КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп'ютерних наук та кібернетики
Кафедра теорії та технології програмування

Кваліфікаційна робота
на здобуття ступеня бакалавра
за спеціальністю 122 Комп'ютерні науки
на тему:

РОЗРОБКА ІНТЕРАКТИВНОЇ БАЗИ ТОЧОК ГЕОКООРДИНАТ. КОРИСТУВАЦЬКИЙ ІНТЕРФЕЙС

Виконав студент 4 курсу
Едуард АНДРАЩУК _____

Науковий керівник:
доцент, к.п.н.
Наталія РУСІНА _____

Засвідчую, що в цій роботі немає запозичень з праць інших авторів без відповідних посилань
Студент _____

Роботу розглянуто й допущено до захисту на засіданні кафедри теорії та технології програмування
«____» ____ 2023р., протокол № _____

Завідувач кафедри
Микола НІКІТЧЕНКО _____

Київ – 2023

РЕФЕРАТ

Обсяг роботи: загальний обсяг 45 сторінок, з них основний текст – 38 сторінко, 10 ілюстрацій, 22 використаних джерела та 1 додаток.

ВЕБЗАСТОСУНОК, АВТОМАТИЗАЦІЯ ОБЛІКУ БАЗИ ГЕОКООРДИНАТ, ВЕБ-ІНТЕРФЕЙС, ДИЗАЙН, КОРИСТУВАЦЬКИЙ ІНТЕРФЕЙС.

Кваліфікаційна робота складається із вступу, трьох розділів, висновків, списку використаних джерел

Об’єктом дослідження є процес відображення бази точок геокординат.

Предмет роботи – користувацький інтерфейс для керування точками інтерактивної бази точок геокоординат.

Метою кваліфікаційної роботи є розробка користувацького інтерфейсу для керування точками інтерактивної бази точок геокоординат.

Методи розроблення: аналіз наявних аналогічних застосунків, проєктування та розробка дизайну та користувацького інтерфейсу.

Інструменти розроблення: Для дизайну була використана програма Figma. Для програмної реалізації було обрано інтегроване середовище WebStorm, мову програмування TypeScript. Система керування базами даних – MySQL.

Результат роботи: Користувацький веб-інтерфейс, що дозволяє керувати точками інтерактивної бази точок геокоординат.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

CSS – Cascading style sheets, каскадні таблиці стилів

DOM – Document Object Model, об'єктна модель документа

ERW – Explosive remnants of war, вибухонебезпечні залишки війни

HTML - Hypertext markup language, мова розмітки гіпертексту

IDE – Integrated development environment, інтегроване середовище розробки

JS – JavaScript, Джава Скрипт

NPM – Node Package Manager, менеджер пакетів Node

OSM – OpenStreetsMap, Опен Стріт Мап

SQL – Structured query language, мова структурованих запитів

UI – User interface, інтерфейс користувача

URL – Uniform Resource Locator, уніфікований локатор ресурсів

ГІС – Геоінформаційна система

ВСТУП

Оцінка сучасного стану об’єкта розробки. Доступ до картографічних систем без доступу до Інтернету на сьогоднішній день є актуальним питанням. Зокрема, в місцях бойових дій на території України, в яких була зруйнована енергетична та комунікаційна інфраструктура; а також в країнах з низьким рівнем Інтернет-покриття. І це є серйозним недоліком подібних систем, які не можуть працювати в автономному режимі, що є серйозним обмеженням в умовах відсутності стабільного інтернет-з'єднання. На сьогоднішній день значна частина картографічних систем немає достатньої гнучкості та адаптивності. Це обмежує їх використання в різних сценаріях. Ці проблеми актуальні для таких систем як Google Maps [1], Mapbox [2], MapQuest [3], Combat Vision [4], GIS Arta [5], Bing Maps [6] та інших.

Актуальність роботи та підстави для її виконання. Наріжною перевагою даної системи є її висока придатність для використання в умовах обмеженого доступу до Інтернету, особливо в тих частинах світу, де мережеве покриття є недостатнім або практично відсутнім. В таких маловивчених, віддалених або прифронтових регіонах, де зв'язок зі світовою мережею є обмеженим, дана система може виявитися незамінною для різноманітних цілей.

Станом на 2023 рік оцінюється, що територія України, яка серйозно постраждала від вибухонебезпечних залишків війни (ERW), становить принаймні 160 000 квадратних кілометрів, що відповідає площі Великої Британії. Це число зростає щодня, оскільки війна триває. До проведення детальних обстежень ці оцінки залишаються просто оцінками, хоча попередній досвід показує, що початкові оцінки можуть переоцінювати рівень забруднення [7].

Загроза мін та забруднення на території, що піддана вторгненню, створює серйозні ризики для життя та безпеки людей, а також ускладнює

ведення військових операцій. Ефективна система позначення на карті таких територій дозволяє забезпечити точну та оперативну інформацію про мінні поля, хімічні забруднення та інші небезпечні об'єкти.

Також одним із основних сценаріїв використання цієї системи є наукові дослідження. Вчені та дослідники, які працюють у віддалених локаціях, таких як гірські райони, пустелі або джунглі, можуть скористатися цією системою для збору, обробки та аналізу географічних даних. Навіть без доступу до Інтернету, вони можуть продовжувати свої дослідження, маркуючи та відстежуючи точки геокоординат, створюючи картографічні показники та збираючи цінну інформацію для своїх наукових робіт.

Туризм є ще однією сферою, де дана система може мати велике значення. У туристичних подорожах, особливо в екстремальних або далеких місцях, може бути обмежений або навіть відсутній доступ до Інтернету. Завдяки можливостям автономної роботи цієї системи, туристи можуть використовувати її для навігації, позначення та відстеження своєї маршруту, а також відкривати нові місця та досліджувати незаймані туристами території. Це робить її цінним інструментом для пригодницьких туристів, екологів та дослідників, які шукають унікальні природні та культурні об'єкти.

Таким чином, необхідно враховувати, що малодоступність Інтернету в деяких частинах світу, особливо маловивчених, а також потреба в автономному функціонуванні робить дану систему вкрай актуальною та корисною для наукових досліджень, туризму та багатьох інших сфер діяльності. Вона забезпечує надійну та універсальну базу даних геокоординат, яка може бути використана навіть у віддалених районах, де інші системи можуть бути непридатними.

Мета й завдання роботи. Мета даної роботи полягає в розробці користувацького інтерфейсу для керування точками інтерактивної бази

точок геокоординат. Розробка зручного та ефективного інтерфейсу є важливим кроком для покращення користувацького досвіду та забезпечення зручного доступу до географічних даних.

Відповідно до поставленої мети, визначені такі завдання:

- провести огляд існуючих застосунків на сучасному ринку;
- спроєктувати інтерфейс користувача;
- розробити візуальний дизайн інтерфейсу;
- провести тестування.

Об'єкт, методи й засоби розроблення. Об'єктом розробки є процес відображення бази точок геокординат. Дизайн інтерфейсу був розроблений у програмі Figma [7]. Програмна розробка проводилась в інтегрованому середовищі WebStorm [8], за допомогою мови програмування TypeScript [10].

Можливі сфери застосування. Інтерактивна база даних геокоординатних точок з можливістю використання в автономному режимі може бути надзвичайно корисною для широкого спектру застосувань. Зокрема, одним з можливих варіантів використання може бути відображення актуальних замінованих чи забруднених територій, а також дослідження та туризм у малодосліджених та віддалених регіонах.

Зв’язок з іншими роботами. У кваліфікаційній роботі використано як основу проєкт, який був розроблений у ході виконання групового проєкту студентів 2 та 4 курсів, поєднаного з практичними частинами дисциплін «Коректність програм та логіки програмування» (4 курс) та «Методи специфікації програм» (4 курс), «Інструментальні середовища та технології програмування» (2 курс).

РОЗДІЛ 1 ОГЛЯД ІСНУЮЧИХ СИСТЕМ

1.1 Веб-сервіс Google Maps

Google Maps – це картографічний веб-сервіс, який надає детальну географічну інформацію, супутникові знімки, карти вулиць і панорамні види вулиць на 360° (Перегляд вулиць). Він також пропонує інформацію про стан дорожнього руху в реальному часі (Google Traffic) і планування маршрутів для подорожей пішки, автомобілем, велосипедом, повітрям (у бета-версії) і громадським транспортом. Однією з ключових особливостей Карт Google є функція прокладання маршрутів (рисунок 1.1).

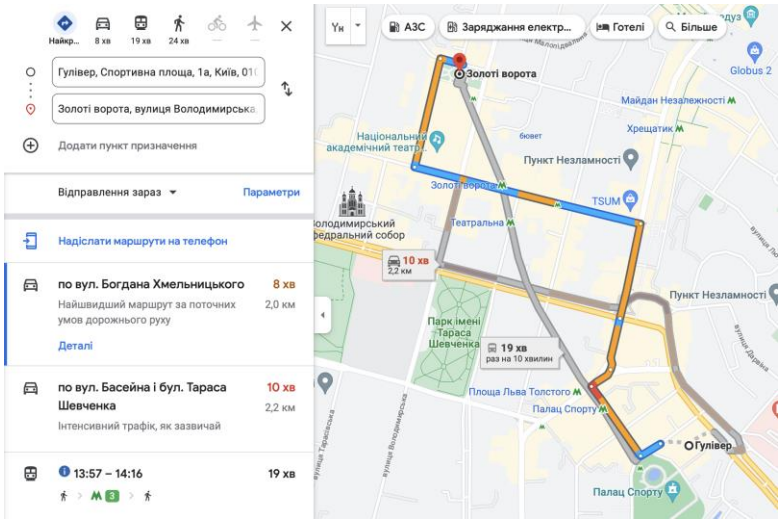

Рисунок 1.1 – Побудова маршруту за допомогою Google Maps

Він надає користувачам детальні покрокові інструкції, приблизний час у дорозі та альтернативні маршрути на основі поточних дорожніх умов. Ця функція особливо корисна для планування подорожей, поїздок на роботу або логістичних операцій.

Карти Google також пропонують комплексну функцію пошуку, яка дозволяє користувачам знаходити певні місця, підприємства або об'єкти. Ця функція доповнюється великою базою даних сервісу, яка включає ресторани, готелі, магазини тощо.

З точки зору кастомізації, Карти Google дозволяють користувачам створювати персоналізовані карти з власними об'єктами та маршрутами. Ці кастомні карти можна зберігати, ділитися ними і навіть вбудовувати на веб-сайти.

Однак, хоча Карти Google пропонують певну офлайн-функціональність, вона обмежена. Користувачі можуть завантажити карти певних територій для використання в автономному режимі, але ці карти не включають всіх можливостей онлайн-сервісу. Наприклад, на офлайн-картах не відображаються актуальні дані про дорожній рух, маршрути пішохідних переходів чи інформацію про транзит. Крім того, за один раз можна завантажити лише обмежену територію, а термін дії цих офлайн-карт закінчується через 30 днів, якщо їх не оновити.

Таким чином, Карти Google – це комплексний і універсальний картографічний сервіс з широким спектром можливостей, але його офлайн-функціональність обмежена і може не відповідати потребам користувачів, які потребують повного доступу до картографічних функцій без підключення до Інтернету.

1.2 Платформа Mapbox

MapBox – це картографічна платформа, яка пропонує набір сервісів, включаючи карти, пошук, навігацію тощо. Вона широко використовується завдяки картам, що налаштовуються, та широкому набору функцій, які дозволяють розробникам створювати інтерактивні та візуально привабливі додатки.

Однією з ключових особливостей MapBox є карти, що налаштовуються. Користувачі можуть створювати власні карти за

допомогою MapBox Studio (рисунок 1.2), обираючи з безлічі стилів, шарів та функцій, щоб створити унікальну та персоналізовану карту.

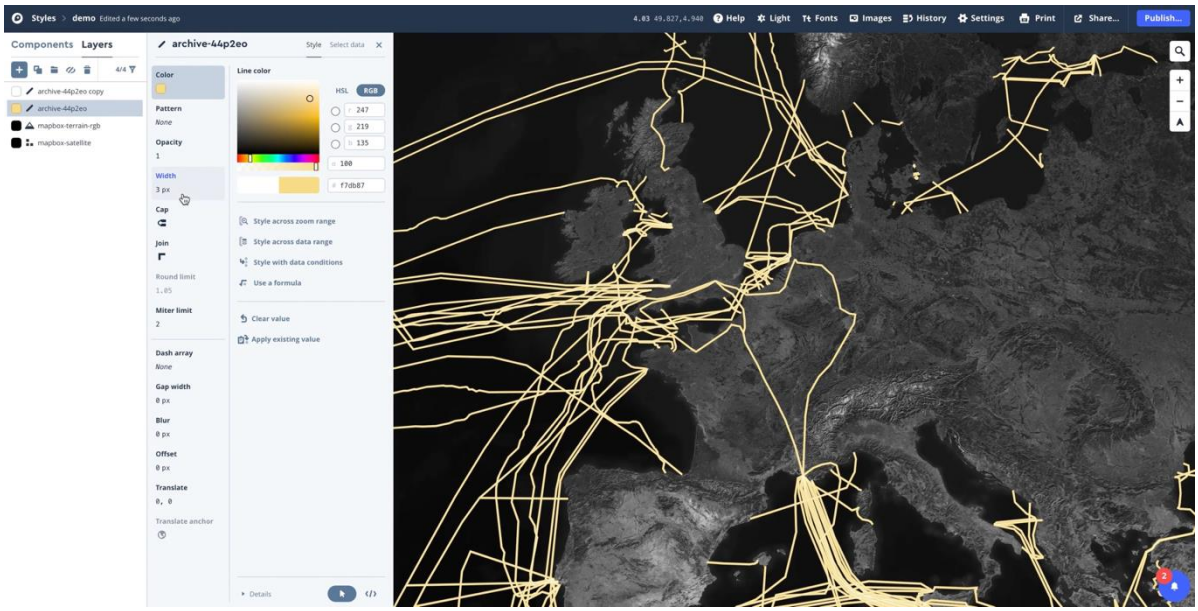

Рисунок 1.2 – Інтерфейс MapBox Studio

Ця функція особливо корисна для розробників, які хочуть, щоб стиль карти відповідав естетиці їхнього додатку або веб-сайту.

MapBox також пропонує надійну функцію пошуку, відому як MapBox Geocoding, яка дозволяє користувачам шукати певні місця або об'єкти. Ця функція доповнюється великою базою даних платформи, яка включає мільйони місць по всьому світу.

З точки зору навігації, MapBox надає покрокові інструкції, умови дорожнього руху та оптимізацію маршруту. Ця функція особливо корисна для додатків, які потребують можливостей маршрутизації та навігації, таких як спільні поїздки або служби доставки.

Однак важливо зазначити, що хоча MapBox і надає певні офлайн-функції, вони обмежені. Користувачі можуть завантажувати карти для використання в автономному режимі, але ці карти є статичними і не містять таких функцій, як пошук або навігація. Це означає, що хоча MapBox можна певною мірою використовувати автономно, повний набір його функцій вимагає підключення до Інтернету.

Таким чином, MapBox є універсальною картографічною платформою з широким спектром функцій і можливостей налаштування, але її офлайн-функціональність обмежена.

1.3 Сервіс MapQuest

MapQuest – це універсальний картографічний сервіс, який надає різноманітні функції для прокладання маршрутів, пошуку місцезнаходження та налаштування мап. Він пропонує зручний інтерфейс (рисунок 1.2) і детальні карти, які включають в себе точки, умови дорожнього руху і супутникові знімки.

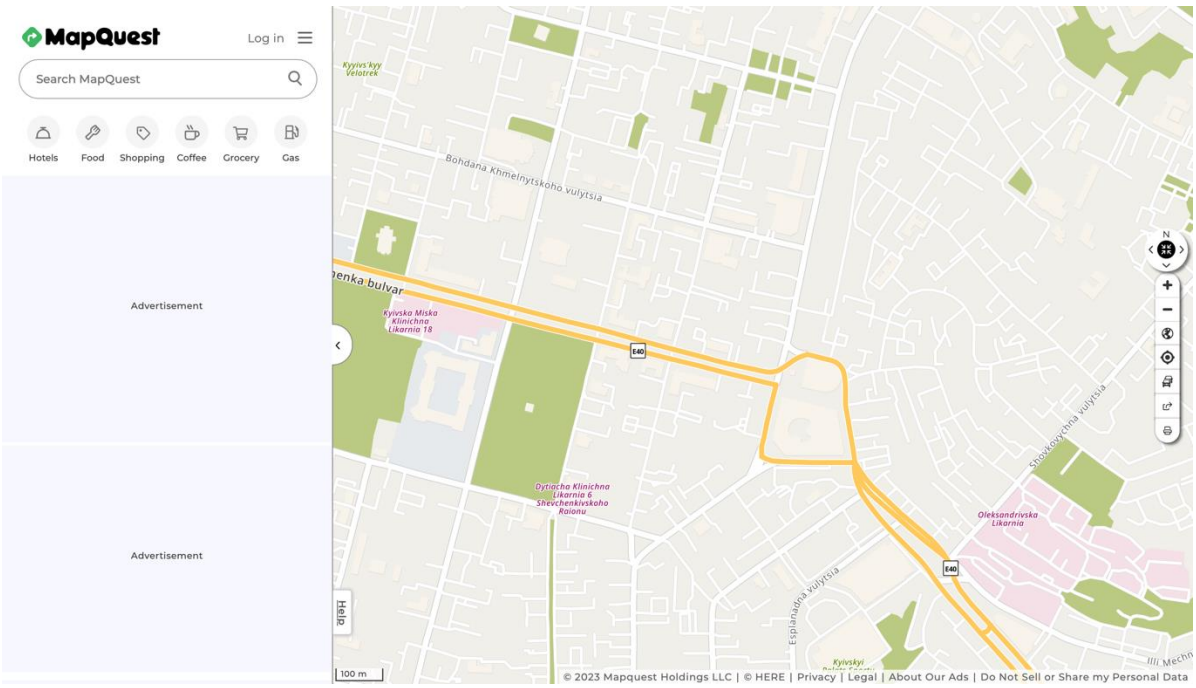

Рисунок 1.3 – Інтерфейс MapQuest

Однією з ключових особливостей MapQuest є функція маршрутизації. Він дозволяє користувачам знаходити найкращі маршрути між кількома місцями, враховуючи поточну ситуацію на дорогах і надаючи покрокові вказівки.

MapQuest також надає комплексну функцію пошуку, яка дозволяє користувачам знаходити конкретні місця, підприємства або визначні

пам'ятки. Ця функція доповнюється великою базою даних сервісу, яка включає ресторани, готелі, магазини тощо.

З точки зору кастомізації, MapQuest дозволяє користувачам персоналізувати свої карти, додаючи маркери, малюючи фігури або накладаючи зображення. Ця функція може бути корисною для різних цілей, від планування туристичної подорожі до візуалізації бізнес-даних.

Однак важливо зазначити, що для роботи MapQuest потрібне підключення до Інтернету. Він не пропонує офлайн-режиму, а це означає, що його не можна використовувати автономно без доступу до інтернету. Це обмеження може бути суттєвим недоліком у районах з поганим або відсутнім інтернет-зв'язком.

Підсумовуючи, MapQuest – це комплексний картографічний сервіс з можливостями маршрутизації та пошуку, але відсутність офлайн-функцій може обмежити його корисність у певних ситуаціях.

1.4 Веб-сервіс Bing Maps

Bing Maps – це картографічний веб-сервіс від Microsoft, який пропонує різноманітні функції, включаючи карти вулиць, 3D-моделі міст, супутникові знімки, аерофотозйомку тощо. Він також надає додаткові послуги, такі як інформація про дорожній рух, планування маршрутів для різних видів транспорту та визначні пам'ятки.

Однією з ключових особливостей Bing Maps є його детальні та точні картографічні можливості (рисунок 1.4).

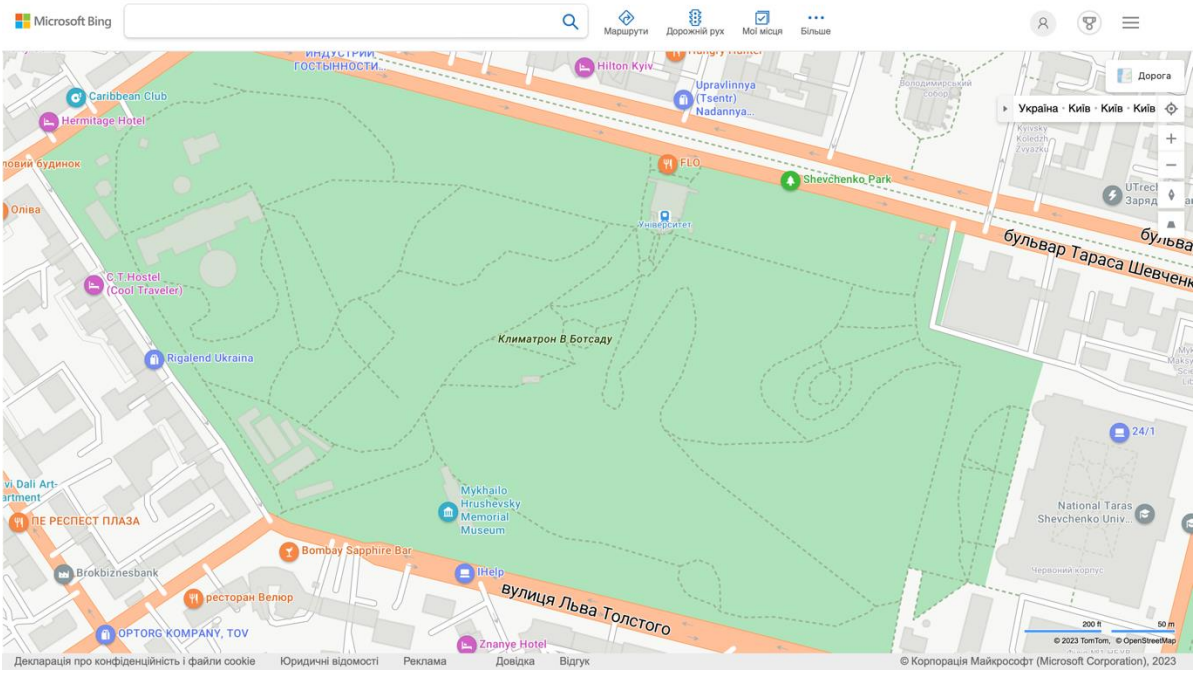

Рисунок 1.4 – Детальність карт Bing Maps

Він надає користувачам вичерпну географічну інформацію, включно з деталями вулиць, пам'ятками та списками підприємств. Ця функція особливо корисна для навігації та пошуку на основі місцезнаходження.

Bing Maps також пропонує надійні можливості прокладання маршрутів і визначення напрямків. Користувачі можуть отримати покрокові вказівки для проїзду на автомобілі, пішки або громадським транспортом. Сервіс враховує поточні умови дорожнього руху, щоб запропонувати найефективніший маршрут і приблизний час у дорозі.

З точки зору пошукової функціональності, Bing Maps дозволяє користувачам шукати певні місця, підприємства або об'єкти. Ця функція доповнюється великою базою даних сервісу, яка включає ресторани, готелі, магазини тощо.

Однак важливо зазначити, що Bing Maps, як і багато інших картографічних онлайн-сервісів, вимагає підключення до Інтернету для доступу до більшості своїх функцій. Хоча Bing Maps дозволяє

користувачам завантажувати карти для перегляду в автономному режимі, ці офлайн-карти не включають всі можливості онлайн-сервісу. Наприклад, офлайн-карти не надають оновлень про дорожній рух в режимі реального часу або покрокової навігації. Це означає, що хоча Bing Maps можна певною мірою використовувати без підключення до Інтернету, його повна функціональність вимагає доступу до мережі.

Таким чином, Bing Maps – це комплексний картографічний сервіс з широким спектром можливостей, але його офлайн-функціональність обмежена, що може обмежити його використання в ситуаціях, коли підключення до Інтернету недоступне.

Було розглянуто різні існуючі картографічні системи, включаючи Google Maps, Bing Maps, MapQuest та MapBox. Кожна з цих систем пропонує унікальний набір функцій і можливостей, таких як детальне картографування, надійна маршрутизація та комплексні функції пошуку. Однак загальним обмеженням цих систем є їхня залежність від підключення до Інтернету для повноцінної роботи, що може бути суттєвим недоліком у районах з поганим або відсутнім інтернет-зв'язком.

Тому потреба в картографічній системі з автономною функціональністю та надійною, універсальною базою геокоординат є дуже актуальною. Це забезпечить безперервну роботу з геоінформаційними системами незалежно від наявності стабільного інтернет-з'єднання та географічного розташування користувачів.

РОЗДІЛ 2 ВИКОРИСТАНІ ТЕХНОЛОГІЇ

2.1. Графічний редактор Figma

Для створення дизайну користувацького інтерфейсу було використано програму Figma [7], Figma – це універсальний інструмент дизайну, який широко відомий завдяки своїм можливостям для розробки користувацького інтерфейсу та дизайну користувацького досвіду, а також для створення інтерактивних прототипів. Його хмарна природа дозволяє співпрацювати в режимі реального часу, що робить його популярним вибором для командних проектів.

Однією з ключових особливостей Figma є можливості векторного дизайну та створення прототипів. Це дозволяє користувачам створювати, налаштовувати та анімувати векторну графіку, яку можна масштабувати без втрати якості. Ця функція особливо корисна для проектування інтерфейсів для різних розмірів екранів і роздільної здатності.

Figma також пропонує широкий набір інструментів для створення та керування компонентами дизайну. Користувачі можуть створювати багаторазові компоненти, організовувати їх у бібліотеки та ділитися ними між проектами. Це може значно пришвидшити процес проектування та забезпечити узгодженість між проектами.

З точки зору співпраці, Figma виділяється своїм середовищем для спільного проектування в режимі реального часу. Кілька користувачів можуть одночасно працювати над одним дизайном, бачити зміни один одного в режимі реального часу і навіть співпрацювати віддалено. Ця функція, у поєднанні з можливістю коментувати та надавати відгуки безпосередньо на дизайн, робить Figma корисним інструментом для командної роботи над проєктами.

Figma також підтримує прототипування. Користувачі можуть пов'язувати елементи дизайну, створювати інтерактивні компоненти та

імітувати взаємодію з користувачем. Це дає можливість створювати і тестувати функціональні прототипи, не виходячи з середовища проектування.

2.2. Інтегроване середовище WebStorm

Розробка велася в інтегрованому середовищі, WebStorm. WebStorm – це інтегроване середовище розробки (IDE), розроблене компанією JetBrains спеціально для розробки на JavaScript. Воно підтримує широкий спектр сучасних технологій, включаючи Node.js, HTML, CSS та сучасні фреймворки JavaScript, такі як React, Angular та Vue.js.

Однією з ключових особливостей WebStorm є інтелектуальне завершення коду. Ця функція надає підказки та автоматичне завершення для змінних, функцій та ключових слів під час введення, що робить кодування швидшим та менш схильним до помилок.

WebStorm також пропонує надійні інструменти для налагодження та тестування. Вбудований відладчик дозволяє переглядати код і перевіряти змінні та вирази, що полегшує виявлення та виправлення проблем. Інструменти тестування підтримують різні фреймворки тестування і надають візуальний інтерфейс для запуску тестів і перегляду результатів.

З точки зору контролю версій, WebStorm має вбудовану підтримку Git та інших систем контролю версій. Це дозволяє легко керувати змінами, переглядати відмінності та вирішувати конфлікти безпосередньо в IDE.

WebStorm також пропонує ряд інструментів для роботи з базами даних і розробки на стороні сервера. Це вбудований термінал, підтримка Docker, інструменти для роботи з SQL і базами даних.

З точки зору кастомізації, WebStorm дуже добре налаштовується. Ви можете налаштувати макет, ярлики та стиль кодування відповідно до ваших уподобань. Він також підтримує широкий спектр плагінів для розширення його функціональності.

Таким чином, WebStorm – це комплексне середовище розробки JavaScript, що пропонує широкий спектр функцій та інструментів для підтримки сучасної веб-розробки.

2.3 Система Git

Для контролю версій було використано Git – це розподілена система контролю версій, яка широко використовується у розробці програмного забезпечення. Вона була розроблена для швидкої та ефективної роботи з будь-якими проєктами – від невеликих до дуже великих. Git простий у вивченні, займає мініатюрний розмір і має блискавичну продуктивність.

Однією з ключових особливостей Git'у є його модель розгалуження. Це дозволяє розробникам створювати гілки для різних функцій або версій, працювати над ними незалежно, а потім об'єднувати їх назад в основну кодову базу, коли вони будуть готові. Це полегшує управління складними проектами і дозволяє декільком розробникам працювати над одним проектом одночасно, не заважаючи один одному.

Git також пропонує надійні інструменти для відстеження змін та управління версіями. Щоразу, коли ви фіксуєте або зберігаєте стан вашого проекту в Git'i, він фактично робить знімок того, як виглядають всі ваші файли в цей момент, і зберігає посилання на цей знімок. Це дозволяє легко відстежувати зміни, повертатися до попередніх версій і вирішувати конфлікти.

З точки зору співпраці, Git призначений для розподіленої розробки. Кожен розробник отримує власний локальний репозиторій з повною історією комітів. Зміни переносяться між репозиторіями у вигляді набору патчів, що дозволяє створювати гнучкі робочі процеси та моделі співпраці.

Git також надає інструменти для забезпечення цілісності вашого коду. Кожен файл і кожна фіксація в Git'i мають контрольну суму, і при зворотному вилученні відновлюються за контрольною сумою. Це означає,

що неможливо змінити вміст будь-якого файлу або каталогу без відома Git'a, що забезпечує високий рівень безпеки для ваших проектів.

Отже, Git – це інструмент для контролю версій та спільної розробки програмного забезпечення, що пропонує широкий спектр можливостей для керування кодом та роботи з іншими користувачами.

2.4. Мова програмування TypeScript

Для розробки було використано популярну мову програмування TypeScript [10], яка є статично типізованою надмножиною JavaScript [11], яка додає до мови додаткові типи. Розроблений і підтримуваний корпорацією Майкрософт, TypeScript покликаний зробити розробку великомасштабних JavaScript-додатків простішою та ефективнішою.

Однією з ключових особливостей TypeScript є статична типізація. Це дозволяє розробникам визначати типи змінних, параметрів функцій та властивостей об'єктів. Компілятор TypeScript перевіряє ці типи під час компіляції, допомагаючи виявити помилки до того, як код буде запущено. Це може призвести до створення більш надійного коду, який легше підтримувати.

TypeScript також підтримує сучасні можливості JavaScript, включаючи класи, модулі та функції стрілок, і може компілювати цей код до старих версій JavaScript для сумісності зі старими браузерами. Це робить TypeScript гарним вибором для розробки великомасштабних додатків, які мають працювати на різних платформах.

На додаток до своїх основних можливостей, TypeScript також пропонує ряд інструментів і функцій, які покращують процес розробки. Сюди входять такі функції, як автозавершення, виведення типів та інтерфейси, які можуть зробити код легшим для читання та розуміння.

TypeScript також добре інтегрується з багатьма популярними JavaScript-фреймворками та бібліотеками, такими як React, Angular та

Vue.js, і підтримується з коробки у багатьох популярних середовищах розробки, включаючи Visual Studio Code, WebStorm та Atom.

Таким чином, TypeScript – це корисний інструмент для розробки JavaScript, що пропонує статичну типізацію та сучасні функції JavaScript, а також інструменти та інтеграції, які можуть покращити досвід розробки.

2.5. Середовище виконання Node.js

Як середовище виконання було обрано Node.js – середовище виконання JavaScript, яке дозволяє розробникам створювати масштабовані серверні додатки. Воно побудоване на JavaScript-двигуні Google Chrome V8 і забезпечує ефективний спосіб запуску JavaScript на стороні сервера. Node.js був вперше випущений у 2009 році, і з того часу він став однією з найпопулярніших технологій у світі веб-розробки [12].

Однією з головних переваг Node.js є його легкість та швидкість. Це дозволяє розробникам створювати високопродуктивні веб-додатки, які можуть обробляти велику кількість одночасних з'єднань без шкоди для швидкості. Крім того, Node.js має широку екосистему модулів і пакетів, які полегшують створення складних вебзастосунків [13]. Ця екосистема включає, зокрема, інструменти для веб-розробки, тестування та розгортання.

Node.js також є кросплатформним, що означає, що він може працювати на різних операційних системах, включаючи Windows, macOS та Linux. Це дозволяє розробникам легко створювати додатки, які можуть працювати на різних платформах без значних змін у кодовій базі.

Ще однією значною перевагою Node.js є його неблокуюча модель вводу/виводу, яка дозволяє йому обробляти декілька запитів одночасно. Це робить його ідеальним для створення веб-додатків у реальному часі, таких як чат-додатки, онлайн-ігри та інструменти для спільної роботи. Node.js також може одночасно обробляти декілька підключень до баз даних, що

робить його придатним для створення додатків, що працюють з великими обсягами даних.

Node.js широко використовується для створення веб-серверів, інструментів командного рядка та веб-додатків у режимі реального часу. Це важливий інструмент для будь-якого розробника, який прагне створювати масштабовані та ефективні веб-додатки. Його універсальність, швидкість і простота використання роблять його популярним вибором серед розробників по всьому світу. Спільнота Node.js також активна і постійно розробляє нові функції та вдосконалення.

2.6. Бібліотека React

Також для побудови інтерфейсу була використана популярна JavaScript-бібліотека для створення користувацьких інтерфейсів React [14], яка здобула величезну популярність серед розробників по всьому світу. Розроблена компанією Meta [15], React стала основним вибором для розробників, які хочуть створювати високопродуктивні та масштабовані користувацькі інтерфейси.

Однією з основних переваг React є те, що він дозволяє розробникам розбивати інтерфейс на багаторазові компоненти, що полегшує управління та підтримку коду [16]. Це означає, що розробники можуть створювати модульний код, який можна використовувати в різних частинах програми, що робить розробку більш ефективною.

Ще однією чудовою особливістю React є його віртуальна DOM, яка робить оновлення інтерфейсу користувача більш ефективними. Віртуальний DOM дозволяє React оновлювати лише ті частини інтерфейсу користувача, які змінилися, замість того, щоб перемальовувати весь інтерфейс, що призводить до більш швидкого та більш ефективного оновлення.

React також пропонує широку екосистему інструментів та бібліотек,

що полегшує розробникам створення складних користувацьких інтерфейсів. Екосистема React включає в себе інструменти для управління станами, маршрутизації, тестування та багато іншого, що допомагає розробникам оптимізувати робочий процес і підвищити продуктивність.

React - це перевірена часом бібліотека для створення динамічних та інтерактивних користувацьких інтерфейсів. Її ключові особливості, такі як віртуальний DOM, багаторазові компоненти та потужна екосистема, роблять її гарним вибором для багатьох типів проєктів.

2.7 Бібліотека Material-UI

Також було використано бібліотеку Material-UI для React. Це популярна бібліотека з відкритим вихідним кодом, яка надає розробникам готові React-компоненти, що відповідають принципам Material Design від Google. Material-UI надає ряд компонентів, які можна використовувати для створення сучасних, адаптивних та зручних для користувача веб-додатків. Ці компоненти включають в себе кнопки, форми, введення даних, навігацію та багато іншого.

Однією з головних переваг Material-UI є те, що він відповідає керівним принципам Material Design, які надають розробникам перевірений набір принципів дизайну [17]. Це полегшує розробникам створення веб-додатків, які є не лише візуально привабливими, але й зручними та доступними для користувача.

Material-UI також надає широкий спектр можливостей налаштування, що дозволяє розробникам пристосовувати компоненти до своїх конкретних потреб. Це включає можливість змінювати кольорову палітру, типографіку та інтервали між компонентами.

Ще однією перевагою Material-UI є те, що вона надає чудову документацію та підтримку. Бібліотека має активну спільноту розробників, які роблять свій внесок у її розвиток і надають підтримку

іншим користувачам. Це полегшує розробникам вивчення та використання бібліотеки.

Для реалізації таблиці було використано бібліотеку @mui/x-data-grid яка є дуже гнучким і настроюваним компонентом сітки даних, спеціально створений для React-додатків. Він розроблений для безперебійної роботи з іншими компонентами Material-UI, що полегшує розробникам створення складних багатоколонкових сіток, які відображають великі обсяги даних в організованому та зручному для користувача вигляді.

Однією з ключових особливостей @mui/x-data-grid є його здатність обробляти великі набори даних без шкоди для продуктивності, що дуже важливо для геопросторових даних. Компонент створено для швидкої та ефективної роботи навіть з тисячами рядків і стовпців. Він також надає розширені можливості фільтрації та сортування, що полегшує користувачам пошук потрібних даних.

Достатньо гнучким є @mui/x-data-grid з широким спектром можливостей для стилізації, компонування та поведінки. Розробники можуть легко налаштувати зовнішній вигляд сітки відповідно до дизайну свого додатку, а також додати власну функціональність, використовуючи багатий API компонента. Це дозволяє легко створювати персоналізований та цікавий користувацький досвід.

Компонент @mui/x-data-grid особливо корисний для розробників, яким потрібно керувати та відображати великі обсяги даних у своїх React-додатках. Він надає розширені можливості, такі як пагінація, виділення рядків та зміна розміру стовпців, що дозволяє легко створювати ефективну сітку даних.

Загалом, @mui/x-data-grid - це гнучкий та простий у використанні інструмент для управління та відображення великих обсягів даних у React-додатках. Він надає функції та можливості, необхідні розробникам для створення адаптивних, зручних для користувача сіток даних, які можуть

обробляти навіть найскладніші набори даних. Незалежно від того, чи створюєте ви просту таблицю, чи складну візуалізацію даних, @mui/x-data-grid є важливим інструментом у вашому наборі інструментів для розробки React.

2.8 Інструменти роботи з картами

Для відмальовування карта були використані OpenStreetMap та бібліотека Leaflet. OpenStreetMap – це глобальна ініціатива, метою якої є створення безкоштовної карти світу з відкритим вихідним кодом. Вона була створена у 2004 році Стівом Костом і підтримується глобальною спільнотою дописувачів, які невтомно працюють над оновленням та вдосконаленням карти [18]. OpenStreetMap (OSM) унікальна тим, що її повністю створюють волонтери, які додають інформацію про дороги, будівлі, пам'ятки та інше.

OSM використовується приватними особами, організаціями та урядами по всьому світу для широкого спектру цілей, від навігації та міського планування до реагування на стихійні лиха та гуманітарної допомоги. Однією з ключових переваг OpenStreetMap є її відкритість і доступність, що означає, що будь-хто може зробити свій внесок у створення карти і користуватися нею безкоштовно. Це робить її цінним ресурсом для громад та організацій, які не мають доступу до комерційних картографічних даних.

На відміну від інших картографічних сервісів, OpenStreetMap надає гнучку платформу, яка дозволяє користувачам додавати власні дані та адаптувати карту до своїх конкретних потреб. Це робить його ідеальним вибором для організацій, яким потрібно створювати власні карти для своїх проектів. Крім того, дані OSM доступні безкоштовно, що є значною перевагою над комерційними картографічними сервісами.

OSM постійно розвивається, а дописувачі щодня додають нову

інформацію на карту. Це означає, що карта завжди актуальна і точна, відображаючи останні зміни у світі. Як результат, OSM часто є більш детальною і точною, ніж комерційні картографічні сервіси, що робить її ідеальним вибором для тих, кому потрібні надійні та актуальні картографічні дані.

Ще однією перевагою OSM є гнучкість, яку вона пропонує. Оскільки карта має відкритий вихідний код, користувачі можуть налаштовувати її відповідно до своїх конкретних потреб. Це означає, що вони можуть додавати або видаляти шари, змінювати колірну схему і навіть створювати власні карти на основі даних OSM. Така кастомізація стала можливою завдяки відкритому коду проекту, який дозволяє користувачам отримувати доступ до коду, що лежить в основі карти, і змінювати його.

Загалом, OpenStreetMap є цінним ресурсом для всіх, хто потребує надійних та актуальних картографічних даних. Його відкритість і доступність у поєднанні з гнучкістю і точністю роблять його ідеальною платформою для спільного картографування. Незалежно від того, чи ви приватна особа, чи організація, чи державна установа, OSM є корисним інструментом, який може допомогти вам орієнтуватися в навколишньому світі та розуміти його суть. Завдяки великій та різноманітній спільноті дописувачів, OSM впевнено продовжуватиме рости та розвиватися, надаючи точні та актуальні картографічні дані на довгі роки.

Leaflet – це бібліотека JavaScript з відкритим вихідним кодом для створення інтерактивних карт [19]. Вона розроблена як легка та модульна, що робить її простою у використанні та налаштуванні. За допомогою Leaflet розробники можуть створювати красиві та функціональні карти, які можна інтегрувати в будь-який веб-додаток.

Однією з ключових особливостей Leaflet є її простота. Бібліотека розроблена так, щоб бути легкою у використанні та розумінні, що робить її доступною для розробників усіх рівнів. Її модульна архітектура також

дозволяє легко налаштовувати і розширювати її, дозволяючи розробникам додавати нові функції і можливості до своїх карт.

Ще однією чудовою особливістю Leaflet є його сумісність з широким спектром джерел даних та плагінів. Це дозволяє легко додавати на карту маркери, шари та інші елементи, а також інтегрувати дані із зовнішніх джерел, таких як файли GeoJSON або веб-сервіси [20].

Leaflet також пропонує відмінну продуктивність і масштабованість, що робить його ідеальним для створення карт, які можуть обробляти велику кількість точок даних і взаємодій. Завдяки невеликому розміру та ефективному механізму рендерингу карти, створені за допомогою Leaflet, швидко завантажуються і плавно реагують на введення даних користувачем, навіть на малопотужних пристроях.

Загалом, Leaflet – це універсальна бібліотека, яка надає розробникам все необхідне для створення красивих та функціональних карт. Її простота, модульність і сумісність із зовнішніми джерелами даних і плагінами роблять її чудовим вибором для тих, хто хоче створювати високоякісні карти для своїх веб-додатків.

Для поєднання Leaflet та React було використано бібліотеку react-leaflet. react-leaflet – це бібліотека React для створення інтерактивних карт. Це легка та проста у використанні бібліотека, яка забезпечує простий спосіб інтеграції карт Leaflet у React-додатки. За допомогою React-Leaflet розробники можуть легко додавати маркери, спливаючі вікна та інші інтерактивні елементи до своїх карт. Він також надає набір корисних компонентів з коробки, таких як регулятор масштабу та регулятор масштабу. React-Leaflet побудований на основі Leaflet, популярної бібліотеки JavaScript з відкритим вихідним кодом для інтерактивних карт.

2.9 Бібліотека Redux

Для керування станом програми було використано бібліотеку Redux [21]. Redux – це популярна бібліотека JavaScript, яка використовується для керування станом програми. Вона слугує передбачуваним контейнером станів, який допомагає розробникам писати додатки, що є послідовними, працюють у різних середовищах та легко тестуються. Redux надає централізоване місце для управління станом, що полегшує розробникам міркування про зміни стану та їх вплив на роботу програми.

Однією з ключових переваг використання Redux є те, що він допомагає гарантувати, що стан програми є передбачуваним і легким для розуміння. Це пов'язано з тим, що Redux дотримується суворого набору правил, які визначають, як слід вносити зміни в стан і як вони повинні поширюватися по всьому додатку. Така передбачуваність полегшує розробникам розуміння того, як працює їхній додаток, і налагодження будь-яких проблем, що виникають.

Ще однією перевагою Redux є те, що його можна використовувати з широким спектром JavaScript-фреймворків та бібліотек, включаючи React, Angular та Vue.

Загалом, Redux є корисним інструментом для управління станом додатку і пропонує ряд переваг для розробників, які прагнуть створювати складні, масштабовані додатки. Незалежно від того, чи ви створюєте невеликий веб-додаток, чи велику складну систему, Redux може допомогти вам керувати станом у послідовний, передбачуваний спосіб, що полегшує підтримку вашої кодової бази з часом.

Також було використано бібліотеку React-Redux, яка забезпечує зв'язок між бібліотекою управління станами Redux та бібліотекою React. Вона дозволяє легко підключити React-компонент до сховища Redux,

надаючи компоненту доступ до стану та відправлення дій. React-Redux також надає набір утиліт, які полегшують керування станом у React-додатку. Використовуючи React-Redux, розробники можуть легко інтегрувати Redux у свої React-додатки та скористатися перевагами можливостей управління станами, які надає Redux.

2.10 Бібліотека React Router DOM

React Router DOM – це популярна та широко використовувана бібліотека в екосистемі React, станом на червень 2023 року, вона має більше 8 мільйонів завантажень щотижня на NPM [22]. Вона призначена для того, щоб допомогти розробникам керувати маршрутизацією на стороні клієнта в їхніх односторінкових додатках.

У традиційних веб-додатках використовувалася маршрутизація на стороні сервера, що означало, що кожного разу, коли користувач натискав на посилання або переходив на нову сторінку, браузер надсилав запит на сервер, а сервер повертав HTML-сторінку. Такий підхід мав ряд недоліків, включаючи повільніше завантаження, меншу інтерактивність та менш плавний користувацький досвід. Для вирішення цієї проблеми була використана бібліотека React Router DOM. React Router DOM – це популярна та широко використовувана бібліотека в екосистемі React. Вона призначена для того, щоб допомогти розробникам керувати маршрутизацією на стороні клієнта в їхніх односторінкових додатках.

React Router DOM вирішує проблеми, описані вище, дозволяючи розробникам керувати маршрутизацією на стороні клієнта. Це означає, що браузер може обробляти навігацію без необхідності надсилати запити до сервера.

Бібліотека надає набір компонентів, які дозволяють розробникам визначати правила маршрутизації для своїх додатків. Ці компоненти працюють разом, щоб створити набір правил, які визначають, які

компоненти мають бути відображені на основі поточної URL-адреси.

Однією з чудових можливостей React Router DOM є можливість визначати вкладені маршрути. Це означає, що розробники можуть з легкістю створювати складні, багатосторінкові додатки. React Router DOM також пропонує спосіб передачі параметрів через URL-адреси, що дозволяє розробникам створювати динамічні та настроювані маршрути.

Використовуючи React Router DOM, розробники можуть створювати динамічні, адаптивні додатки, які дозволяють користувачам переходити між різними сторінками без необхідності перезавантажувати весь додаток. Це може призвести до набагато більш плавного користувацького досвіду в цілому, оскільки користувачі можуть швидко і легко переходити між різними розділами програми без необхідності чекати на завантаження нових сторінок.

React Router DOM широко використовується в React-спільноті і сумісний з іншими популярними бібліотеками та фреймворками, такими як Redux, Next.js та Gatsby. Це інструмент, який допомагає розробникам створювати сучасні веб-додатки за допомогою React, що призводить до швидшого завантаження, кращої інтерактивності та більш безшовного користувацького досвіду в цілому. React Router DOM – це обов'язкова бібліотека для розробників, які створюють односторінкові додатки на React. Вона надає простий спосіб керувати маршрутизацією на стороні клієнта, дозволяючи розробникам легко створювати складні багатосторінкові додатки, що призводить до покращення користувацького досвіду та пришвидшення часу завантаження.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ

Для створення користувацького інтерфейсу бази даних геокоординат у вигляді односторінкового застосунку була розроблена схема, яка показує варіанти дій користувача (рисунок 3.1).

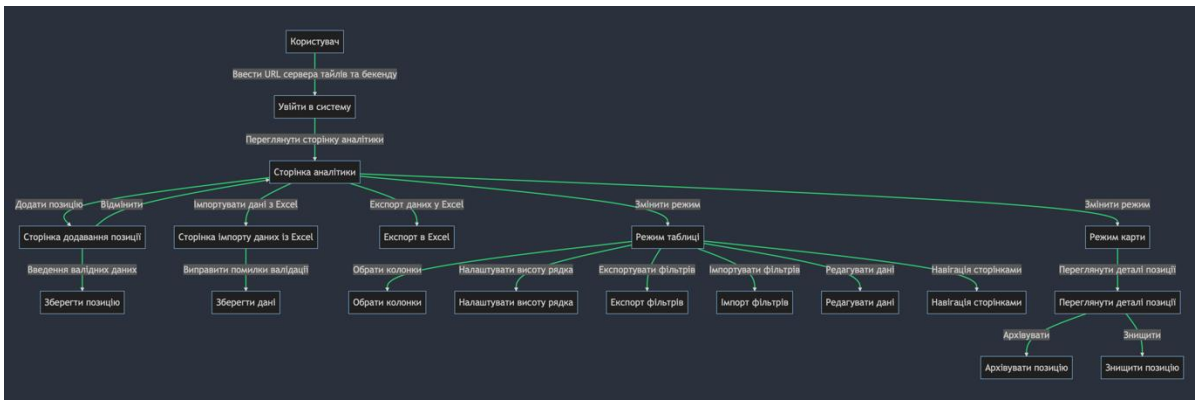

Рисунок 3.1 – Схема дій користувача

Користувацький інтерфейс було розроблено з дотримання принципів Material Design [17]. Material Design – це мова дизайну, розроблена компанією Google, яка має на меті забезпечити послідовний та інтуїтивно зрозумілий користувацький досвід на всіх платформах і пристроях. Вона базується на принципах поліграфічного дизайну і використовує плаский стиль дизайну зі сміливими кольорами та простою типографікою.

Material Design надає набір рекомендацій та інструментів для дизайнерів і розробників для створення візуально привабливих і функціональних додатків. Він наголошує на використанні сміливих кольорів, простої типографіки та мінімалістичних елементів дизайну для створення чистого та сучасного вигляду.

Одним із ключових принципів Material Design є використання "піднесення" для створення відчуття глибини та розмірності. Це досягається шляхом додавання тіней і відблисків до елементів дизайну, завдяки чому вони ніби зависають над або під поверхнею екрану.

Material Design також надає набір попередньо визначених

компонентів інтерфейсу, таких як кнопки, текстові поля та меню, які можна легко налаштувати відповідно до потреб конкретної програми. Ці компоненти розроблені таким чином, щоб бути узгодженими на всіх платформах і пристроях, полегшуючи користувачам навігацію та взаємодію з додатками.

Окрім рекомендацій щодо дизайну, Material Design також надає розробникам набір інструментів і ресурсів для реалізації цих рекомендацій у своїх додатках. Серед них бібліотека Material Design для Android, бібліотека Material Design Lite для веб-додатків і бібліотека Material Design Icons для додавання іконок у додатки.

Сторінка аналітики має 2 режима:

Режим таблиці, має вигляд (рисунок 3.2)

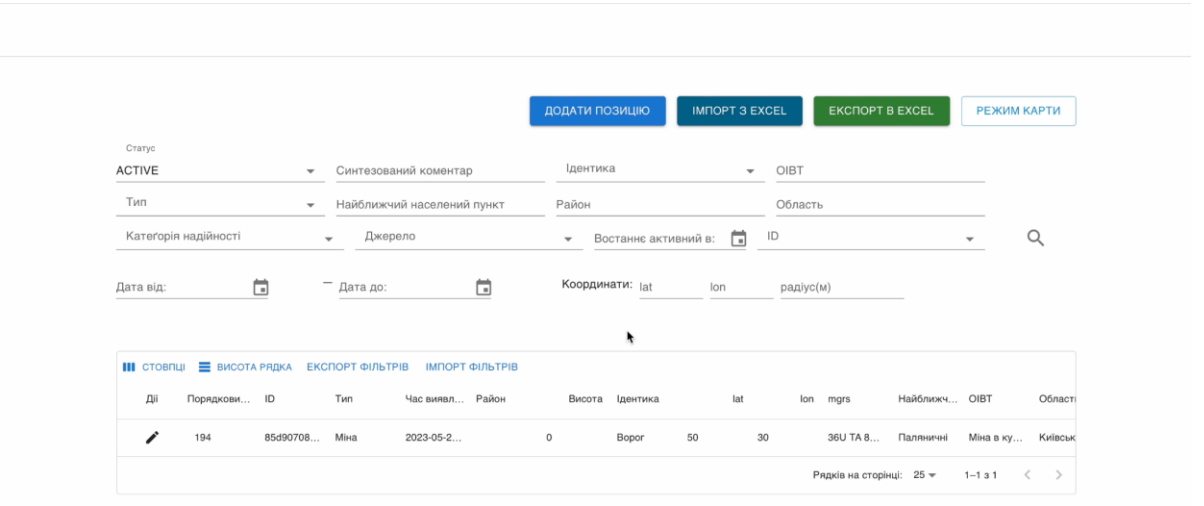

Рисунок 3.2 – Інтерфейс режиму таблиці

Було використано сітку даних з бібліотеки @mui/x-data-grid в якості таблиці для відображення точок. Даний вибір було зроблено через відповідність її стилю Material Design та такі функції, як вбудовану віртуалізацію рядків та стовпчиків, що дозволяє легко працювати з великими обсягами даних, що є дуже важливим при роботі з геопросторовими даними. При цьому було розроблено обгортку для більш гнучкої та декларативної конфігурації стовпчиків таблиці. Режим карти,

який має вигляд (рисунок 3.3)

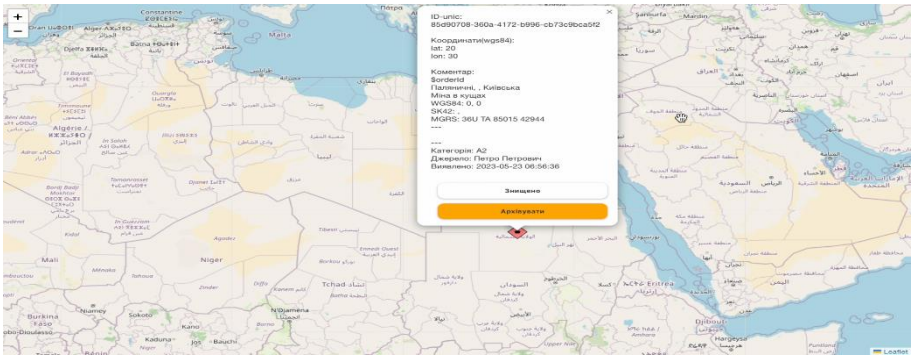

Рисунок 3.3 – Інтерфейс режиму карти

В режимі карти є можливість вибору окремих точок для перегляду повної інформації про них або їх знищення чи архівування, що дозволяє швидко та зручно реагувати на зміну обставин (рисунок 3.4).

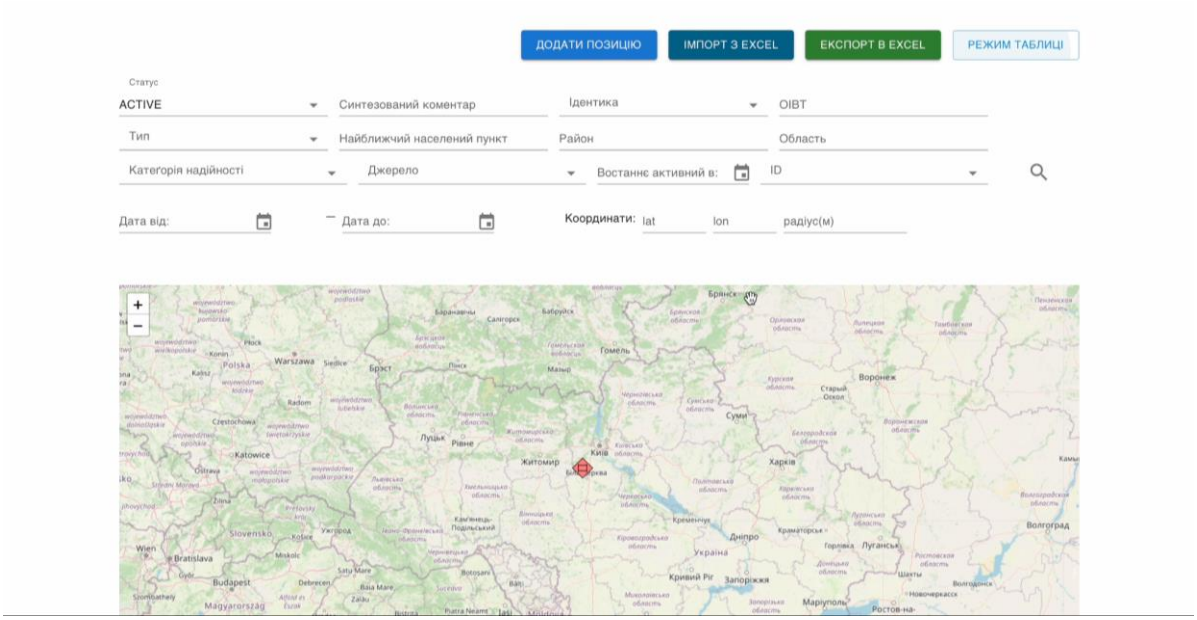

Рисунок 3.4 – Інтерфейс перегляду окремої точки на карті

Таким чином, ця система надає користувачам широкий набір функціональних можливостей для роботи з геоданими, включаючи додавання нових позицій, редагування, імпорт та експорт даних, а також зручну навігацію та відображення інформації в різних режимах – таблиця та карта.

РОЗДІЛ 4. ВИКОРИСТАННЯ СИСТЕМИ КЕРУВАННЯ ТОЧОК ГЕОКООРДИНАТ

При вході в систему, користувачу потрібно вказати URL серверу тайлів та бекенду, після чого він опиниться на сторінці аналітики в режимі таблиці, відбудеться автоматичне витягування першої сторінки даних із бази даних. Панель інструментів сторінки має наступні кнопки:

- **Додати позицію**. Переносить користувача на сторінку додавання нової позиції, де йому потрібно буде валідно ввести всі необхідні поля, для того щоб зберегти позицію в базу даних або користувач може натиснути на кнопку “Скасувати”, щоб припинити створення позиції та повернутися на сторінку аналітики.
- **Імпорт з Excel**. Відкриває діалог вибору файлу, де потрібно обрати файл з розширенням .xls, після чого користувач буде перенаправлений на сторінку редагування імпортованих даних.
- **Експорт в Excel**. При натисканні, на комп’ютер користувача буде завантажено файл з усіма позиціями з таблиці.
- **Режим таблиці/карти**. Кнопка "Режим таблиці" змінює відображення на сторінці з таблиці на карту та навпаки, надаючи користувачам різноманітні способи представлення геоданих.

Таблиця має власну панель інструментів з даними функціями:

- **Вибір стовпців.** Випадаючий список, який дозволяє користувачам вибрати, які стовпці вони хочуть відобразити або приховати в сітці. Він пропонує гнучкість у налаштуванні видимих стовпців відповідно до уподобань користувача.
- **Висота рядка.** Надає зручний спосіб налаштувати вертикальний розмір кожного рядка в сітці. Змінюючи висоту рядка, ви можете точно налаштувати візуальний вигляд і макет сітки, щоб краще відповідати вимогам дизайну і вмісту вашого додатка.
- **Експорт фільтрів:** При натисканні відкривається діалог

вибору ім’я файлу для збереження всіх виставлених фільтрів.

- **Імпорт фільтрів:** Відкриває діалог вибору файлу, де потрібно обрати коректний файл з розширенням .json, після чого будуть виставлені фільтри, відповідно до цього файлу.

Також таблиця має інші елементи керування, зокрема:

- **Кнопка редагування:** Ця кнопка знаходиться в першому стовпчику таблиці та дозволяє користувачам змінювати існуючі дані в таблиці.
- **Керування сторінками:** Таблиця містить опції пагінації, такі як кнопки навігації сторінками та випадаючий список для вибору кількості рядків на сторінці. Ці елементи керування полегшують навігацію по даних.

В режимі карти при натисканні на окрему точку(позицію) відкривається меню з детальною інформацією про неї та кнопками “Архівувати” та “Знищити”, при натисканні на які точка буде переведена у відповідний статус.

На сторінці додавання позиції (рисунок 4.1) користувачу треба увести всі необхідні дані коректно для того, щоб він міг додати цю позицію до бази даних

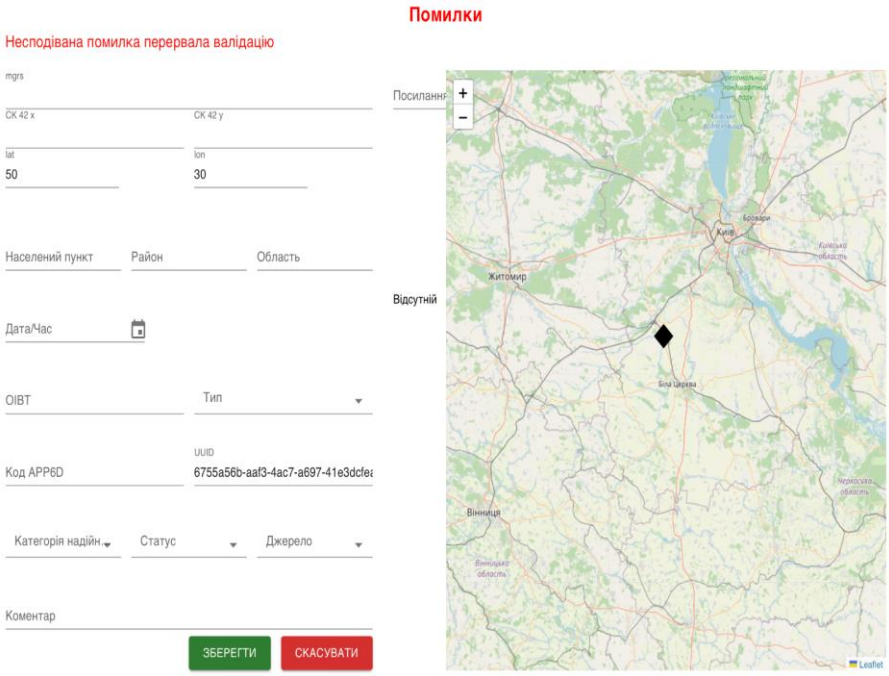

Рисунок 4.1 – Сторінка додавання позиції

На сторінці редагування імпортованих даних (рисунок 4.2) користувачу необхідно виправити всі помилки валідації, після чого він зможе натиснути кнопку “Зберегти” і дані будуть збережені в базу даних

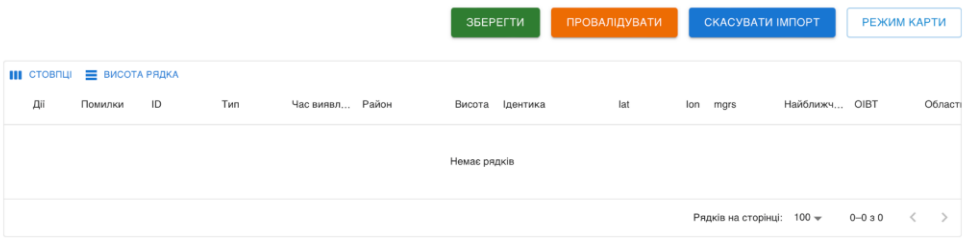

Рисунок 4.2 – Сторінка редагування імпортованих даних

У системі аналітики геоданих користувачам надається зручний інтерфейс для роботи з даними. Вхід в систему вимагає вказання URL Тайл Серверу та бекенду. Після успішного входу, користувач потрапляє на сторінку аналітики, де може переглядати дані в режимі таблиці.

На сторінці аналітики користувач має доступ до різних функцій. Він може додавати нові позиції, переходячи на сторінку додавання та

валідувати введені дані перед збереженням. Імпорт та експорт даних у форматі Excel також доступні, що дозволяє зручно обмінюватись та аналізувати геодані.

Також користувач може перемикатись між режимами таблиці та карти. Режим таблиці надає можливість вибору стовпців для відображення та налаштування висоти рядків. Таблиця також підтримує експорт та імпорт фільтрів для зручного управління даними.

У режимі карти користувач може переглядати точки та отримувати детальну інформацію про них. Є можливість архівування або видалення точок з відповідними кнопками у меню.

Таким чином, система надає широкі можливості для роботи з геоданими, зручний інтерфейс і функціонал, що дозволяють користувачам ефективно аналізувати, редагувати та управляти геоданими.

ВИСНОВКИ

В ході виконання цієї роботи було розроблено користувацький інтерфейс застосунку інтерактивної бази точок геокоординат, здатної працювати автономно.

Метою кваліфікаційної роботи була задана розробка зручного та ефективного користувацького інтерфейсу для керування точками інтерактивної бази точок геокоординат.

Відповідно до поставленої мети були виконані наступні завдання:

- проведено огляд існуючих застосунків на сучасному ринку;
- спроєктовано інтерфейс користувача;
- розроблено візуальний дизайн інтерфейсу;
- проведено тестування.

Враховуючи ці фактори, розробка геоінформаційних систем з можливістю автономної роботи та забезпеченням надійної та універсальної бази даних геокоординат стає вкрай актуальною. Це дозволить забезпечити неперервну роботу з геоінформаційними системами, незалежно від наявності стабільного зв'язку з Інтернетом та географічного розташування користувачів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Google Maps [Електронний ресурс] – Режим доступу до ресурсу: https://www.google.com/maps.
2. Mapbox [Електронний ресурс] – Режим доступу до ресурсу: https://www.mapbox.com.
3. MapQuest [Електронний ресурс] – Режим доступу до ресурсу: https://www.mapquest.com.
4. Combat Vision [Електронний ресурс] – Режим доступу до ресурсу: https://combat.vision/.
5. Gis Arta [Електронний ресурс] – Режим доступу до ресурсу: https://gisarta.org/en/index.html.
6. Bing Maps [Електронний ресурс] – Режим доступу до ресурсу: https://www.bing.com/maps.
7. Landmines and Explosive Remnants of War in Ukraine Will Take Decades to Clear [Електронний ресурс] – Режим доступу до ресурсу: https://www.kyivpost.com/post/14439.
8. Figma [Електронний ресурс] – Режим доступу до ресурсу: https://www.figma.com.
9. WebStorm [Електронний ресурс] – Режим доступу до ресурсу: https://www.jetbrains.com/webstorm/.
10. TypeScript [Електронний ресурс] – Режим доступу до ресурсу: https://www.typescriptlang.org.
11. JavaScript [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Web/javascript.
12. 40 Node JS Statistics 2023: Market Share and Popularity [Електронний ресурс] – Режим доступу до ресурсу: https://codeless.co/node-js-statistics/.
13. NPM [Електронний ресурс] – Режим доступу до ресурсу: https://www.npmjs.com.

14. React [Електронний ресурс] – Режим доступу до ресурсу: https://react.dev.
15. Meta [Електронний ресурс] – Режим доступу до ресурсу: https://about.meta.com.
16. The Power of React Components: Creating Reusable UI Elements [Електронний ресурс] – Режим доступу до ресурсу: https://www.lindritsulaj.com/blog/the-power-of-react-components-creating-reusable-ui-elements.
17. Material Design [Електронний ресурс] – Режим доступу до ресурсу: https://m3.material.io/.
18. OpenStreetMap [Електронний ресурс] – Режим доступу до ресурсу: https://www.openstreetmap.org/about.
19. Leaflet [Електронний ресурс] – Режим доступу до ресурсу: https://leafletjs.com.
20. Leaflet plugins [Електронний ресурс] – Режим доступу до ресурсу: https://leafletjs.com/plugins.html.
21. Redux [Електронний ресурс] – Режим доступу до ресурсу: https://redux.js.org.
22. Кількість навантажень React Router DOM [Електронний ресурс] – Режим доступу до ресурсу: https://www.npmjs.com/package/react-router-dom.

ДОДАТКИ

Додаток А
Код обгортки для конфігурації стовпців

```
export type Position = {
  category: string;
  detectionTime: string;
  district: string;
  height: number;
  id: string;
  identity: string;
  lat: number;
  link: string;
  lon: number;
  mgrs: string;
  mostCloseSettlement: string;
  oivt: string;
  region: string;
  reliabilityCategory: string;
  sk42_x: string;
  sk42_y: string;
  source: string;
  status: string;
  summaryComment: string;
  comment: string;
  app6d: number | `${number}`;
};

export type Column = Exclude<keyof Position, "link">;

export type ColumnConfig = {
  headerName: string;
  type: string;
} & Partial<GridColDef>;

export type ColumnsRecord<TColumn extends string | number | symbol> =
Record<
  TColumn,
```

```
  ColumnConfig
>;

type LastElement<T> = T extends [...unknown[], infer LastItem]
  ? LastItem
  : never;

type Operator<A, B> = (value: A) => B;
// type OperatorA<T> = T extends Operator<infer A, any> ? A : never;
type OperatorB<T> = T extends Operator<any, infer B> ? B : never;

type PipeOperators<Operators extends unknown[], Input> = Operators extends
[
  infer Item,
  ...infer Tail
]
  ? [Operator<Input, OperatorB<Item>>, ...PipeOperators<Tail,
OperatorB<Item>>]
  : Operators;
type PipeOperatorsOutput<Operators extends unknown[]> = OperatorB<
  LastElement<Operators>
>;

export const columnsBase = {
  id: {
    headerName: "ID",
    type: "string",
  },
  category: {
    headerName: "Тип",
    type: "string",
  },
  detectionTime: {
    headerName: "Час виявлення",
    type: "dateTime",
  },
  district: {
```

```
  headerName: "Район",
  type: "string",
},
height: {
  headerName: "Висота",
  type: "number",
},
identity: {
  headerName: "Ідентика",
  type: "string",
},
lat: {
  headerName: "lat",
  type: "number",
},
lon: {
  headerName: "lon",
  type: "number",
},
mgrs: {
  headerName: "mgrs",
  type: "string",
},
mostCloseSettlement: {
  headerName: "Найближчий НП",
  type: "string",
},
oivt: {
  headerName: "OIBT",
  type: "string",
},
region: {
  headerName: "Область",
  type: "string",
},
reliabilityCategory: {
  headerName: "Категорія надійності",
```

```
    type: "string",
  },
  sk42_x: {
    headerName: "sk42_x",
    type: "string",
  },
  sk42_y: {
    headerName: "sk42_y",
    type: "string",
  },
  source: {
    headerName: "Джерело",
    type: "string",
  },
  status: {
    headerName: "Статус",
    type: "string",
  },
  summaryComment: {
    headerName: "Синтезований коментар",
    type: "string",
  },
  comment: {
    headerName: "Коментар",
    type: "string",
  },
  link: {
    headerName: "Посилання",
    type: "string",
  },
  app6d: {
    headerName: "app6d",
    type: "string",
  },
};

export const mapColumns =
```

```
  <
    TColumn1 extends string | number | symbol,
    TColumn2 extends string | number | symbol = TColumn1
  >(
    fn: ({ name, config }: { name: TColumn1; config: ColumnConfig }) => {
      name: TColumn2;
      config: ColumnConfig;
    }
  ) =>
  (columns: ColumnsRecord<TColumn1>) =>
    Object.entries(columns).reduce((acc, [name, config]) => {
      const { name: newName, config: newConfig } = fn({
        name: name as TColumn1,
        config: config as ColumnConfig,
      });
      return { ...acc, [newName]: newConfig };
    }, {}) as ColumnsRecord<TColumn2>;

export const filterColumns =
  <TColumn extends string | number | symbol>(
    fn: ({ name, config }: { name: TColumn; config: ColumnConfig }) =>
boolean
  ) =>
  (columns: ColumnsRecord<TColumn>) =>
    Object.entries(columns).reduce(
      (acc, [name, config]) =>
        fn({
          name: name as TColumn,
          config: config as ColumnConfig,
        })
          ? { ...acc, [name]: config }
          : acc,
      {}
    ) as ColumnsRecord<TColumn>;

export const appendColumn =
  <
```

```
    TColumn extends string | number | symbol,
    TAddedColumn extends string | number | symbol
  >({
    name,
    config,
  }: {
    name: TAddedColumn;
    config: ColumnConfig;
  }) =>
  (columns: ColumnsRecord<TColumn>) =>
    ({
      ...columns,
      [name]: config,
    } as ColumnsRecord<TColumn | TAddedColumn>);

export const prependColumn =
  <
    TColumn extends string | number | symbol,
    TAddedColumn extends string | number | symbol
  >({
    name,
    config,
  }: {
    name: TAddedColumn;
    config: ColumnConfig;
  }) =>
  (columns: ColumnsRecord<TColumn>) =>
    ({
      [name]: config,
      ...columns,
    } as ColumnsRecord<TColumn | TAddedColumn>);

export const patchColumn =
  (fn: (column: ColumnConfig) => ColumnConfig) =>
  <TColumn extends string | number | symbol>(name: TColumn) =>
    mapColumns((column) =>
      column.name === name ? { name, config: fn(column.config) } : column
```

```
    );

export const pipeColumns =
  <TColumnInput extends string | number | symbol, Operators extends
unknown[]>(
    ...operators: PipeOperators<Operators, ColumnsRecord<TColumnInput>>
  ): ((input: ColumnsRecord<TColumnInput>) =>
PipeOperatorsOutput<Operators>) =>
  (columns) =>
    operators.reduce(
      (acc, fn) => (fn as (val: unknown) => unknown)(acc),
      columns
    ) as PipeOperatorsOutput<Operators>;

export const columnsToArray = <TColumn extends string | number | symbol>(
  columns: ColumnsRecord<TColumn>
): GridColDef[] =>
  Object.entries(columns).map(([name, config]) => ({
    field: name,
    ...(config as ColumnConfig),
  }));
```