дозволяють динамічно відображати геопросторові дані, забезпечуючи користувачам інтуїтивно зрозумілий та ефективний інтерфейс для візуалізації складних даних про надзвичайні події та погодні умови. Документацію до цих бібліотек можна знайти на їх офіційних веб сайтах.

4.2 Вимоги до обчислювальної техніки

Для ефективної роботи з веб-додатком у браузері, система користувача має відповідати таким мінімальним технічним вимогам:

Оперативна пам'ять : мінімум 1024 МБ для користувачів Windows 7, рекомендовано 4096 МБ для користувачів Windows 11 для оптимальної продуктивності.

Операційні системи: Підтримка сучасних версій Windows (7, 10, 11), Linux або MacOS, що забезпечує широку сумісність з різними платформами.

Процесор: Мінімально необхідний Intel Pentium 4 або новіша версія.

Крім апаратних вимог, необхідно встановити інтернет-браузер:

Остання версія Google Chrome, Opera,Firefox або Safari для забезпечення найкращої сумісності та безпеки під час використання веб-додатку.

Також система сумісна майже з усіма мобільними пристроями на операційній системі Android та IOS.

Також варто обговорити вимоги до сервера для безперервної підтримки сервісів і можливості доступу до даних для користувачів

Для забезпечення надійної роботи серверу, який відповідає за зберігання даних, відмальовку карт та обробку метеоданих, необхідно врахувати наступні системні вимоги до обчислювальної техніки: Процесор Сервер має бути оснащений багато ядерним процесором високої продуктивності, оптимально Xeon або аналогічним, з підтримкою віртуалізації та паралельної обробки даних для ефективної відмальовки карт та аналізу метеоданих. Оперативна пам'ять Рекомендується мінімум 32 ГБ RAM, щоб можна було ефективно обробляти великі об'єми даних, зокрема для зберігання та швидкого доступу до метеоданих та картографічної інформації. Система зберігання даних Використання SSD (твердотільних накопичувачів) для операційної системи та програм, а також RAID-масивів для забезпечення високої швидкості читання/запису та резервного копіювання даних. Мережеві можливості Гігабітний Ethernet-інтерфейс для швидкого обміну даними між сервером та іншими компонентами мережі, а також для забезпечення доступу користувачів до серверу без затримок. Відеокарта Високопродуктивна відеокарта, сумісна з сервером, з підтримкою GPU-прискорення, що важливо для швидкої відмальовки карт та візуалізації даних. Система безперебійного живлення (UPS) для захисту від перебоїв з електроенергією та забезпечення безперебійної роботи серверу. Система охолодження Ефективна система охолодження для підтримки оптимальної температури роботи обладнання, оскільки висока продуктивність може призвести до значного тепловиділення. Програмне забезпечення Належне серверне програмне забезпечення для управління базами даних, віртуалізації, та забезпечення безпеки, включаючи захист від шкідливих програм та

несанкціонованого доступу. Сервіс та підтримка сервісних договорів для швидкого вирішення будь-яких технічних проблем, що можуть виникнути, та для забезпечення постійного обслуговування та оновлення обладнання.Забезпечення відповідності цим технічним вимогам допоможе створити міцну основу для серверу, котрий буде відповідати за важливі задачі відмальовки інтерактивних карт та обробки складних метеоданих.

4.3 Інтерфейс користувача

Користуватський інтерфейс включає в себе інтерактивну карту подій і наслідків, а також вікно з актуальними подіями приклад на рисунку 4.1. Неавторизовані користувачі не мають можливості додавати або видаляти події.

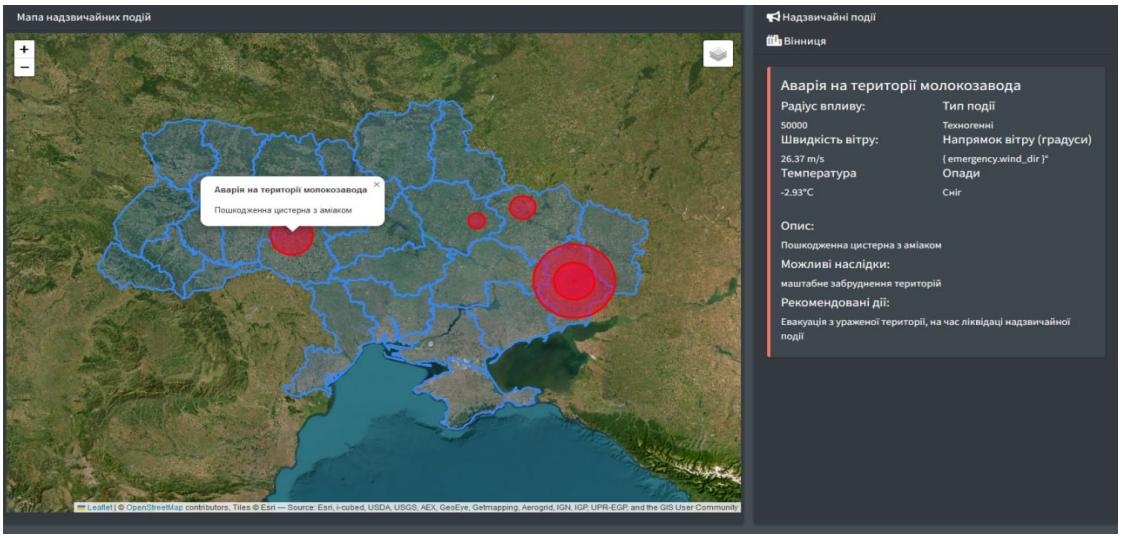


Рисунок 4.1 Сторінка неавторизованого користувача

Для редагування подій створено сторінку адміністратора, яка містить інструменти для додавання, видалення та редагування подій. Щоб отримати права адміністратора, потрібно зареєструватися і отримати пароль для

подальшого входу. Вхід в систему відбувається на сторінці панелі адміністратора як показано на рисунку 4.2

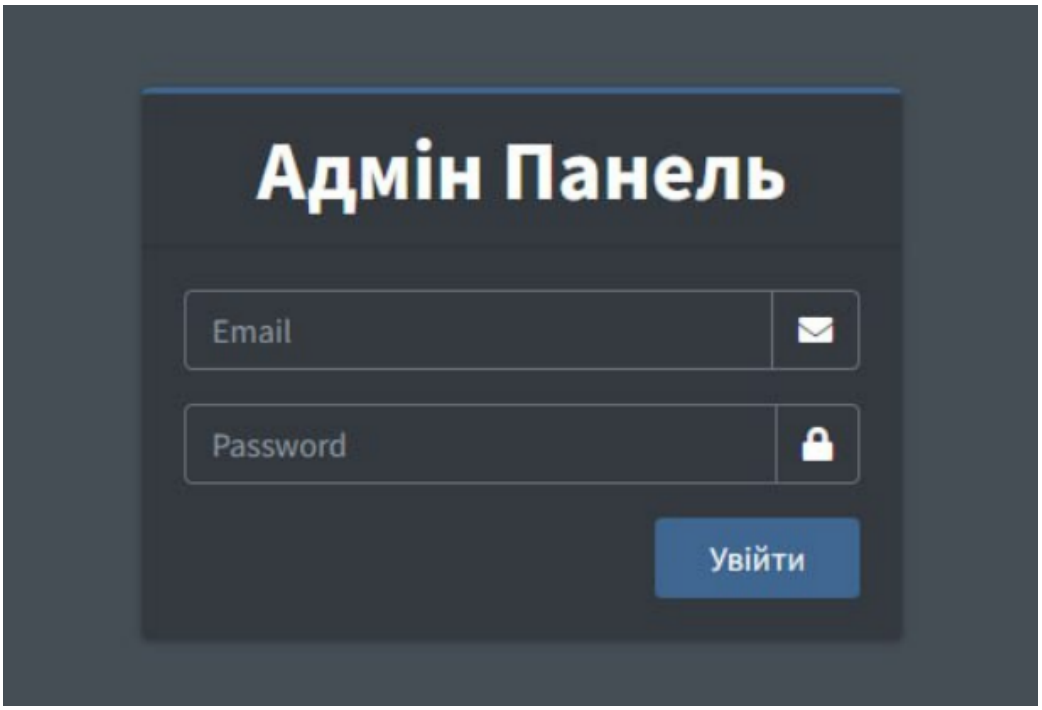


Рисунок 4.2 Сторінка входу для адміністратора

Для заповнення даних про подію, оператор має вибрати населений пункт, додати заголовок, тпи події та зону ураження, яка вимірюється в метрах, також необхідно описати, що відбувається, оголосити про можливі наслідки для населення та попередити про подальші дії, наприклад про евакуацію населення або про рекомендацію залишитись вдома та зачинити вікна, приклад бланку створення події показано на рисунку 4.3

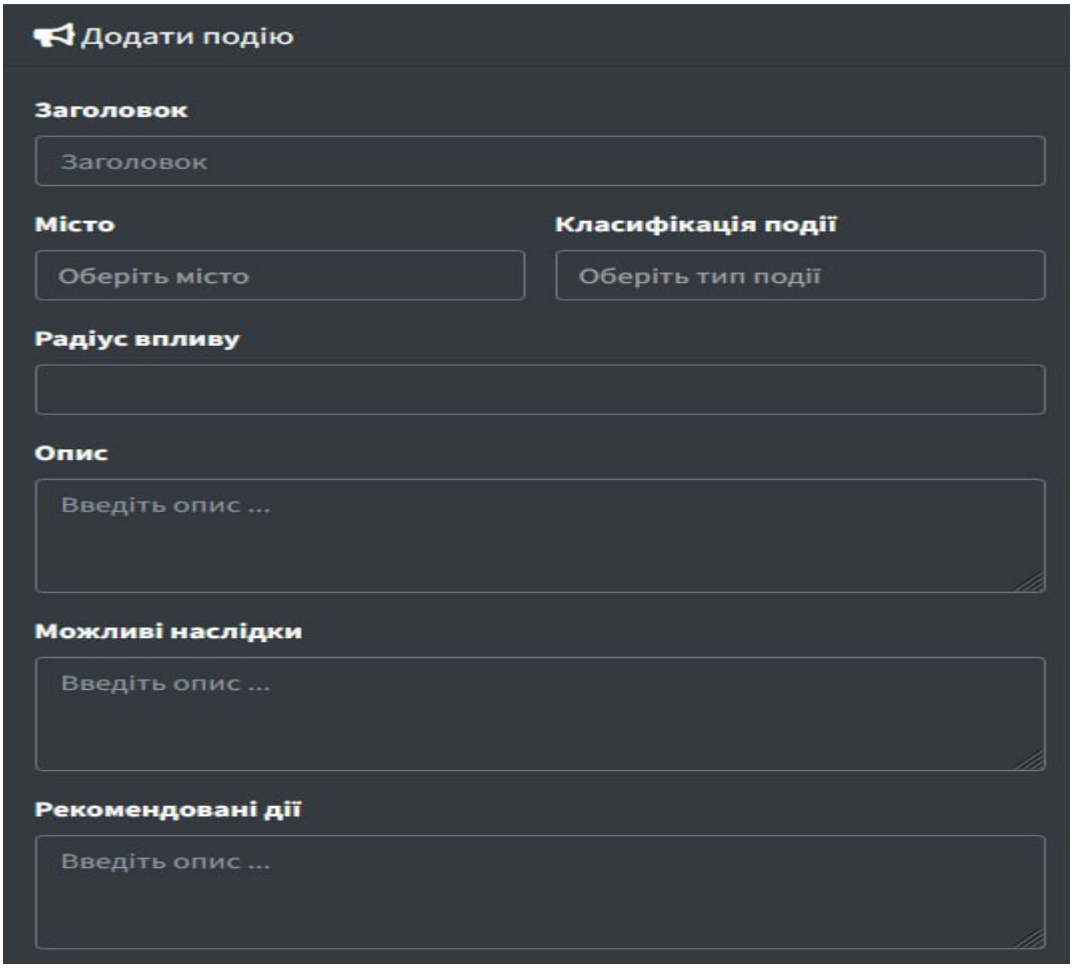


Рисунок 4.3 – Бланк заповнення події

Також для індивідуального користувача доступна можливість прогнозування розповсюдження надзвичайної ситуації з уранком погодних умов, для цього потрібно обрати пункт прогнозування розповсюдження в контекстному меню, та вибрати по яким критеріям прогнозувати, приклади прогнозування показані на рисунках 4.4 та 4.5. Дана можливість доступна всім користувача системи та націлена на впровадження дій щодо попередження подальших наслідків як для населення так для держави, з метою збереження найбільшої кількості життів та майна.

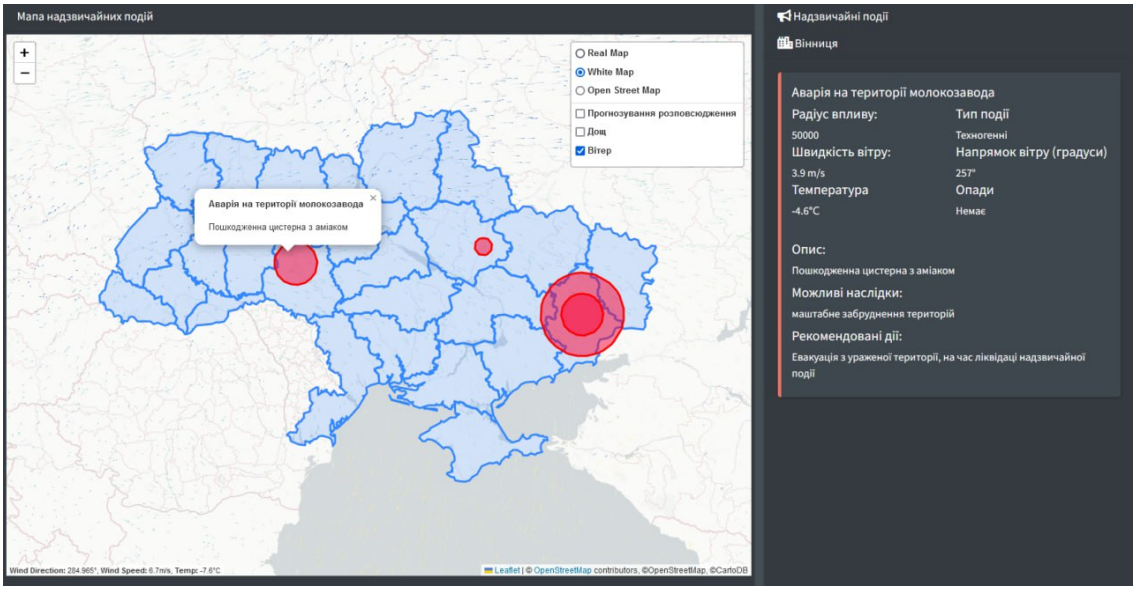


Рисунок 4.4 – Події без прогнозування розповсюдження

Якщо користувач обрав за критерій розрахунку вітер, який в даний момент часу та в даній точці має швидкість та напрям виходячи з геоданих, то зона ураження змінить форму і надасть нову площу.

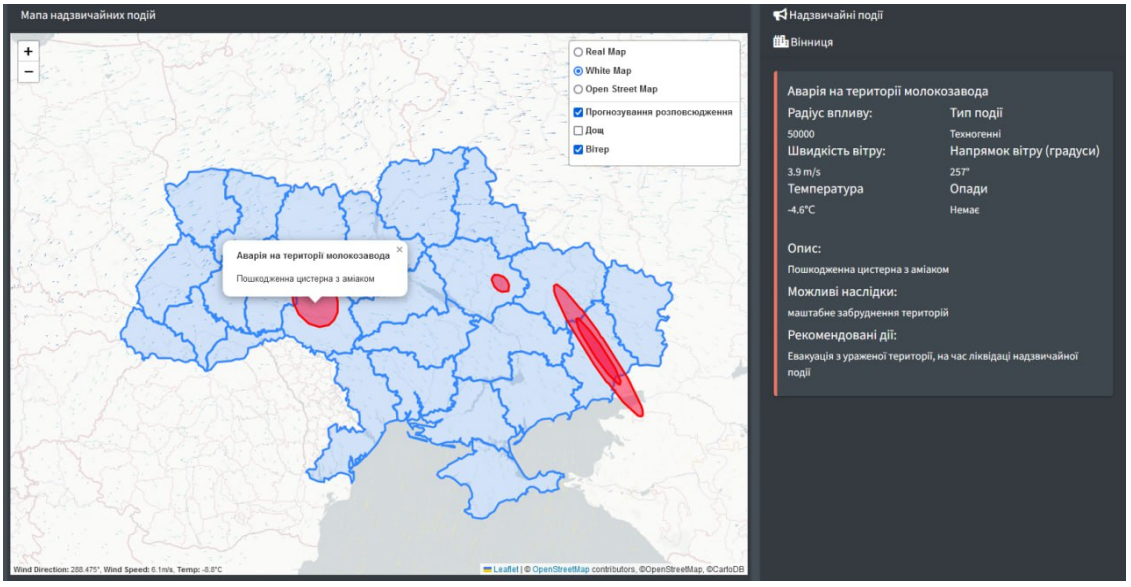


Рисунок 4.5 – Події після прогнозування розповсюдження

Висновки до розділу 4

В даному розділі розглянуто взаємодію користувача з системою візуалізації на інтерактивній карті, що призначена для демонстрації територій, постраждалих від різного роду наслідків. Значну увагу приділено інтерфейсу, що дозволяє користувачам переглядати карту, переглядати деталі подій та аналізувати статистику, адміністратору, взаємодіяти з базою даних.

5 РОЗРОБКА СТАРТАП ПРОЕКТУ

Концепція стартапу зосереджена на розробці інноваційної системи візуалізації для інтерактивних карт, яка спеціалізується на відображенні територій, уражених різними катастрофічними подіями. Така система надасть змогу порівнювати ефективність різних візуалізаційних методик у контексті зображення наслідків цих подій. Цей інструмент стане неоціненним для аналітики візуалізації, а також допоможе в ідентифікації зон ураження, що в свою чергу поліпшить стратегії реагування на надзвичайні ситуації та оптимізацію ресурсів для допомоги постраждалим територіям.

5.1 Опис ідеї стартап-проекту

Проаналізовано зміст ідеї, її можливі напрямки застосування, відмінність запропонованої ідея від існуючих аналогів, а також основні переваги, які може отримати користувач системи. Результати аналізу представлені у таблиці 5.1.

Таблиця 5.1 – Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Створення системи інтерактивної карти територій постраждалих від наслідків	Доступ до даних про надзвичайні події, внесення в реєстр свідчення про нові події та редагування наявних	Знаходження інформації про надзвичайні події в одному місці
	Аналіз наслідків та розрахунок території, яка була уражена внаслідок катастрофи	Автоматизація розрахунків наслідків, можливість прогнозування подальших дій для ліквідації ситуації або розробити план евакуації

На ринку існують системи з подібною тематикою, але їх функціонал часто обмежується лише виявленням та повідомленням про події, без можливості прогнозування їх наслідків. Це суттєво обмежує їх практичність, оскільки визначення наявності події без оцінки його потенційного впливу не надає користувачам повного розуміння ситуації. Крім того, багато з цих систем характеризуються надмірною складністю та громіздкістю, що робить їх менш зручними у використанні та інтеграції в існуючі процеси. Це створює потребу в більш ефективних та інтуїтивних рішеннях, здатних не лише повідомляти про події, але й аналізувати їх потенційні наслідки, дозволяючи користувачам вживати відповідних заходів заздалегідь.

Проведено аналіз потенційних техніко-економічних переваг ідеї порівняно з пропозиціями конкурентів. Результат аналізу у таблиці 5.2.

Таблиця 5.2 – Визначення характеристик ідеї проекту

Техніко-економічні характеристики ідеї	Продукція конкурентів			Слабкі (W), нейтральні (N) та сильні (S) сторони		
Назва продукту	Розроблена система	Google Disaster Map	European Forest Fire Information System	W	N	S
Операційна система та версії	Кроссплатформенна	Кроссплатформенна	Кроссплатформенна		+	
Ціна	безкоштовна	безкоштовна	безкоштовна		+	
Наявність зручного інтерфейсу для роботи з системою	Так	Ні	Ні			+

Кінець таблиці 5.2

Можливість порівнювати ефективність реалізованих алгоритмів	Так	Ні	Ні			+

Розроблена система протестована і працює на будь-якій операційній системі, та на мобільних гаджетах зі встановленою операційною системою Android або IOS. Такоє перевагою системи є мінімалістичний інтерфейс без зайвих елементів керування, що полегшує застосування додатку для нових користувачів, для подальшого охоплення аудиторії на старті.

5.2 Технологічний аудит ідеї проекту

Для проведення технічного аудиту ідеї проекту, потрібно провести аудит технології, за допомогою якої можна реалізувати ідею проекту, визначити чи доступні ці технології, та чи потребують вони доопрацювання [30]. Результат представлений у таблиці 5.3.

Таблиця 5.3 – Технологічна здійсненність ідеї проекту

Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
Розробка веб інтерфейсу	Python	+	Доступно (безкоштовне)
Розробка серверу збору геоданих	NodeJS server	+	Доступне (безкоштовне)

Кінець таблиці 5.3

Розробка алгоритму обробки геоданих	JavaScript	+	Доступно (безкоштовне)
Створення мапи	Leaflet	+	Доступно (безкоштовне)
Створення бази даних	MySQL	+	Доступно (безкоштовне)
Отримання метеоданних	NOMADS	+	Доступно (безкоштовне)
Отримання точних погодних умов	open-meteo	+	Доступно (безкоштовне)
Обробка GRIB файлів	Grib2Json	+	Доступно (безкоштовне)
Проектування вітру на мапу	wind-js-leaflet	+	Доступно (безкоштовне)

Обрані технології для реалізації проекту – це середовище розробки Microsoft Visual Studio Code, платформа розробки веб-інтерфейсу на мові програмування Python, база даних була розроблена інструментами СУБД MySQL.

5.3. Аналіз ринкових можливостей запуску стартап-проекту

Визначення ринкових можливостей, які можна використати під час ринкового впровадження проекту, та ринкових загроз, які можуть перешкодити

реалізації проекту, дозволяє планувати напрями розвитку проекту з урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів-конкурентів . Результати аналізу попиту наведено в таблиці 5.4.

Таблиця 5.4 – Попередня характеристика потенційного ринку стартап-проекту

Показники стану ринку	**Характеристика**
Загальна потреба в продукції	Необхідна
Можливі річні обсяги випуску в натуральних показниках	До 50 копій
Річні обсяги випуску в вартісних показниках	0 – 10000 $
Динаміка ринку (якісна оцінка)	Зростає
Наявність обмежень для входу	Немає
Специфічні вимоги до стандартизації та сертифікації	Немає
Середня норма рентабельності в галузі (або по ринку)	70%

За попередньою оцінкою існуючий ринок здається достатньо привабливим для входження. При аналізу актуальності системи було виявлено що на дослідження методів ізоморфізму зростає попит.

В таблиці 5.5 наведено характеристику потенційних клієнтів стартап-проекту.

Після визначення потенційних груп клієнтів необхідно провести аналіз ринкового середовища: скласти таблиці факторів, що сприяють ринковому впровадженню проекту, та факторів, що йому перешкоджають.

В таблиці 5.6 наведено фактори загроз, які перешкоджають ринковому впровадженню проекту.

В таблиці 5.7 наведено фактори можливостей, які сприяють ринковому впровадженню проекту.

Таблиця 5.5 – Характеристика потенційних клієнтів стартап-проекту

Потреба, що формує ринок	Цільова аудиторія	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
Потреба в обізнаності про наявні надзвичайні події, та їх наслідків та планування і координація дій для подолання наслідків ліз	Відомства надзвичайних ситуацій, громадські організації, компанії, що займаються плануванням безпеки, індивідуальні користувачі	Різні потреби в інформації про надзвичайні події, різні стратегії реагування та планування, різні вимоги до точності та швидкості інформації	Точна та своєчасна інформація про надзвичайні події, ефективні аналітичні інструменти для аналізу наслідків, простота у координації та виконанні планів реагування

Таблиця 5.6 – Фактори загроз

Фактор	Зміст загрози	Можлива реакція компанії
Поява конкурентів	Ризик з'явлення конкурентів із кращими рішеннями у сфері управління надзвичайними подіями.	Розвиток інноваційних підходів до аналізу та реагування на надзвичайні події, удосконалення власних інструментів.
Економічний спад	Зниження попиту на послуги управління надзвичайними подіями через економічні фактори.	Реорганізація цінової політики, пошук нових цільових аудиторій, адаптація послуг до економічних умов.
Зниження репутації компанії	Ризик втрати довіри клієнтів через ефективніші пропозиції конкурентів, або	Посилення маркетингових та PR-зусиль, підвищення якості обслуговування, акцент на відмінності та переваги своїх

Кінець таблиці 5.6

	проблеми з безпековими рішеннями, або витік даних.	послуг, посилення інформаційної безпеки.

Таблиця 5.7 – Фактори можливостей

Фактор	Зміст загрози	Можлива реакція компанії
Значна кількість конкурентів	Наявність значної кількості конкурентів на ринку інтерактивних карт свідчить про високий попит і важливість таких технологій	Для успішного розповсюдження продукту, компанії потрібно зосередитись не лише на базових функціях картографічних систем, але й на унікальних властивостях та інноваціях, які відрізняють їх продукт від конкурентів
Відповідні тенденції ринку	На сучасному ринку існує зростаючий попит на інноваційні технологічні рішення для моніторингу та аналізу наслідків у різних територіях	Створення та розвиток систем, які дозволяють користувачам візуалізувати та аналізувати різноманітні дані про території, постраждалі від різних наслідків

В таблиці 5.8 наведено ступеневий аналіз конкуренції на ринку для визначення загальних рис конкуренції на ринку.

Важливим аспектом є розширення функціоналу таких систем. Це може включати інтеграцію з іншими системами і сервісами, розширення можливостей аналітики, створення інтерактивних інструментів для планування та координації дій на випадок надзвичайних ситуацій. Також важливим є забезпечення доступності інформації для різних цільових аудиторій - від урядових органів до звичайних громадян для подальшого ведення системи на споживчий ринок. Таким чином, розвиток та інтеграція нових технологій та підходів значно підвищить ефективність та функціональність систем, що в кінцевому результаті забезпечить кращу підготовку та реагування на надзвичайні події, а також розширить можливості їхнього використання різними цільовими аудиторіями.

Таблиця 5.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	У чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії)	Тип конкуренції
Тип конкуренції: Чиста	Наявність кількох платформ для моніторингу надзвичайних подій.	Потреба постійних інновацій.	За рівнем конкурентної боротьби
За рівнем конкурентної боротьби: Локальна	Конкуренція на вітчизняному ринку	У зв'язку з новизною продукту, локальна конкуренція поки що не є інтенсивною.	За галузевою ознакою
За галузевою ознакою: Міжгалузева	Конкуренція з компаніями з інших галузей, які пропонують альтернативні рішення для реагування на надзвичайні події	Можливе розширення функціоналу	Конкуренція за видами послуг
Конкуренція за видами товарів: Товарно-родова	На ринку присутні багато компаній, що пропонують різноманітні рішення для аналізу наслідків надзвичайних подій.	Стимуляція стартапу до створення унікальних продуктів, які задовольняють специфічні потреби клієнтів.	За характером конкурентних переваг
За характером конкурентних переваг: Нецінова	Компанії зосереджені на привабленні користувачів з метою максимізації прибутку через розробку нових додатків і систем за допомогою новітніх розробок	Коли один стартап набирає велику базу постійних клієнтів, перевага часто зміщується до якості та унікальності послуг.	За способами конкуренції

В таблиці 5.9 наведено Аналіз конкуренції в галузі за М. Портером.

Таблиця 5.9 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	Ринок є конкурентним, існує багато компаній, які пропонують подібні аналітичні інструменти	Нові стартапи можуть легко увійти в галузь, використовуючи новітні технології та інноваційні дані	Залежність від точних і актуальних даних є високою, постачальники даних мають значний вплив	Клієнти потребують надійних та оновлюваних в реальному часі даних для прийняття рішень	Існують альтернативні методи моніторингу, такі як особисті джерела інформації, соціальні мережі тощо
Висновки:	Інтенсивна конкуренція може вимагати постійних інновацій та унікальності послуг	Висока вірогідність залучення нових конкурентів вимагає стратегій для збереження частки ринку	Потрібно забезпечувати добрі відносини з постачальниками для стабільного доступу до даних	Важливість збереження довіри клієнтів та забезпечення високої якості послуг	Необхідність відрізняти продукт від товарів-замінників, підкреслюючи унікальні особливості і переваги

За визначеними факторами конкурентоспроможності проводиться аналіз сильних та слабких сторін стартап-проекту .

Результати аналізу сильних та слабких сторін проекту зображено в таблиці 5.10.

Таблиця 5.10 – Порівняльний аналіз сильних та слабких сторін проекту

Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів						
		–3	–2	–1	0	+1	+2	+3
Відсутність конкурентів	14		+					
Системні вимоги	18						+	
Простота використання	18				+			
Унікальний функціонал	20	+						

Фінальним етапом ринкового аналізу можливостей впровадження проєкту є складання SWOT-аналізу (матриці аналізу сильних (Strength) та слабких (Weak) сторін, загроз (Troubles) та можливостей (Opportunities) на основі виділених ринкових загроз та можливостей, та сильних і слабких сторін [34]. SWOT-аналіз проекту наведено в таблиці 5.11.

Таблиця 5.11 – SWOT-аналіз проекту

Сильні сторони:	Слабкі сторони:
Висока точність картографічних даних; Інтуїтивно зрозумілий інтерфейс користувача; Швидке оновлення інформації в режимі реального часу; Можливість інтеграції з різними джерелами даних; Метеодані отримані з офіційних джерел	Висока залежність від точності вхідних даних; Обмежені ресурси для підтримки широкомасштабної інфраструктури; Вразливість до кібератак та технічних збоїв;
Можливості: Співпраця з урядовими та некомерційними організаціями для покращення громадської безпеки; Використання передових технологій, таких як штучний інтелект для аналізу даних; Можливість отримання попереджень про надзвичайні події	Загрози: Конкуренція з великими технологічними компаніями, які мають більше ресурсів; Правові та регуляторні обмеження, пов'язані з збором та обробкою даних; Зміни клімату та непередбачуваність надзвичайних подій можуть збільшити вимоги до системи; Зміна законодавчих норм, що регулюють надання послуг з відображення новин та надання інформації про можливі загрози населенню

На основі SWOT-аналізу здійснюється розробка альтернативи ринкової поведінки для виведення стартап-проекту на ринок та орієнтовний оптимальний

час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок.

В таблиці 5.12 визначені альтернативи аналізуються з точки зору строків та ймовірності отримання ресурсів.

Таблиця 5.12 – Альтернативи ринкового впровадження стартап-проекту

Альтернатива ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
Вихід на нові ринки	Пошук інвесторів	Після виходу на ринок основного продукту
Розширення функціоналу	Пошук інвесторів	Після виходу на ринок основного продукту

Після проведеного аналізу зроблено висновок, що спочатку потрібно запустити програмний продукт, а вже потім розширювати функціонал і виходити на нові ринки.

5.4. Розроблення ринкової стратегії проекту

Розробка ринкової стратегії спочатку передбачає визначення стратегії охоплення ринку, а саме опис цільових груп потенційних споживачів . В таблиці 5.13 описано цільові групи потенційних споживачів.

Таблиця 5.13 – Вибір цільових груп потенційних споживачів

Опис цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в сегменті	Інтенсивність конкуренції в сегменті	Простота входу у сегмент

Кінець таблиці 5.13

Індивідуальні користувачі	Готові	Попит є	Середня	Просто
Місцеві та національні урядові органи	Готові	Невеликий попит	Висока	Складно
Засоби масової інформації та новинні агентства	Готові	Попит є	Мала	Просто
Академічні та дослідницькі установи	Готові	Попит є	Середня	Просто

За результатами аналізу потенційних груп споживачів (сегментів) автори ідеї обирають цільові групи, для яких вони пропонуватимуть свій товар, та визначають стратегію охоплення ринку.

В таблиці 5.14 визначається базова стратегія розвитку.

Таблиця 5.14 – Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Розробка нових продуктів	Створення інтерактивної карти з можливістю відображення даних в реальному часі. Розробка інтуїтивно зрозумілого і простого у використанні інтерфейсу. Постійне оновлення картографічної інформації та функціоналу.	Стратегія диференціації

Кінець таблиці 5.14

Концентрація на ринку	Визначення ключових сегментів ринку, які найбільш потребують такого продукту. Надання персоналізованих рішень для користувачів, таких як місцеві органи влади, рятувальні служби, та інші зацікавлені сторони.	Стратегія диференціації

Стратегія конкурентної поведінки визначається у таблиці 5.15

Таблиця 5.15 – Визначення базової конкурентної поведінки

Чи є проект «першопроходцем» на ринку	Ні
Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Основна стратегія залучення нових користувачів через інноваційні функції, такі як прогнозування ризиків та інтеграція з екстреними службами
Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Так. Адаптація найкращих функцій конкурентів з додаванням унікальних елементів, таких як підтримка спільноти та дані для покращення інформації на карті.
Стратегія конкурентної поведінки	Активна конкурентна стратегія

На основі вимог споживачів з сегментів до постачальника (стартап-компанії) та до програмного продукту, а також в залежності від обраної базової стратегії розвитку та стратегії конкурентної поведінки треба розробляється стратегія позиціонування, яка полягає у формуванні ринкової позиції, за яким споживачі мають ідентифікувати торгівельну марку чи проект. Стратегія позиціонування визначається в таблиці 5.16.

Таблиця 5.16 – Визначення стратегії позиціонування

Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту
Доступна ціна, простота і зручність використання	Стратегія диференціації	Легкість і простота у використанні. Доступність через ціну та технічні характеристики. Вирішення унікальних поставлених задач	стандарти якості; метрики програмного забезпечення.

Таблиця відображає стратегічне бачення розвитку продукту, націлене на створення конкурентних переваг, забезпечення високої доступності для цільової аудиторії, та встановлення ясних вимог до якості та характеристик продукту.Робота стартап-проекту повинна бути спланована певним чином: за стратегією диференціації виконуватиметься і поширюватиметься програмний продукт з унікальними якостями, дотримуючись у конкурентній поведінці стратегії виклику лідера, тобто буде випускатись один тип товару для всіх користувачів.

Висновки до розділу 5

Реалізація проекту, що має на меті створення інноваційних методів візуалізації для інтерактивних карт зон, що постраждали від різного роду подій, виправданий і має великий потенціал для підприємців у стартап-індустрії та інвесторів. Глибокий розбір існуючих систем візуалізації відкрив нові шляхи для розвитку унікальних функцій, які можна інтегрувати у новий продукт. Оцінюючи нинішню ринкову ситуацію, потреби користувачів та конкурентні переваги запропонованої системи, є всі підстави вважати, що програмне рішення матиме успіх на ринку.

ВИСНОВКИ

Було сформульовано задачу розробки інтерактивної карти для відстеження та відображення надзвичайних ситуацій. Відповідно до цієї задачі було проведено аналіз предметної області, вивчено основні методики та технології, які використовуються у сфері створення інтерактивних карт.

Створено базу даних з інформацією про надзвичайні події, розроблено інтерфейс для інтерактивного взаємодії користувача з картою та реалізовано систему сповіщень про нові надзвичайні ситуації в заданих регіонах. Користувачі можуть в режимі реального часу передивлятись повідомлення про надзвичайні ситуації, які вони спостерігають, що допомагає забезпечити актуальність інформації на карті.

Дані про надзвичайні ситуації зберігаються в базі даних, що дозволяє аналізувати та прогнозувати потенційні загрози в певних регіонах. Ці дані можуть використовуватися органами управління надзвичайними ситуаціями для оптимізації роботи рятувальних служб.

Для розробки інтерактивного інструменту використано мову програмування Python, фреймворк Flask для серверної частини, та JavaScript з бібліотекою Leaflet для реалізації інтерактивної карти.

Інтерактивна карта надзвичайних ситуацій дозволяє користувачам оперативно отримувати інформацію про поточні події та реагувати на них відповідним чином, що забезпечує безпеку громадян та ефективність реагування служб надзвичайних ситуацій.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Python. [Електроний ресурс]. - 2020 - : https://www.python.org/about/
2. JavaScript Notes for Professionals [Електронний ресурс] – Режим доступу: https://goalkicker.com/JavaScriptBook/JavaScriptNotesForProfessionals.pdf (дата звернення 09.09.2023)
3. Bootstrap [Електронний ресурс] –Режим доступу: https://getbootstrap.com/docs/5.0/customize/overview/ (дата звернення 18.09.2023)
4. Flask. [Електроний ресурс]. 2020 - : https://flask.palletsprojects.com/en/1.1.x/
5. Grinberg M. Flask Web Development: Developing Web Applications with Python 2nd Edition / Miguel Grinberg., 2018. – 310 с.
6. Sweigart A. Automate the Boring Stuff with Python: Practical Programming for Total Beginners. 2-ге вид. No Starch Press, 2019. 592 с.
7. Welcome to flask – flask documentation (2.3.x). Welcome to Flask – Flask Documentation (2.3.x). URL: https://flask.palletsprojects.com/en/2.3.x/ (дата звернення: 04.06.2023).
8. Tragura S. J. Building python microservices with fastapi: build secure, scalable, and structured python microservices from design concepts to infrastructure. Packt Publishing, Limited, 2022.
9. Summerfield M. Programming in python 3 A complete introduction to the python language. New York : Addison-Wesley, 2010. 630 с.
10. Stouffer J. Mastering flask. Birmingham : Packt Publishing, 2015. 288 с.
11. Martin R. C., Feathers M. C., Ottinger T. R. Clean code. New York: Pearson 2008. 464 с.
12. Mammen E. Introduction. When Does Bootstrap Work?. New York: 3Island Press, 1992. 212 с.
13. Lutz M. Learning python powerful object-oriented programming. Sebastopol : O'Reilly Media, 2013. 1600 с.

14. Customize. Bootstrap · The most popular HTML, CSS, and JS library in the world. URL: https://getbootstrap.com/docs/5.0/customize/overview/ (дата звернення: 04.06.2023).
15. Bibeault B. B., Katz Y. Y., Aurelio De Aurelio De Rosa. JQuery in action. Manning Publications Company, 2015. 504 с.
16. Chopra V. Websocket Essentials: Building Apps with Html5 Websockets. Packt Publishing, Limited, 2015.
17. Paul DuBois. MySQL Cookbook: Solutions for Database Developers and Administrators 3rd Edition / Paul DuBois., 2014. 866 с.
18. Jake VanderPlas. Python Data Science Handbook: Essential Tools for Working with Data 1st Edition / Jake VanderPlas., 2016. 548 с.
19. Building web cloud and mobile solutions with F. O'Reilly Media, 2013.
20. Chris Albon. Machine Learning with Python Cookbook: Practical Solutions from Preprocessing to Deep Learning 1st Edition / Chris Albon., 2018. 358 с.
21. John C. Shovic. Luciano Ramalho. Fluent Python: Clear, Concise, and Effective Programming 1st Edition / Luciano Ramalho., 2015. 792 с.
22. SWOT-аналіз із прикладами: веб-сайт. URL: https://esputnik.com/uk/blog/swot-analiz-iz-prikladami (дата звернення 10.11.2023).
23. Йохна М., Стадник В. В. Економіка і організація інноваційної діяльності: навчальний посібник. К.: Видавничий центр „Академія”, 2005. 400 с.
24. MySQL Workbench. URL: https://www.mysql.com/products/workbench/ (дата звернення: 03.10.2023).