Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

бакалавра
(назва освітньо-кваліфікаційного рівня)

студент *Брова Владислав Анатолійович*
(ПІБ)

академічної групи *121-20-2*
(шифр)

спеціальності *121 Інженерія програмного забезпечення*
(код і назва спеціальності)

освітньої програми *Інженерія програмного забезпечення*
(назва освітньої програми)

на тему: *Розробка програмного забезпечення веб-застосунку візуалізації даних агромоніторингу*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**Інституційною**	
кваліфікаційної роботи	*проф. Лактіонов І.С.*			
розділів:				
спеціальний	*проф. Лактіонов І.С.*			
економічний	*доц. Касьяненко Л.В.*			
Рецензент				
Нормоконтролер	*ст.викл. Мартиненко А.А.*			

Дніпро
2024

Міністерство освіти і науки України
НТУ «Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
завідувач кафедри
програмного забезпечення комп'ютерних систем
(повна назва)

_____ М.О. Алексєєв
(підпис) (прізвище, ініціали)

« » _____ 2024 року

ЗАВДАННЯ
на кваліфікаційну роботу

бакалавра
(назва освітньо-кваліфікаційного рівня)

студент *121-20-2* *Брова Владислав Анатолійович*
(група) (прізвище та ініціали)

тема кваліфікаційної роботи *Розробка програмного забезпечення веб-застосунку візуалізації даних агромоніторингу*

затверджена наказом ректора НТУ «ДП» від *23.05.2024* № *469-с*

Розділ	Зміст виконання	Термін виконання
Спеціальний	***На основі матеріалів проєктно-технологічної та інших науково-технічних джерел провести аналіз стану рішення проблеми та постановку задачі. Обґрунтувати вибір та здійснити реалізацію методів вирішення проблеми***	***10.06.2024 р.***
Економічний	***Провести розрахунок трудомісткості розробки програмного забезпечення, витрат на створення ПЗ й тривалості його розробки***	***14.06.2024 р.***

Завдання видав _____ *проф. Лактіонов І.С*
(підпис) (посада, прізвище, ініціали)

Завдання прийняв до виконання _____ *Брова В.А.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *14.01.2024 р.*

Термін подання кваліфікаційної роботи до ЕК: *21.06.2024 р.*

РЕФЕРАТ

Пояснювальна записка: 76 с., 33 рис., 3 дод., 22 джерел.

Об'єкт розробки: веб-застосунок для візуалізації даних агромоніторингу.

Мета кваліфікаційної роботи: створення інструменту для відображення, аналізу та інтерпретації даних агромоніторингу з різних джерел для підвищення ефективності управління сільськогосподарськими ресурсами.

У вступі обґрунтовано актуальність візуалізації даних агромоніторингу, проаналізовано стан галузі, визначено мету роботи та її важливість для сільського господарства, конкретизовано завдання створення веб-застосунку.

Актуальність: Візуалізація даних агромоніторингу критично важлива для сучасного сільського господарства, дозволяючи ефективно аналізувати великі обсяги даних для прийняття обґрунтованих рішень.

У першому розділі проаналізовано предметну галузь агромоніторингу, визначено актуальність завдання, призначення веб-застосунку, вимоги до функціоналу, безпеки, технічних засобів та програмної сумісності.

У другому розділі досліджено та порівняно наявні рішення, обґрунтовано вибір платформи та технологій, виконано проектування та розробку програмного забезпечення, описано роботу веб-застосунку, його структуру, алгоритми функціонування, вхідні та вихідні дані, а також склад технічних засобів.

Практичне значення: включає аналіз наявних рішень, створення дизайну та прототипів, реалізацію веб-застосунку, тестування з оцінкою характеристик. Розробка та впровадження веб-застосунку дозволяє аграріям ефективніше управляти ресурсами, підвищувати врожайність та знижувати витрати.

В економічному розділі розраховано трудомісткість розробки програмного забезпечення, вартість створення веб-застосунку та необхідний час для його розробки.

Список ключових слів: АГРОМОНІТОРИНГ, ВІЗУАЛІЗАЦІЯ ДАНИХ, ВЕБ-ЗАСТОСУНОК, АНАЛІЗ ДАНИХ, СІЛЬСЬКЕ ГОСПОДАРСТВО, УПРАВЛІННЯ РЕСУРСАМИ, HTML, CSS, JAVASCRIPT, VUE.

ABSTRACT

Explanatory note: 76 p., 33 pic., 3 add., 22 sources.

Development Object: A web application for visualizing agricultural monitoring data.

Purpose of the Qualification Work: To create a tool for displaying, analyzing, and interpreting agricultural monitoring data from various sources to enhance the efficiency of agricultural resource management.

The introduction justifies the relevance of visualizing agricultural monitoring data, analyzes the state of the industry, defines the purpose of the work and its importance for agriculture, and specifies the tasks of creating the web application.

Relevance: Visualization of agricultural monitoring data is critically important for modern agriculture, allowing for the effective analysis of large volumes of data to make informed decisions.

The first chapter analyzes the field of agricultural monitoring, defines the relevance of the task, the purpose of the web application, the requirements for functionality, security, technical means, and software compatibility.

The second chapter investigates and compares existing solutions, justifies the choice of platform and technologies, designs and develops the software, describes the operation of the web application, its structure, functioning algorithms, input and output data, as well as the composition of technical means.

Practical Significance: Includes analysis of existing solutions, design and prototyping, implementation of the web application, testing with performance evaluation. The development and implementation of the web application enable farmers to manage resources more efficiently, increase yields, and reduce costs.

The economic section calculates the labor intensity of software development, the cost of creating the web application, and the necessary time for its development.

List of Keywords: AGRICULTURAL MONITORING, DATA VISUALIZATION, WEB APPLICATION, DATA ANALYSIS, AGRICULTURE, RESOURCE MANAGEMENT, HTML, CSS, JAVASCRIPT, VUE.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ГІС – Геоінформаційна система

СУБД – система управління базами даних

ПК – персональний комп’ютер

ВСТУП

Актуальність теми дослідження полягає у необхідності покращення процесів візуалізації та аналізу даних агромоніторингу для підвищення ефективності сільськогосподарського виробництва. Існуючі рішення в цій галузі часто мають обмежені можливості візуалізації, незручний інтерфейс або недостатню функціональність для всебічного аналізу даних.

Завдання даної кваліфікаційної роботи та об’єкт її діяльності пов'язані з напрямом підготовки «Web-технології», та відповідає загальній тематиці кваліфікаційних робіт і переліку виробничих функцій, типових задач діяльності, умінь та компетенцій, необхідних для бакалаврів освітньої програми 121 "Інженерія програмного забезпечення".

Метою кваліфікаційної роботи є створення ефективного та зручного в використанні веб-застосунку, який би забезпечував наочну візуалізацію різноманітних даних агромоніторингу, їх аналіз та підтримку прийняття рішень у сільськогосподарській сфері.

Об'єктом розробки є веб-застосунок для візуалізації даних агромоніторингу.

Задачі:

1. Дослідження теоретичних основ агромоніторингу та візуалізації даних.
2. Аналіз існуючих рішень.
3. Визначення вимог до веб-застосунку.
4. Проектування архітектури та функціональних можливостей застосунку.
5. Вибір технологій та інструментів для розробки.
6. Реалізація веб-застосунку.
7. Визначення перспектив подальшого розвитку та вдосконалення.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Загальні відомості з предметної галузі

Актуальність теми дослідження полягає у необхідності покращення процесів візуалізації та аналізу даних агромоніторингу для підвищення ефективності сільськогосподарського виробництва. Існуючі рішення в цій галузі часто мають обмежені можливості візуалізації, незручний інтерфейс або недостатню функціональність для всебічного аналізу даних, про що свідчить порівняльний аналіз наявних програмних продуктів.

На сучасному етапі розвитку сільського господарства все більшого значення набувають технології точного землеробства, які передбачають використання різноманітних даних для прийняття управлінських рішень. Дані, зібрані з різних джерел, таких як супутники, дрони, сенсори в полі, можуть значно підвищити врожайність та оптимізувати використання ресурсів. Однак, ефективне використання цих даних можливе лише за наявності зручних та функціональних інструментів для їх аналізу та візуалізації.

Сучасні технології візуалізації даних дозволяють інтерактивно відображати інформацію на картах, графіках та інших графічних об'єктах, що значно спрощує процеси моніторингу та управління. Зокрема, використання бібліотек для розробки веб-додатків, таких як Vue.js [1, 2] та Leaflet [3, 4], дозволяє створювати зручні та функціональні інтерфейси для візуалізації агромоніторингових даних. Це сприяє швидкому реагуванню на зміни в стані посівів, своєчасному внесенню добрив, проведенню поливу та інших агротехнічних заходів.

Крім того, інтеграція цих технологій дозволяє розробникам створювати динамічні, масштабовані рішення, які можуть бути адаптовані до потреб конкретного фермерського господарства або регіону. Відображення даних у

реальному часі дозволяє фермерам миттєво реагувати на зміни у стані рослин, що підвищує їхню оперативність та ефективність управління.

Глобальні виклики, такі як зміна клімату, зростання населення та необхідність забезпечення продовольчої безпеки, додають актуальності розвитку ефективних систем агромоніторингу. Вони дозволяють мінімізувати ризики, пов'язані з несприятливими погодними умовами, шкідниками та хворобами рослин, шляхом раннього виявлення проблем та оперативного вжиття заходів. В цьому контексті, візуалізація даних стає ключовим елементом, оскільки вона дозволяє спростити складні аналітичні процеси та робить дані більш доступними для кінцевих користувачів.

Таким чином, розвиток і впровадження сучасних веб-додатків для візуалізації даних агромоніторингу [5] є надзвичайно важливим для підвищення ефективності сільськогосподарського виробництва, зменшення витрат та забезпечення сталого розвитку аграрного сектору. Це дозволяє досягти більшого рівня продуктивності, забезпечуючи при цьому раціональне використання ресурсів та зниження негативного впливу на навколишнє середовище.

Існують декілька ключових аспектів, які підкреслюють важливість покращення систем візуалізації агромоніторингових даних:

1.**Підвищення точності та своєчасності рішень:** Сучасні технології дозволяють збирати величезну кількість даних про стан рослин, якість ґрунту, погодні умови тощо. Візуалізація цих даних в реальному часі допомагає аграріям приймати більш точні та своєчасні рішення, що напряму впливає на врожайність та рентабельність виробництва.

2.**Оптимізація використання ресурсів:** Ефективне управління ресурсами, такими як вода, добрива та засоби захисту рослин, є критично важливим для зниження витрат та зменшення негативного впливу на навколишнє середовище. Візуалізація даних дозволяє фермерам бачити, де і коли необхідно застосовувати ресурси, щоб досягти максимального ефекту.

3.**Підвищення ефективності праці**: Зручні та інтуїтивно зрозумілі інтерфейси для візуалізації даних сприяють більш ефективній роботі персоналу.

Це зменшує потребу в спеціальному навчанні та дозволяє швидко адаптуватися до нових умов роботи.

4.**Інтеграція з іншими системами:** Сучасні веб-додатки можуть легко інтегруватися з іншими системами управління фермою, що забезпечує єдине інформаційне середовище для всіх аспектів агровиробництва. Це дозволяє отримувати більш повну картину стану господарства та приймати комплексні рішення.

5.**Прозорість та підзвітність:** Використання веб-додатків для візуалізації даних агромоніторингу забезпечує прозорість у прийнятті рішень та поліпшує підзвітність перед партнерами, інвесторами та регуляторами. Це сприяє зміцненню довіри та підвищенню рівня співпраці.

Розглянемо детальніше популярні системи агромоніторингу:

1.EO Browser [6] – вебсайт від Sentinel Hub є потужним веб-застосунком для візуалізації та аналізу супутникових знімків Землі.

Центральним елементом інтерфейсу є інтерактивна карта, яка займає більшу частину екрану. На ній відображаються супутникові знімки, вибрані користувачем із різноманітних джерел даних, таких як Sentinel-2, Landsat, MODIS та інші місії спостереження Землі. Користувачі можуть легко переміщатися по карті, змінювати рівень масштабування та вибирати потрібні часові проміжки для аналізу.

Ліва бічна панель містить потужний навігатор, який дозволяє керувати відображенням даних на карті. Тут користувачі можуть обирати джерела даних, застосовувати різноманітні фільтри та налаштування візуалізації. Наприклад, можна вибирати конкретні діапазони хвиль, індекси рослинності, параметри атмосфери та багато інших критеріїв залежно від потреб аналізу.

Одним із ключових переваг EO Browser є можливість комбінувати різні шари даних для створення більш інформативних та наочних візуалізацій. Користувачі можуть накладати шари індексів рослинності, таких як NDVI або EVI, на природні кольори супутникових знімків, щоб краще оцінити стан сільськогосподарських угідь та виявити проблемні ділянки.

Інструменти навігації та масштабування забезпечують зручну роботу з картою, дозволяючи наближати та переміщатися до цікавих регіонів. Додаткові функції, такі як вимірювання відстаней, креслення форм та позначок, допомагають проводити більш глибокий аналіз.

EO Browser також пропонує можливість завантаження історичних даних та порівняння різних часових проміжків. Це є критично важливим для виявлення змін та тенденцій у розвитку рослинності, ерозії ґрунтів, забрудненні водойм та інших процесів, пов'язаних з агромоніторингом.

Можливості та функціонал:

1. Вибір області інтересу:

- Можливість обрати локацію на карті або шляхом введення координат.

- Інтерактивне масштабування для деталізації та панорамування для огляду великих територій (Рис. 1.1).

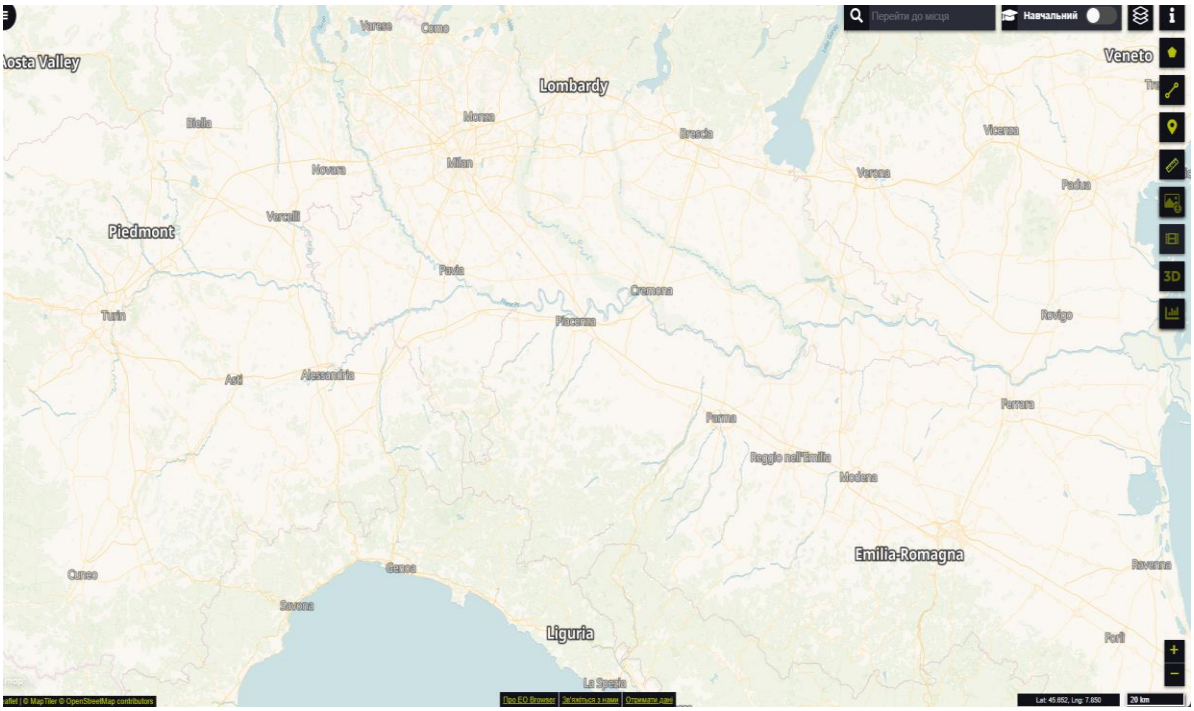

Рис. 1.1. Програма EO Browser

2. Доступ до даних:

- Підтримка даних з різних супутників, таких як Sentinel-1, Sentinel-2, Landsat, MODIS (Рис. 1.2).

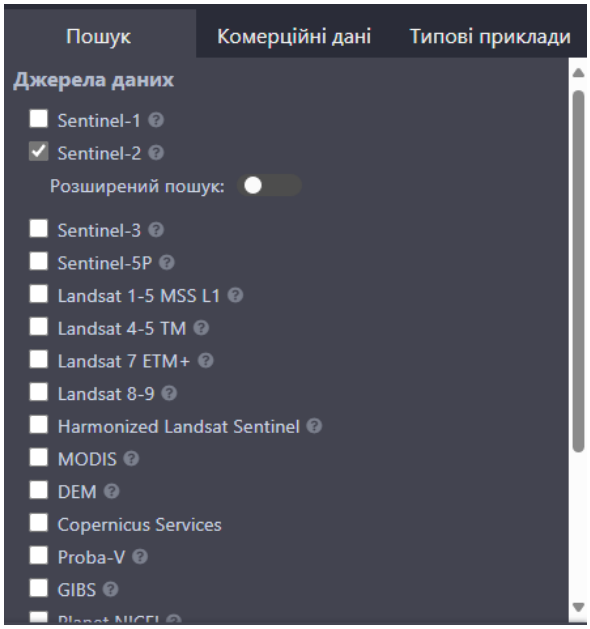

Рис. 1.2. Інтерфейс програми EO Browser

- Перегляд історичних даних (Рис. 1.3) за обраний часовий період (Рис. 1.4).

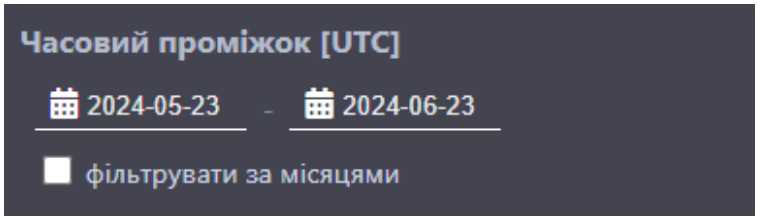

Рис. 1.3. Інтерфейс програми EO Browser

Рис. 1.4. Інтерфейс програми EO Browser

3. Візуалізація даних:

- Перегляд зображень у різних спектральних діапазонах (Рис. 1.5) для виявлення різних характеристик території (Рис. 1.6).

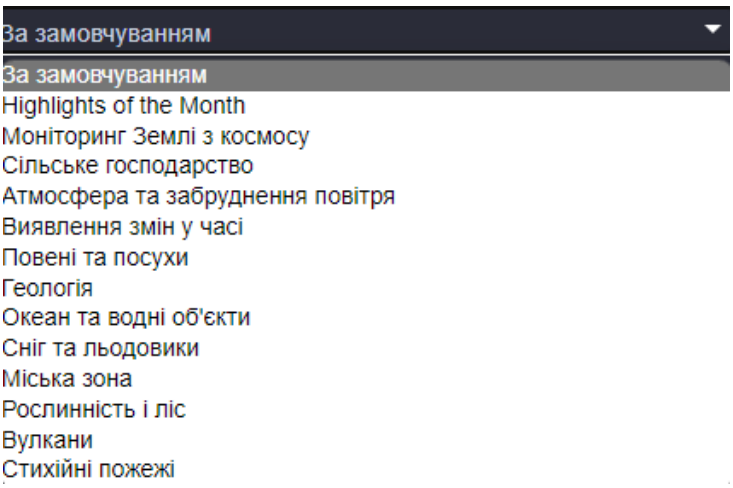

Рис. 1.5. Інтерфейс програми EO Browser

Рис. 1.6. Інтерфейс програми EO Browser

- Налаштування параметрів візуалізації для кращої видимості певних деталей (Рис. 1.7).

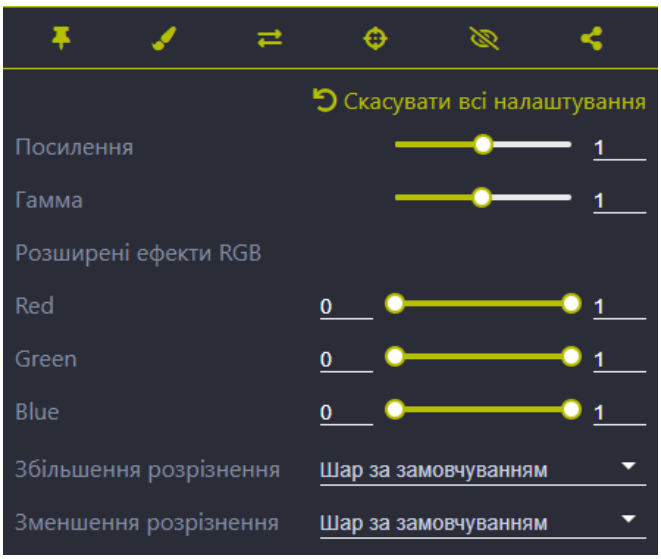

Рис. 1.7. Інтерфейс програми EO Browser

4. Інструменти аналізу:

- Інструменти для аналізу рослинного покриву, вологості ґрунту, температури поверхні та інших агрономічних показників (Рис. 1.8).

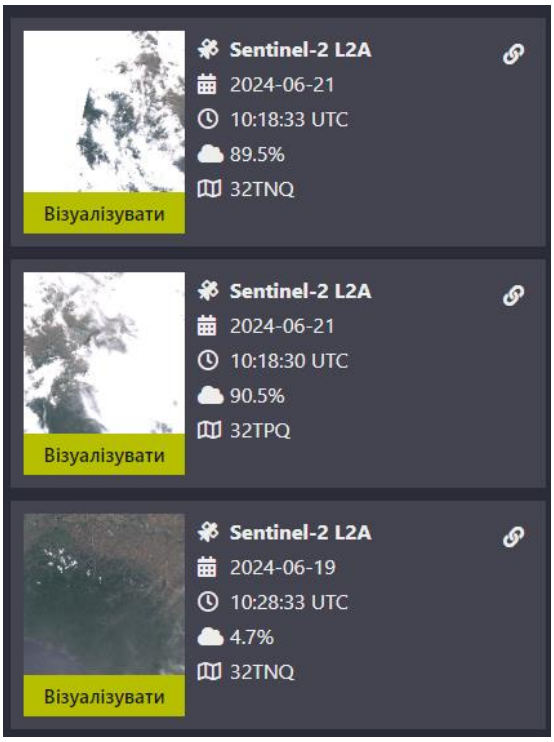

Рис. 1.8. Інтерфейс програми EO Browser

5. Експорт даних:

- Завантаження зображень та аналітичних даних у різних форматах (JPEG, GeoTIFF, PNG) (Рис. 1.9).

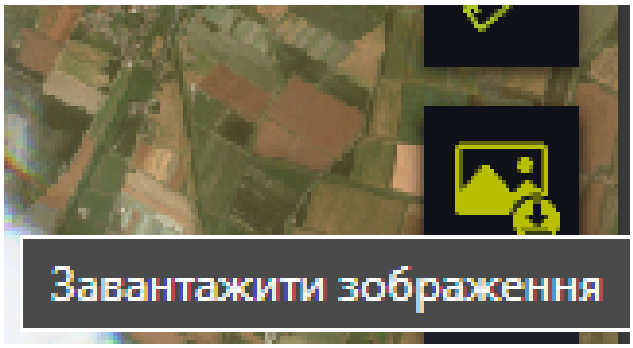

Рис. 1.9. Інтерфейс програми EO Browser

- Підтримка API для інтеграції з іншими системами та програмами.

Відповідність темі дипломної роботи:

Sentinel Hub EO Browser є ідеальним інструментом для візуалізації та аналізу супутникових даних, що робить його надзвичайно корисним для агромоніторингу. Основні функції цього веб-застосунку забезпечують ефективне відстеження та аналіз агрономічних показників:

1.**Інтерактивні карти:** Відображення агрономічних даних у вигляді інтерактивних карт дозволяє користувачам візуально аналізувати великі території, що є важливим для сільськогосподарського моніторингу.

2. **Аналіз даних:** Інструменти для аналізу рослинного покриву, вологості ґрунту та інших показників допомагають аграріям приймати обґрунтовані рішення щодо управління сільськогосподарськими ресурсами.

3.**Часовий аналіз:** Можливість переглядати історичні дані дозволяє аналізувати довгострокові тенденції та робити прогнози на майбутнє, що є важливим для планування агротехнічних заходів.

4. **Інтеграція з іншими системами:** Завдяки підтримці API, цей інструмент може бути інтегрований з іншими агромоніторинговими системами, що забезпечує комплексний підхід до управління сільським господарством.

Таким чином, Sentinel Hub EO Browser демонструє практичне застосування теоретичних принципів візуалізації та аналізу даних, описаних у дипломній роботі, для підвищення ефективності агромоніторингу.

2. Crop Monitor [7] є веб-порталом, розробленим для моніторингу стану сільськогосподарських культур у всьому світі, що робить його ідеальним рішенням для застосування в межах дипломної роботи на тему "Розробка програмного забезпечення веб-застосунку візуалізації даних агромоніторингу".

Crop Monitor Exploring Tool [8] є спеціалізованим веб-інструментом для моніторингу стану сільськогосподарських культур на глобальному рівні. Цей інструмент розроблений завдяки спільним зусиллям вчених з різних організацій, таких як Європейський Союз та Організація Об'єднаних Націй. CMET є ідеальним для використання в рамках дипломної роботи на тему "Розробка програмного забезпечення веб-застосунку візуалізації даних агромоніторингу".

Основні можливості та функціонал:

1. Інтерактивний інтерфейс карти:

- Основною складовою інтерфейсу CMET є інтерактивна карта світу, яка дозволяє користувачам переглядати різноманітні шари даних, пов'язані з сільськогосподарською діяльністю (Рис. 1.10).

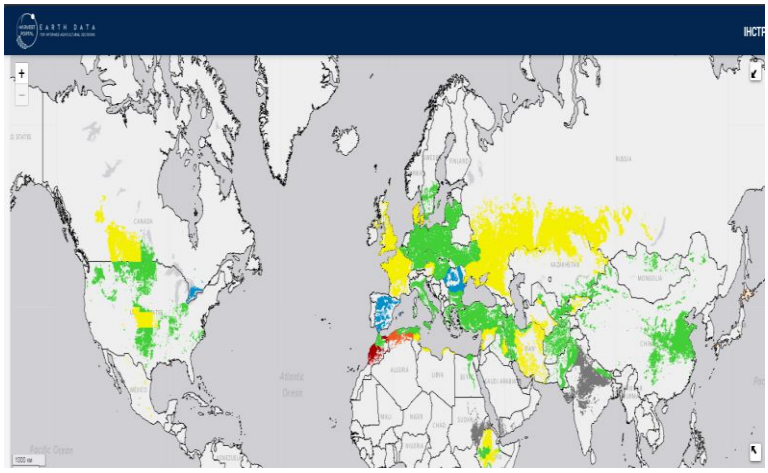

Рис. 1.10.Програма Crop Monitor Exploring Tool

- Користувачі можуть обирати конкретні культури, такі як пшениця, кукурудза, рис, та задавати часові проміжки для аналізу (Рис. 1.11).

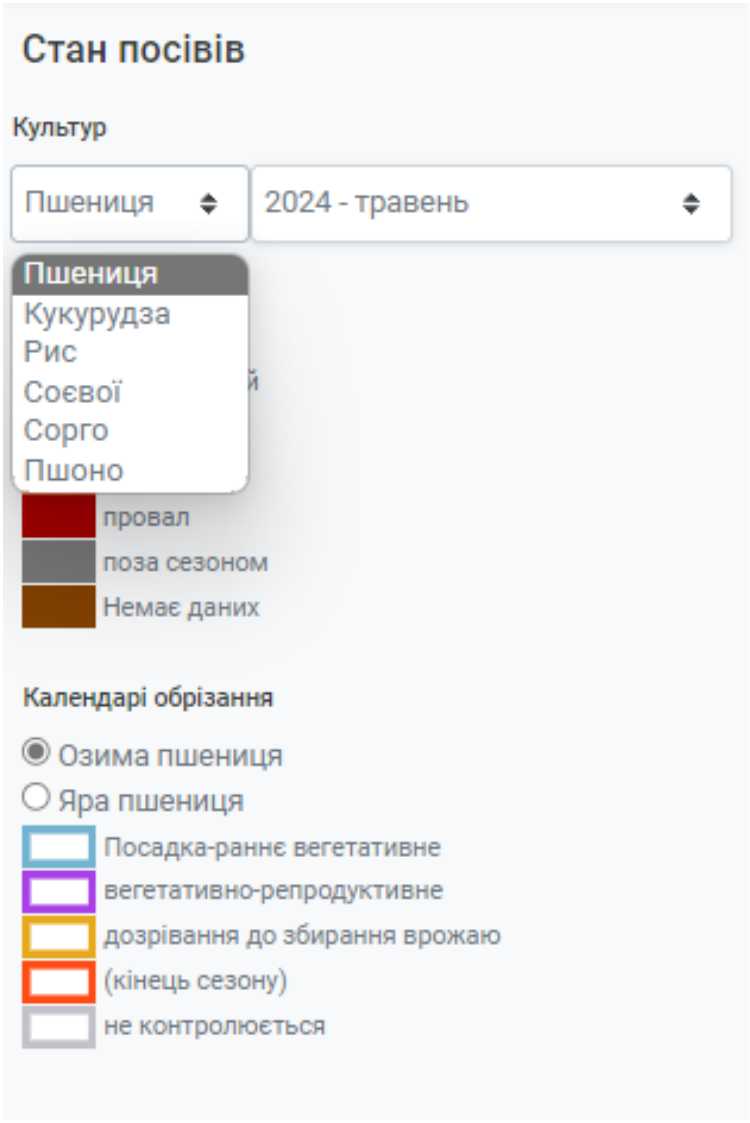

Рис. 1.11. Інтерфейс програми Crop Monitor Exploring Tool

- Карта дозволяє змінювати масштаб, переміщатися по ній та накладати різні шари даних для більш детального аналізу (Рис. 1.12).

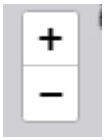

Рис. 1.12. Інтерфейс програми Crop Monitor Exploring Tool

2. Бічна панель:

- бічна панель містить численні фільтри та інструменти для налаштування відображення даних (Рис. 1.13).

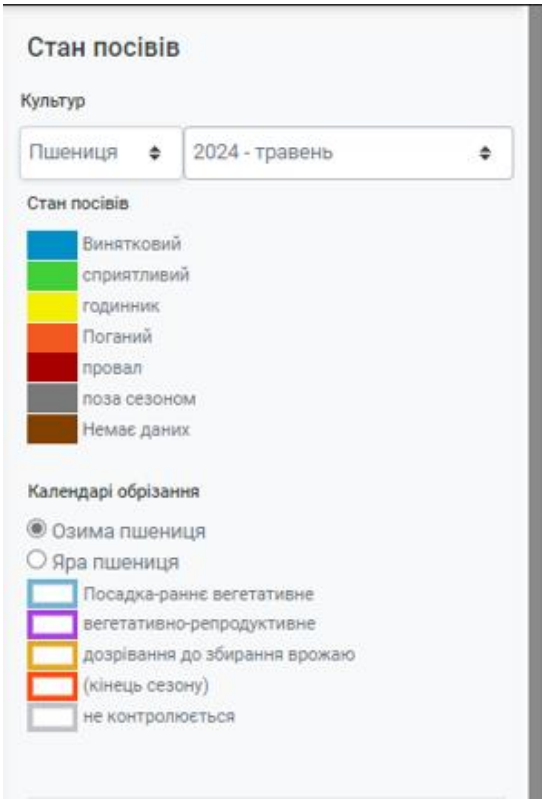

Рис. 1.13. Інтерфейс програми Crop Monitor Exploring Tool

- Користувачі можуть фільтрувати інформацію за регіонами, країнами або адміністративними одиницями, а також застосовувати фільтри за типами ґрунтів, кліматичними зонами або фазами росту рослин. Це забезпечує гнучкість та можливість детального аналізу даних (Рис. 1.14).

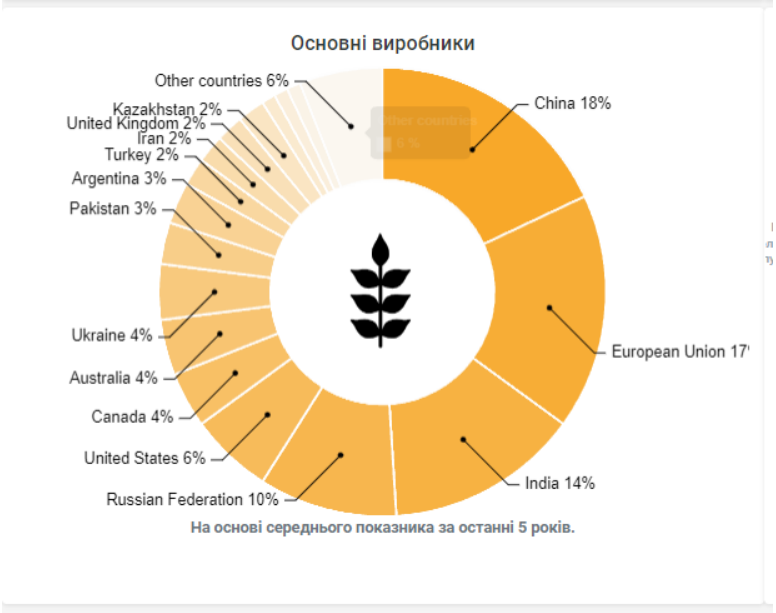

Рис. 1.14. Інтерфейс програми Crop Monitor Exploring Tool

3. Аналітичні інструменти:

- Інструменти аналізу дозволяють обчислювати статистику стану сільськогосподарських культур для заданих регіонів або окремих культур.

- Користувачі можуть отримувати дані про врожайність, динаміку змін, сезонні коливання та інші показники. Ці функції є надзвичайно корисними для агропромислового планування та управління ресурсами (Рис. 1.15).

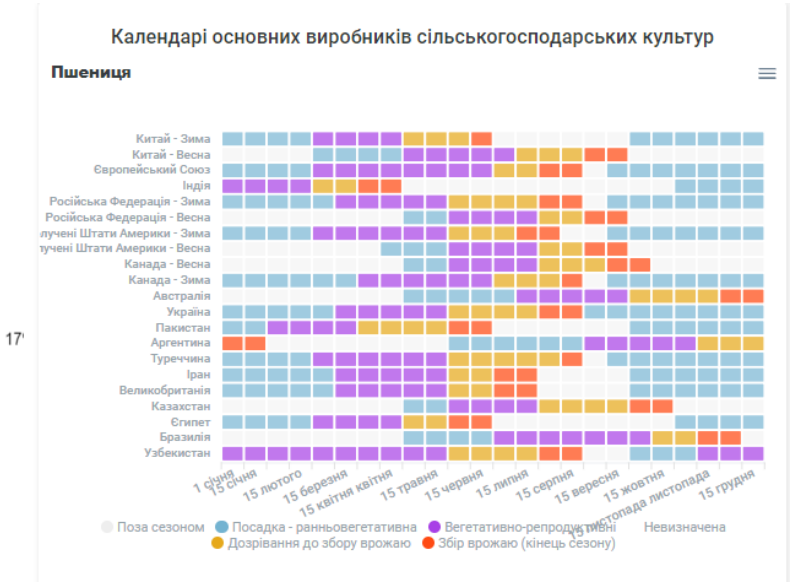

Рис. 1.15. Інтерфейс програми Crop Monitor Exploring Tool

4. Експорт та інтеграція даних:

- СМЕТ пропонує функції експорту та завантаження даних у різних форматах, включаючи растрові зображення, векторні шари та таблиці з метричними даними. Це дозволяє інтегрувати отримані результати в інші аналітичні системи або геоінформаційні програми для подальшої обробки та аналізу (Рис. 1.16).

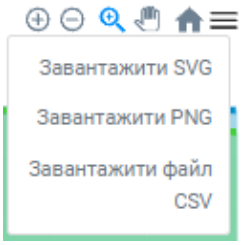

Рис. 1.16. Інтерфейс програми Crop Monitor Exploring Tool

5. Доступ до історичних даних:

- Надає доступ до архіву історичних даних, що дозволяє відстежувати тенденції та закономірності в сільському господарстві протягом тривалих періодів часу. Це є надзвичайно корисним для прогнозування врожайності, планування сівозмін та розробки стратегій управління сільськогосподарською діяльністю (Рис. 1.17).

Рис. 1.17. Інтерфейс програми Crop Monitor Exploring Tool

Загалом, Crop Monitor Exploring Tool є потужним та спеціалізованим інструментом для візуалізації та аналізу стану сільськогосподарських культур на глобальному рівні. Його функціональність, гнучкість налаштувань та можливості інтеграції з іншими системами роблять його ідеальним рішенням для веб-застосунку візуалізації даних агромоніторингу.

3.AgriSense [9] є сучасною системою агромоніторингу, яка використовує супутникові знімки, дані з безпілотних літальних апаратів та сенсорні мережі для забезпечення комплексного моніторингу стану сільськогосподарських угідь. Основною перевагою AgriSense є можливість інтеграції з іншими системами управління сільським господарством, що дозволяє отримувати повний спектр інформації про стан посівів, ґрунтові умови та кліматичні показники в реальному часі.

Візуальна частина

Головною складовою інтерфейсу AgriSense є інтерактивна карта, на якій відображаються дані з різних джерел. Користувачі можуть легко переміщатися по карті, змінювати масштабування та вибирати потрібні часові проміжки для аналізу. Інтерфейс має чистий та інтуїтивний дизайн, що забезпечує зручність користування. Користувачі можуть легко орієнтуватися в системі та знаходити необхідні дані завдяки добре організованій структурі.

Можливості та функціонал

1. Моніторинг даних в реальному часі:

- AgriSense надає можливість відстежувати агрономічні дані в реальному часі, що є критично важливим для ефективного управління сільськогосподарськими процесами.

- Дані збираються з різноманітних сенсорів та IoT пристроїв, що встановлені на полях (Рис. 1.18).

Рис. 1.18.Програма AgriSense

2. Різноманітні типи візуалізацій:

- Графіки та діаграми для аналізу динаміки змін агрономічних показників (Рис. 1.19).

- Карти для географічного відображення даних, що дозволяє оцінювати стан сільськогосподарських угідь в різних регіонах (Рис. 1.20).

- Індикатори для швидкого оцінювання ключових параметрів (температура, вологість, рівень вологості ґрунту тощо) (Рис. 1.21).

Рис. 1.19. Інтерфейс програми AgriSense

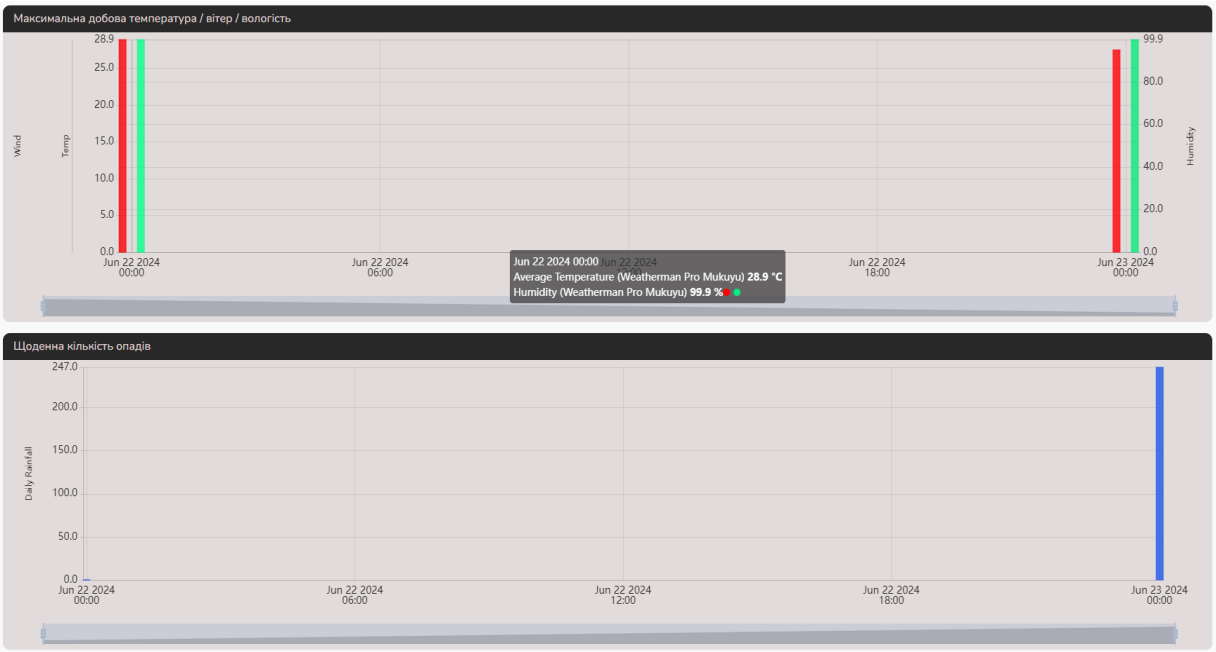

Рис. 1.20. Інтерфейс програми AgriSense

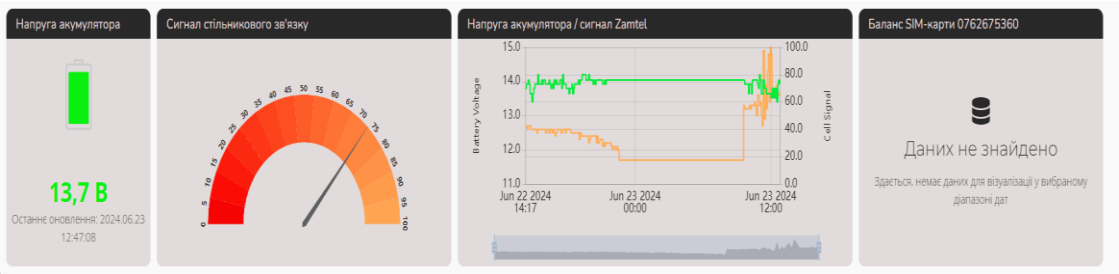

Рис. 1.21. Інтерфейс програми AgriSense

3. Фільтри та налаштування:

- панель містить численні фільтри та інструменти для налаштування відображення даних. Користувачі можуть фільтрувати інформацію за регіонами, країнами або адміністративними одиницями, а також застосовувати фільтри за типами ґрунтів, кліматичними зонами або фазами росту рослин.

4. Звітність та експорт даних:

- AgriSense пропонує можливість завантаження історичних даних та порівняння різних часових проміжків, що є важливим для виявлення змін та тенденцій у розвитку рослинності, ерозії ґрунтів, забрудненні водойм та інших процесів (Рис. 1.22).

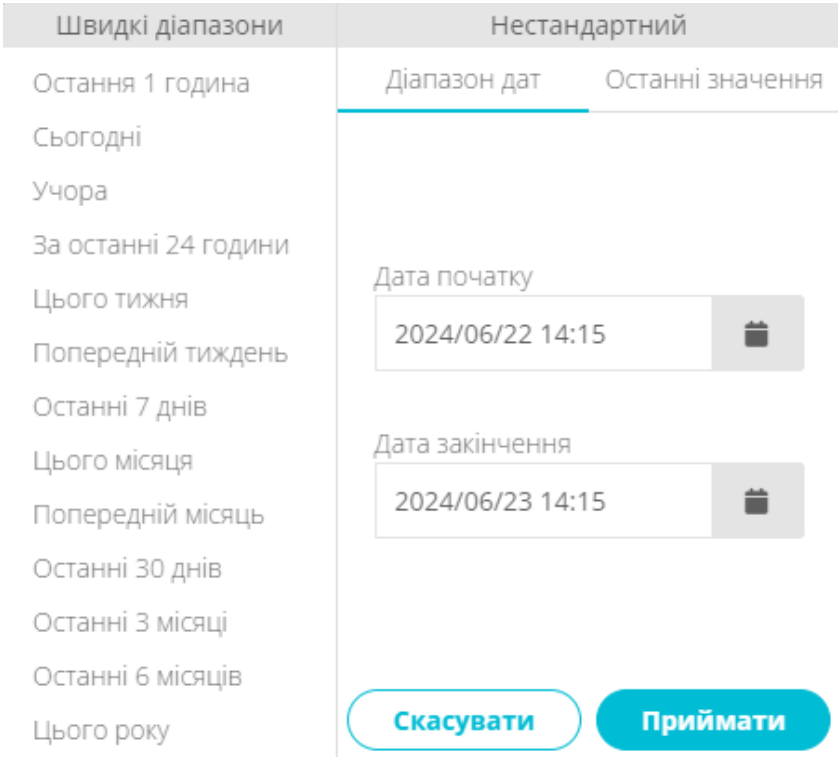

Рис. 1.22. Інтерфейс програми AgriSense

5. Інтеграція з іншими платформами:

- Платформа підтримує інтеграцію з іншими аналітичними системами та IoT платформами, що забезпечує комплексний підхід до збору та аналізу агрономічних даних.

6. Інструменти навігації та масштабування:

- Інструменти навігації та масштабування забезпечують зручну роботу з картою, дозволяючи наближати та переміщатися до цікавих регіонів. Додаткові функції, такі як вимірювання відстаней, креслення форм та позначок, допомагають проводити глибокий аналіз.

Відповідність для агромоніторингу:

Комплексний аналіз: AgriSense надає можливість проводити комплексний аналіз агрономічних даних завдяки широкому набору візуалізаційних та аналітичних інструментів.

Реальний час: Відстеження даних у реальному часі дозволяє оперативно реагувати на зміни в агрономічних умовах, що є надзвичайно важливим для ефективного управління сільським господарством.

Гнучкість налаштувань: Фільтри та налаштування відображення даних забезпечують можливість персоналізованого аналізу відповідно до конкретних потреб користувача.

Інтеграція: Можливість інтеграції з іншими системами дозволяє використовувати AgriSense як частину більшого аналітичного комплексу для агромоніторингу.

Загалом, AgriSense є потужним інструментом для моніторингу та аналізу агрономічних даних, який забезпечує широкі можливості для візуалізації та прийняття обґрунтованих рішень у сфері сільського господарства. Його функціональність та гнучкість роблять його ідеальним рішенням для використання веб-застосунку візуалізації даних агромоніторингу.

На основі наданої інформації про різні веб-інструменти для візуалізації та аналізу геопросторових та сільськогосподарських даних, можна зробити такий висновок:

Розробка програмного забезпечення веб-застосунку для візуалізації даних агромоніторингу є актуальною та перспективною темою для дипломної роботи. Існує багато потужних інструментів, таких як EO Browser, Crop Monitor та AgriSense які демонструють сучасні підходи та рішення у цій галузі.

Ці веб-застосунки пропонують користувачам зручні інтерфейси з інтерактивними картами, панелями інструментів та фільтрами для вибору та налаштування відображення даних. Вони забезпечують доступ до величезних архівів супутникових знімків, метеорологічних даних, агрономічних моделей та інших джерел інформації, які є критично важливими для агромоніторингу.

Використання хмарних технологій та потужних обчислювальних ресурсів дозволяє ефективно обробляти великі обсяги геопросторових даних, візуалізувати їх у різних формах (карти, графіки, діаграми) та виконувати складні аналітичні завдання, такі як відстеження змін, виявлення тенденцій та прогнозування.

Функції експорту та можливості інтеграції з іншими системами роблять ці веб-інструменти гнучкими та придатними для використання в різних галузях, пов'язаних з агромоніторингом, управлінням земельними ресурсами, екологічними дослідженнями та прийняттям рішень.

Загалом, найкраще підходить для використання у дипломній роботі EO Browser. Цей додаток дуже підходить для створення веб-сайту для агромоніторингу завдяки своїм численним перевагам:

1. **Інтуїтивний інтерфейс:** EO Browser має зручний та інтуїтивний інтерфейс, що робить його легким у використанні навіть для користувачів з мінімальним досвідом роботи з супутниковими даними.

2. **Різноманітність джерел даних:** Підтримка численних джерел даних, таких як Sentinel-2, Landsat, MODIS, забезпечує широкий вибір для аналізу та візуалізації.

3. **Функціональність:** Інтерактивна карта, можливість комбінування шарів даних, потужні інструменти для аналізу, завантаження історичних даних

та інші функції роблять EO Browser потужним інструментом для агромоніторингу.

4. **Безкоштовний доступ:** EO Browser надає безкоштовний доступ до своїх інструментів, що робить його доступним для використання у дипломній роботі без додаткових витрат.

Цей додаток є ідеальним рішенням для візуалізації даних агромоніторингу, що повністю відповідає вимогам дипломної роботи.

1.2. Призначення розробки та галузь застосування

Повна назва розробленої системи: "АгроВізуаль" - веб-застосунок для візуалізації та аналізу даних агромоніторингу".

Ключові терміни:

- Агромоніторинг - комплексний моніторинг стану сільськогосподарських угідь, ґрунтового покриву, рослинності та кліматичних умов з використанням новітніх технологій збору та обробки даних.

- Візуалізація даних - процес перетворення великих масивів структурованих та неструктурованих даних у наочні візуальні форми, такі як інтерактивні карти, графіки, діаграми та Dashboard, для полегшення аналізу, виявлення закономірностей та тенденцій.

Необхідність розробки веб-застосунку "АгроВізуаль" виникла через стрімке зростання обсягів даних агромоніторингу, отриманих з різноманітних джерел, таких як супутникові знімки, метеостанції, безпілотні літальні апарати та сенсорні мережі. Ефективна обробка, аналіз та інтерпретація цих даних стали критично важливими для підвищення ефективності сільськогосподарського виробництва, раціонального використання ресурсів та прийняття обґрунтованих управлінських рішень.

Галузі застосування веб-застосунку "АгроВізуаль":

1. Точне землеробство - візуалізація даних дозволяє оптимізувати використання добрив засобів захисту рослин та водних ресурсів на основі добр-

ив, засобів захисту рослин та водних ресурсів на основі аналізу стану ґрунту, рослинного покриву та кліматичних умов.

2. Прогнозування врожайності - аналіз історичних даних, поточного стану посівів та погодних умов дає можливість робити достовірні прогнози врожайності для різних культур.

3. Моніторинг стану сільськогосподарських угідь - візуалізація супутникових знімків, даних з безпілотників та датчиків дозволяє відстежувати стан полів, виявляти проблемні ділянки та своєчасно вживати заходів.

4. Управління сільськогосподарськими ресурсами - візуалізація просторових даних полегшує планування та оптимізацію використання земельних, водних та трудових ресурсів.

5. Дослідження в галузі агрономії та кліматології - візуалізація багатовимірних даних є потужним інструментом для виявлення взаємозв'язків між різними факторами та вивчення їх впливу на продуктивність сільськогосподарських культур.

Таким чином, веб-застосунок "АгроВізуаль" є унікальним рішенням, яке об'єднує передові технології збору, обробки та візуалізації даних агромоніторингу. Він надає агровиробникам, дослідникам та фахівцям у галузі сільського господарства можливість ефективно аналізувати великі масиви даних. Це сприяє прийняттю обґрунтованих рішень для підвищення продуктивності та сталого розвитку агропромислового комплексу. Завдяки інтеграції різноманітних джерел даних, таких як супутникові знімки, метеодані та дані з сенсорних мереж, веб-застосунок забезпечує комплексний підхід до агромоніторингу, що дозволяє знизити витрати, підвищити врожайність та зменшити негативний вплив на довкілля.

1.3. Підстави для розробки

В кінці навчання, студент виконує кваліфікаційну роботу (проект). Тема роботи узгоджується з керівником проекту, випускаючою кафедрою.

Підставою для розробки кваліфікаційної роботи на тему «Розробка веб-застосунку візуалізації даних агромоніторингу» є наказ № 469 від 23.05.2024 р. по Національному технічному університету «Дніпровська політехніка».

1.4. Постановка завдання

Мета та призначення: створення веб-застосунку [10] для візуалізації та аналізу даних агромоніторингу, що дозволить агровиробникам та фахівцям у галузі сільського господарства ефективно обробляти, аналізувати та інтерпретувати великі обсяги даних, отриманих з різноманітних джерел. Розробка такого веб-застосунку є доцільною, оскільки візуалізація даних полегшує виявлення закономірностей, тенденцій та прийняття обґрунтованих рішень для підвищення ефективності сільськогосподарського виробництва.

Призначення вихідної інформації: Візуалізація даних у формі інтерактивних карт, графіків, діаграм та Dashboard для аналізу стану посівів, прогнозування врожайності, планування агротехнічних заходів, моніторингу ефективності використання ресурсів.

Об'єкти, у ході управління якими використовують розроблену систему(Рис. 1.23):

Рис. 1.23. Система моніторингу сільськогосподарських угідь

Вимоги до організації збору та передачі вхідної інформації: Забезпечення інтеграції з різноманітними джерелами даних (супутникові знімки, метеодані, дані з безпілотників, сенсорних мереж та ін.) за допомогою стандартизованих інтерфейсів та протоколів передачі даних. Організація контролю якості та коригування вхідних даних.

Умови, при яких припиняється розв'язання завдання автоматизованим способом: Відсутність або некоректність вхідних даних, технічні збої в роботі системи, відсутність доступу до мережі Інтернет.

Зв'язок з іншими задачами: Веб-застосунок може інтегруватися з системами управління сільськогосподарським виробництвом, системами управління ресурсами, геоінформаційними системами тощо.

Розподіл функцій між персоналом та технічними засобами: Персонал (агрономи, фахівці з точного землеробства, дослідники) здійснює аналіз та інтерпретацію візуалізованих даних, приймає рішення щодо агротехнічних заходів. Технічні засоби (веб-застосунок, сервери, мережеве обладнання) забезпечують збір, обробку, зберігання та візуалізацію даних агромоніторингу.

1.5. Вимоги до веб-застосунку "АгроВізуаль" для візуалізації даних агромоніторингу

1.5.1. Вимоги до функціональних характеристик

Веб-застосунок "АгроВізуаль" повинен забезпечувати наступні функціональні можливості:

1. Багатоканальний імпорт даних агромоніторингу [11]:

- Інтеграція з різноманітними джерелами даних: метеостанціями, супутниковими знімками, безпілотними літальними апаратами, сенсорними мережами моніторингу ґрунтів тощо.

- Підтримка імпорту даних у форматах XML, JSON, CSV, GeoTIFF та інших стандартизованих форматах.

- Автоматизована обробка та перетворення вхідних даних у формат, придатний для візуалізації.

2. Гнучка візуалізація та аналіз даних:

- Інтерактивні геоінформаційні карти з шарами даних про стан посівів, погодні умови, вологість ґрунту тощо.

- Різноманітні типи графіків та діаграм для відображення часових рядів, порівняння показників, виявлення трендів.

- Налаштовувані Dashboard з комбінацією різних візуалізацій для комплексного аналізу.

- Інструменти фільтрації, сортування, масштабування та порівняння даних за різними критеріями.

3. Прогнозування та підтримка прийняття рішень:

- Прогнозування врожайності та продуктивності культур з використанням машинного навчання та статистичних методів.

- Моделювання впливу різних факторів (погодні умови, агротехнічні заходи тощо) на врожайність [15].

- Рекомендації щодо оптимальних агротехнічних заходів на основі аналізу даних [17].

4. Експорт даних та інтеграція з іншими системами:

- Можливість обміну даними та результатами аналізу з системами управління сільськогосподарським виробництвом, геоінформаційними [18] системами тощо.

5. Зручний інтерфейс користувача:

- Адаптивний дизайн для забезпечення зручного використання на різних пристроях (ПК, планшети, смартфони).

- Інтуїтивно зрозумілий інтерфейс з можливістю налаштування робочого простору та збереження індивідуальних налаштувань.

- Підтримка багатокористувацького режиму роботи з диференціацією прав доступу.

6. Безперервне оновлення та моніторинг даних [12]:

- Автоматичне оновлення даних відповідно до надходження нових даних з джерел агромоніторингу.

- Моніторинг критичних ситуацій (несприятливі погодні умови, пошкодження посівів тощо) та своєчасне повідомлення користувачів [16].

- Журналювання змін та подій для подальшого аналізу.

1.5.2. Вимоги до інформаційної безпеки

Веб-застосунок "АгроВізуаль" повинен забезпечувати належний рівень інформаційної безпеки, включаючи:

- Шифрування конфіденційних даних (паролі користувачів, комерційно чутлива інформація тощо) під час передачі та зберігання.

- Контроль доступу з диференціацією прав користувачів відповідно до їх ролей та повноважень.

- Регулярне резервне копіювання даних для забезпечення цілісності та відновлення у разі збоїв або кібератак.

1.5.3. Вимоги до складу та параметрів технічних засобів

Для функціонування веб-застосунку "АгроВізуаль" необхідні наступні технічні засоби:

1. Серверне обладнання (Рис. 1.24):

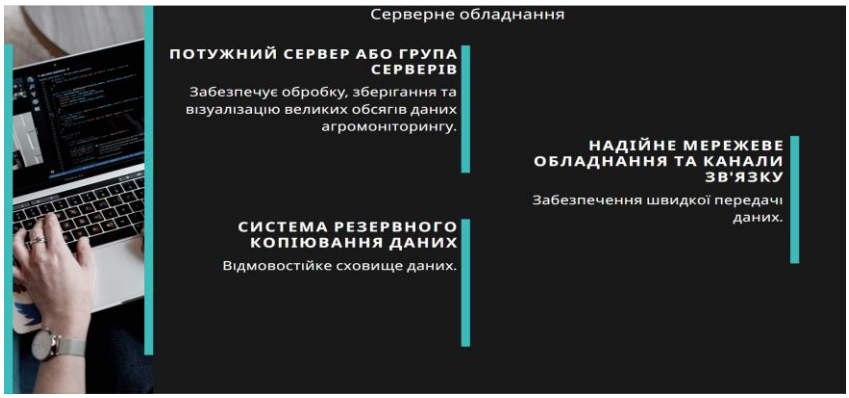

Рис. 1.24. Серверне обладнання

2. Клієнтські пристрої (Рис. 1.25):

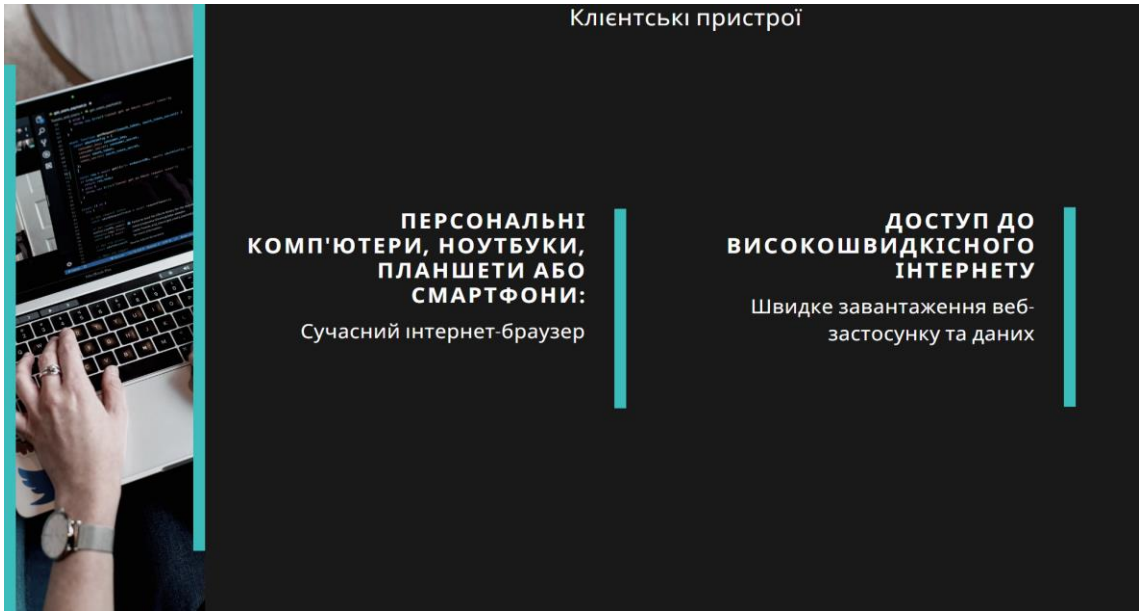

Рис. 1.25. Клієнтські пристрої

3. Допоміжне обладнання (Рис. 1.26):

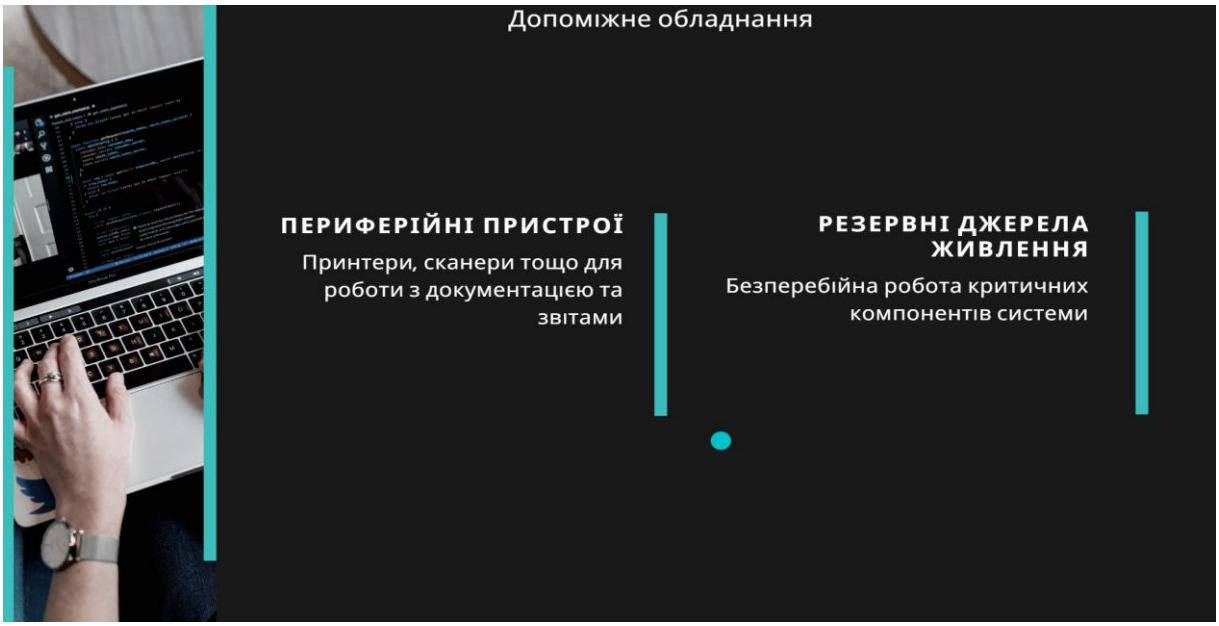

Рис. 1.26. Допоміжне обладнання

Точні технічні вимоги до складу та параметрів обладнання будуть визначені на подальших етапах розробки веб-застосунку "АгроВізуаль" з урахуванням обсягів даних, кількості користувачів та інших факторів.

1.5.4. Вимоги до інформаційної та програмної сумісності

Веб-застосунок "АгроВізуаль" повинен забезпечувати сумісність з різними інформаційними системами та програмними продуктами, зокрема:

- Сумісність з популярними інтернет-браузерами (Google Chrome, Mozilla Firefox, Microsoft Edge тощо) на різних операційних системах (Windows, Android).

- Можливість інтеграції з геоінформаційними системами (ГІС) та системами управління базами даних [19] (СУБД) за допомогою стандартизованих інтерфейсів та протоколів обміну даними.

- Підтримка стандартів обміну даними в галузі сільського господарства та агромоніторингу.

- Забезпечення сумісності з різноманітними апаратними платформами для збору даних агромоніторингу [21].

Висновок

У першому розділі був проведений детальний аналіз предметної галузі агромоніторингу. Це включало розгляд сучасних технологій та методів, які застосовуються у сфері точного землеробства. Було визначено ключові проблеми, такі як недостатня точність прогнозів, обмежена інтеграція даних з різних джерел та необхідність покращення візуалізації даних для підтримки прийняття рішень.

Також були сформульовані вимоги до функціональних характеристик веб-застосунку, включаючи необхідність підтримки різних форматів даних, забезпечення високої продуктивності та безпеки. Було проведено аналіз існуючих рішень на ринку та виявлено їхні слабкі сторони, що дало змогу чітко визначити напрямки для покращення та розробки унікального програмного продукту "АгроВізуаль"

РОЗДІЛ 2
ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

2.1. Функціональне призначення програми

Розроблений веб-застосунок **"АгроВізуаль"** призначений для візуалізації [20] та аналізу даних агромоніторингу, що надає можливість аграрним підприємствам та фахівцям у галузі сільського господарства приймати обґрунтовані рішення на основі отриманих даних. Основними функціями програми є:

1.Відображення інтерактивної карти з аграрними ділянками:

- Інтерактивна карта дозволяє користувачам бачити розташування кожної аграрної ділянки, її межі та основні характеристики. Карта забезпечує зручну навігацію та масштабування для детального аналізу обраних ділянок.

2.Виведення інформації про стан посівів та ґрунту:

- Програма відображає різноманітну інформацію про стан посівів та ґрунту, включаючи температуру, опади, вологість та інші індекси. Це дозволяє агрономам ефективно моніторити стан сільськогосподарських угідь та приймати обґрунтовані рішення щодо їх обробки.

3.**Можливість перегляду та додавання коментарів до кожної ділянки:**

- Користувачі можуть залишати коментарі до кожної ділянки, що сприяє кращому розумінню та обговоренню стану посівів та інших важливих аспектів. Коментарі можуть містити текстову інформацію, зображення та посилання на додаткові ресурси.

4.**Пошук ділянок за координатами або назвами:**

- Веб-застосунок забезпечує функцію пошуку ділянок за координатами або назвами, що спрощує навігацію та робить процес моніторингу більш зручним. Користувачі можуть швидко знайти необхідну ділянку за допомогою інтерактивного пошукового інтерфейсу.

2.2. Опис застосованих математичних методів

У розробці системи не використовувалися складні математичні методи. Основними математичними операціями є обробка даних з JSON-файлів та їх відображення на карті. Дані представлені у вигляді масивів та об'єктів, які зберігають інформацію про координати ділянок, їх розміри, температуру, індекси та коментарі.

1. Обробка даних:

- Веб-застосунок використовує прості алгоритми для обробки даних, зокрема фільтрацію, сортування та агрегування. Дані з JSON-файлів завантажуються у вигляді масивів об'єктів, що дозволяє легко маніпулювати ними та отримувати необхідну інформацію.

2. Візуалізація даних:

- Веб-застосунок використовує бібліотеку Leaflet для відображення даних на карті. Це включає побудову маркерів для кожної ділянки та відображення інформаційних вікон з деталями про стан посівів та ґрунту.

2.3. Опис використаної архітектури та шаблонів проектування

Застосунок побудовано за архітектурою "клієнт-сервер". Це забезпечує розподіл функцій між клієнтською та серверною частинами, що покращує масштабованість та продуктивність системи.

1. Серверна частина:

- Серверна частина реалізована за допомогою Node.js, що забезпечує обробку запитів та взаємодію з файлами даних. Сервер виконує роль посередника між клієнтом та базою даних, забезпечуючи швидке та надійне обслуговування запитів користувачів.

2. Клієнтська частина:

- Клієнтська частина розроблена з використанням фреймворку Vue.js, що дозволяє створювати реактивні та інтерактивні компоненти. Це забезпечує вис-

оку швидкість роботи інтерфейсу та зручність для користувачів.

3. Архітектурний шаблон:

- Основний шаблон проектування – Model-View-Controller (MVC). У цій архітектурі модель представлена даними, контролер обробляє логіку, а вид відповідає за відображення інформації. Це дозволяє розділити різні аспекти програми та полегшує її підтримку та розширення.

2.4. Опис використаних технологій та мов програмування

Для розробки застосунку використано наступні технології та мови програмування:

- Node.js:

- Серверна платформа для обробки запитів та взаємодії з файлами даних. Node.js забезпечує високу продуктивність та масштабованість серверної частини.

- Express:

- Фреймворк для створення веб-серверу на базі Node.js. Express спрощує розробку серверної частини, надаючи зручні інструменти для обробки HTTP-запитів та маршрутизації.

- Vue.js:

- Фреймворк для розробки інтерфейсу користувача. Vue.js дозволяє створювати реактивні компоненти, що забезпечують високу швидкість роботи інтерфейсу та зручність для користувачів.

- Leaflet:

- Бібліотека для роботи з інтерактивними картами. Leaflet використовується для візуалізації аграрних ділянок та відображення інформації про них на карті.

- JavaScript:

- Основна мова програмування для фронтенду та бекенду. JavaScript використовується для створення динамічних та інтерактивних веб-сторінок, заб-

езпечуючи взаємодію між користувачем та сервером.

- JSON:

- Формат зберігання даних. JSON використовується для передачі даних між сервером та клієнтом, забезпечуючи зручність та ефективність, забезпечуючи зручність та ефективність обробки інформації.

2.5. Опис структури програми та алгоритмів її функціонування

Схематично, процес взаємодії між інтерфейсом та сервером можна представити таким чином (Рис. 2.1):

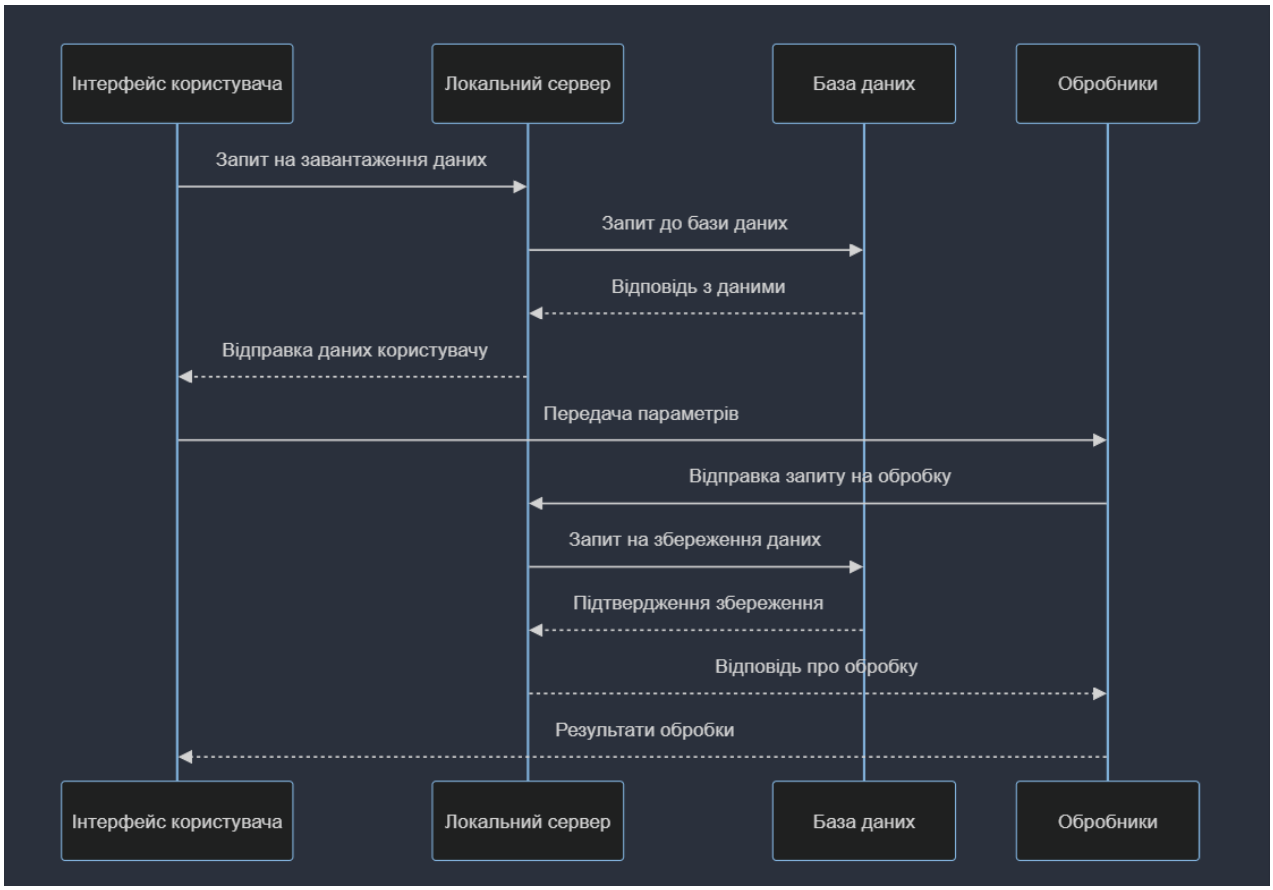

Рис. 2.1. Схема взаємодії інтерфейсу з сервером

Структура програми складається з наступних компонентів:

- App.vue:

- Головний компонент, що містить заголовок та основні компоненти прог-

рами. Він відповідає за ініціалізацію програми та завантаження основних компонентів.

- MapView.vue:

- Компонент, що відповідає за відображення карти та інформації про ділянки. Він завантажує дані з сервера, відображає аграрні ділянки на карті та надає користувачам можливість взаємодії з ними.

- Sidebar.vue:

- Компонент бічної панелі, що містить меню навігації. Він забезпечує користувачам доступ до різних розділів програми, включаючи інформацію про ділянки та коментарі.

Алгоритм функціонування:

1.Завантаження головного компонента App.vue:

- При завантаженні сторінки головний компонент ініціалізує програму, відображає заголовок та основні елементи інтерфейсу, включаючи карту та бічну панель.

Взаємодія здійснюється за допомогою функцій, які відправляють запити на сервер і отримують відповіді. Ці універсальні функції, що містять кілька аргументів, можна використовувати для будь-якої таблиці та для пошуку/отримання будь-яких рядків за різними пошуковими параметрами. Нижче наведено приклад такої функції:

```
async fetchData() {
  try {
    const response = await fetch("http://localhost:3000/data");
    const data = await response.json();
    this.markers = data;
  } catch (error) {
    console.error("Error fetching data:", error);
  }
}
```

2.Компонент MapView.vue завантажує дані з сервера:

- Компонент отримує дані про аграрні ділянки з сервера у форматі JSON та відображає їх на карті. Користувачі можуть взаємодіяти з картою, переглядати інформацію про ділянки та додавати коментарі.

3.Взаємодія з ділянками на карті:

- При натисканні на ділянку відображається детальна інформація про стан посівів та ґрунту, а також коментарі. Користувачі можуть переглядати та додавати нові коментарі до обраних ділянок.

4.Додавання нових коментарів:

- Користувачі можуть додавати нові коментарі через відповідну форму, яка відображається при натисканні на ділянку. Коментарі зберігаються у базі даних та стають доступними для інших користувачів. Схема маршруту користувача:

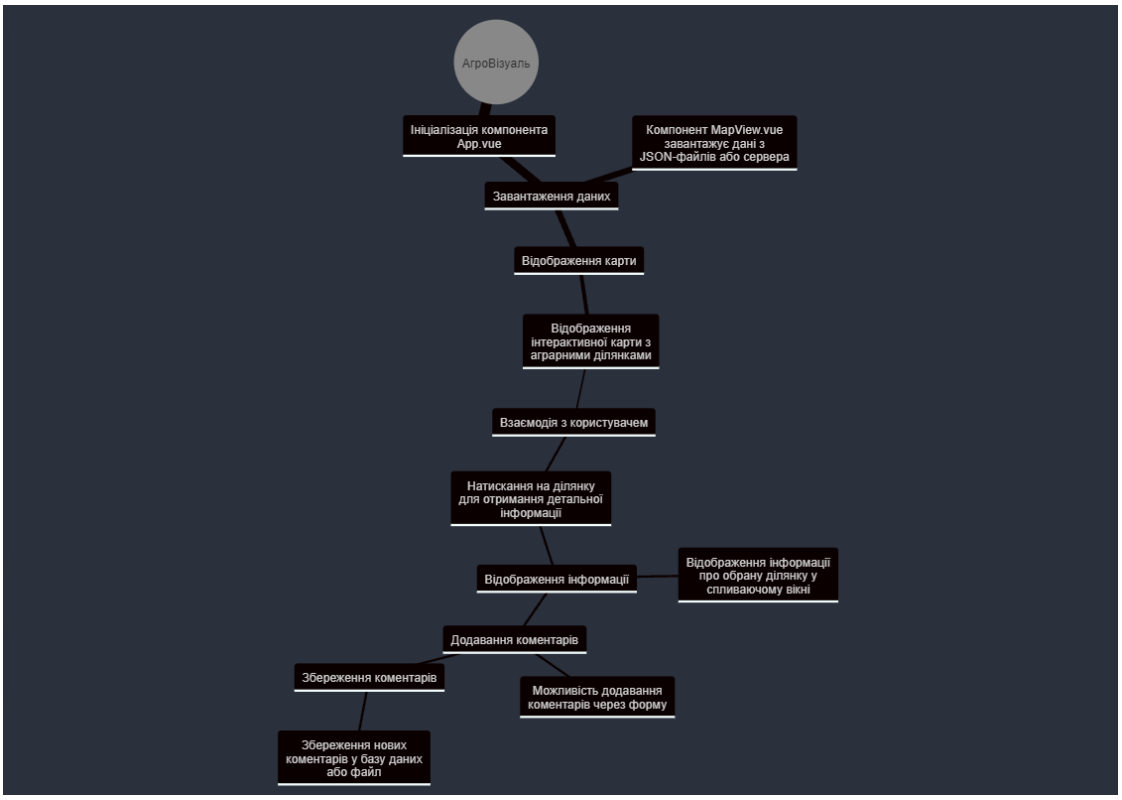

Рис. 2.2. Блок-схема алгоритму

2.6. Обґрунтування та організація

Вхідних та вихідних даних програми:

Вхідні дані:

- Дані супутникового моніторингу у форматі JSON, що містять координати ділянок, їх розміри, температурні показники, індекси та коментарі.

- Метеодані, що включають інформацію про температуру повітря, опади, вологість та швидкість вітру.

- Дані з безпілотних літальних апаратів та сенсорних мереж, що включають мультиспектральні знімки посівів та карти рельєфу.

Вихідні дані:

- Візуалізація аграрних ділянок на інтерактивній карті.

- Таблиці та спливаючі вікна, що містять детальну інформацію про стан посівів та ґрунту, а також коментарі.

Організація вхідних та вихідних даних:

- Вхідні дані завантажуються у форматі JSON, що забезпечує їх зручне зберігання та передачу між сервером та клієнтом. Кожен JSON-файл містить інформацію про аграрні ділянки, включаючи їх координати, розміри, температурні показники, індекси та коментарі.

- Вихідні дані відображаються на карті у вигляді маркерів та інформаційних вікон. Користувачі можуть взаємодіяти з картою, переглядати детальну інформацію про кожну ділянку та додавати коментарі.

2.7. Опис розробленого програмного продукту

2.7.1. Використані технічні засоби

Розробка та тестування програмного продукту здійснювались на персональному комп'ютері з операційною системою Windows 10, процесором Intel Core i5, 8 ГБ оперативної пам'яті та 256 ГБ SSD. Для забезпечення надійності та продуктивності використовувались додаткові технічні засоби:

-Веб-сервери:

- Використовувались для обробки та зберігання даних, забезпечуючи швидкий доступ до інформації та її безпеку.

-Персональні комп'ютери та ноутбуки:

- Використовувались для розробки, тестування та демонстрації програмного продукту.

-Периферійні пристрої:

- Принтери та сканери використовувались для роботи з документацією та звітами.

2.7.2. Використані програмні засоби

Для розробки та функціонування веб-застосунку використовувались наступні програмні засоби:

- Операційна система:

- Windows 10, що забезпечувала стабільне середовище для розробки та тестування.

-Редактор коду:

- Notepad++ [22], який надавав зручні інструменти для написання та налагодження коду.

- Серверна платформа:

- Node.js [14], яка забезпечувала високу продуктивність та масштабованість серверної частини застосунку.

- Фреймворки:

- Express [23], що спрощував розробку веб-сервера.

- Vue.js [13], що дозволяв створювати реактивні компоненти для інтерфейсу користувача.

- Бібліотеки:

- Leaflet, яка використовувалась для інтерактивного відображення карт та аграрних ділянок.

2.7.3. Виклик та завантаження програми

Програма запускається через командний рядок за допомогою команди `np-

m start`, яка запускає сервер на базі Node.js. Клієнтська частина завантажується при відкритті веб-браузера за адресою `http://localhost:3000`.

1. Запуск серверної частини:

- Для запуску серверної частини використовується команда `npm start`, яка ініціює роботу веб-сервера на базі Node.js та Express.

2. Завантаження клієнтської частини:

- Клієнтська частина застосунку завантажується при відкритті веб-браузера за вказаною URL-адресою. Користувачам надається доступ до інтерактивної карти та інших функцій застосунку.

2.7.4. Опис інтерфейсу користувача

Інтерфейс користувача веб-застосунку "Агромоніторинг" складається з наступних основних елементів:

1.Головна сторінка з інтерактивною картою:

- Інтерактивна карта є основним елементом інтерфейсу, на якій відображаються аграрні ділянки. Користувачі можуть масштабувати карту, переглядати детальну інформацію про кожну ділянку та додавати коментарі (Рис. 2.3).

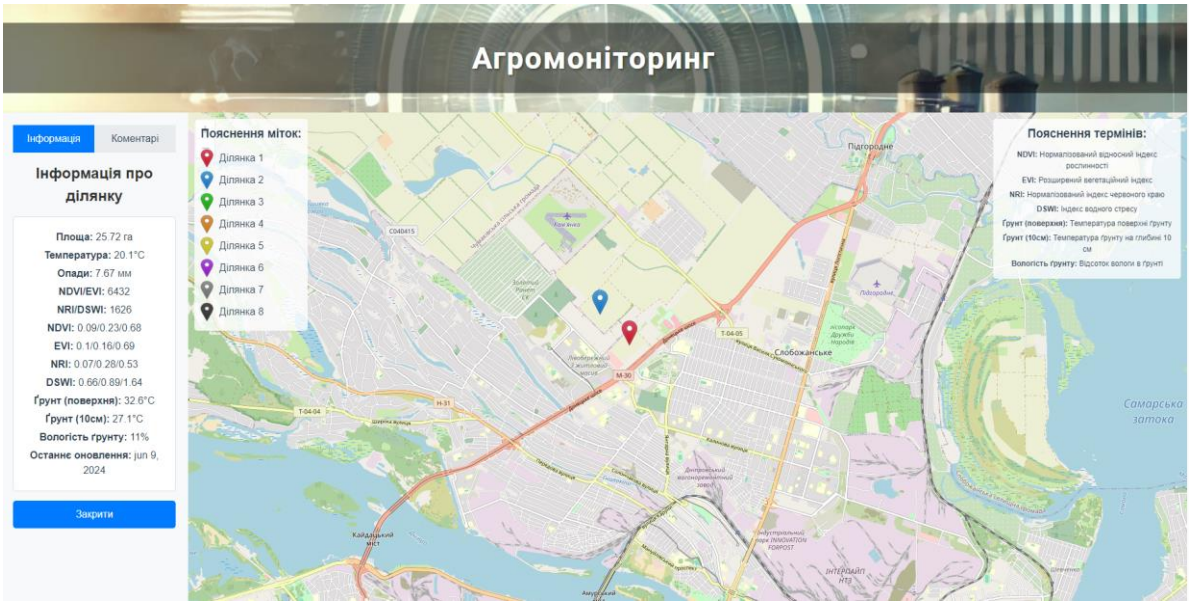

Рис. 2.3. Головна сторінка сайту

2.Бічна панель навігації:

- Бічна панель містить меню навігації, що забезпечує користувачам доступ до різних розділів програми, включаючи інформацію про ділянки, коментарі та налаштування (Рис. 2.4).

Рис. 2.4. Бічна панель навігації

3.Форми для додавання коментарів:

- Користувачі можуть додавати нові коментарі через спеціальні форми, що

відображаються при натисканні на ділянку. Це дозволяє легко додавати текстову інформацію, зображення та інші дані (Рис. 2.5).

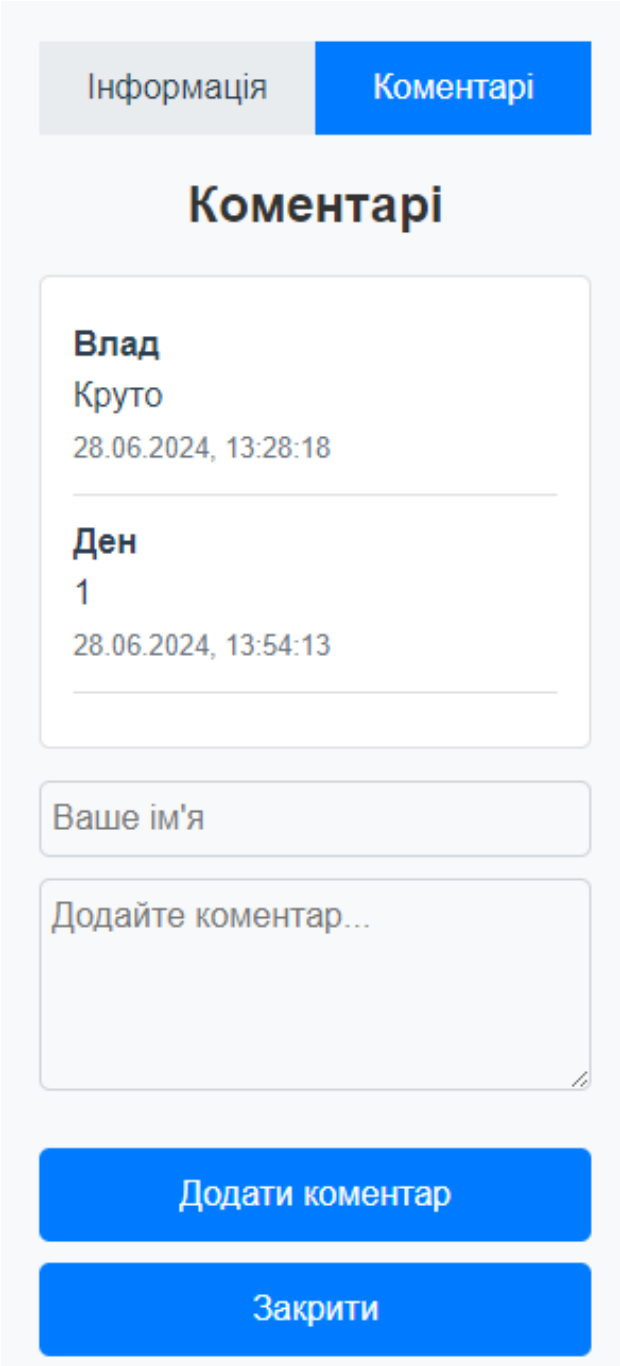

Рис. 2.5. Додавання коментарів

4.Таблиці та спливаючі вікна:

- Інформація про ділянки відображається у вигляді таблиць (Рис. 2.6) та спливаючих вікон, що полегшує аналіз даних та надає користувачам зручний доступ до необхідної інформації (Рис. 2.7).

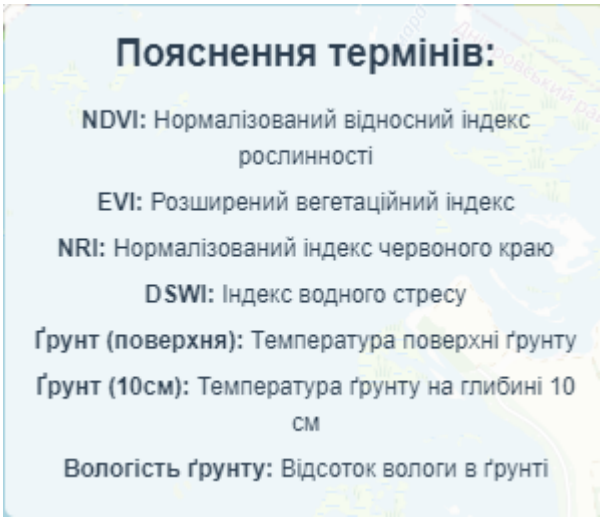

Рис. 2.6. Пояснення термінів

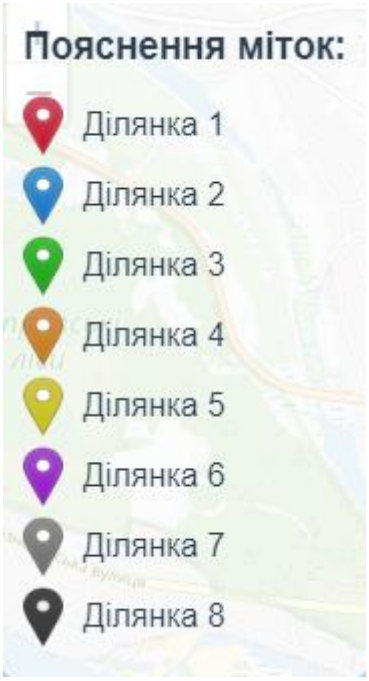

Рис. 2.7. Пояснення міток

Розроблений веб-застосунок "АгроВізуаль" є потужним інструментом для аграріїв, дослідників та фахівців у галузі сільського господарства. Він забезпечує ефективну візуалізацію та аналіз даних, підтримку прийняття рішень та підвищення продуктивності аграрного сектору.

Висновок

У другому розділі детально розглянуто процес проектування та розробки веб-застосунку "АгроВізуаль". Було визначено функціональне призначення програми – відображення інтерактивної карти з аграрними ділянками, виведення інформації про стан посівів та ґрунту, а також можливість перегляду та додавання коментарів.

Обґрунтовано вибір архітектури клієнт-сервер, яка забезпечує високу продуктивність та масштабованість системи. Серверна частина реалізована з використанням Node.js та фреймворку Express, що забезпечує ефективну обробку запитів та взаємодію з файлами даних. Клієнтська частина розроблена на базі Vue.js, що дозволяє створювати реактивні та інтерактивні компоненти для відображення даних.

Детально описано структуру програми, яка включає основні компоненти: головний компонент App.vue, компонент MapView.vue для відображення карти та інформації про ділянки, та компонент Sidebar.vue для бічної панелі навігації. Алгоритм функціонування включає завантаження даних з сервера, їх обробку та візуалізацію на інтерактивній карті.

Вхідні дані програми представлені у форматі JSON і містять інформацію про координати ділянок, їх розміри, температурні показники, індекси та коментарі. Вихідні дані – це відображення на карті та виведення інформації у вигляді таблиць та спливаючих вікон, що забезпечує зручну взаємодію користувачів з системою.

Розроблений веб-застосунок "АгроВізуаль" є важливим інструментом для аграріїв, що забезпечує ефективну візуалізацію та аналіз агроданих. Використання сучасних технологій та інструментів, таких як Node.js, Vue.js та Leaflet, дозволяє створювати зручний та надійний інтерфейс, що підвищує ефективність управління сільськогосподарськими ресурсами.

РОЗДІЛ 3
ЕКОНОМІЧНИЙ РОЗДІЛ

3.1. Розрахунок трудомісткості та вартості розробки програмного продукту

Початкові дані:

1. q – передбачуване число операторів програми – 850;
2. C – коефіцієнт складності програми – 1,25;
3. p – коефіцієнт корекції програми в ході її розробки – 0,2;
4. $C_{пр}$ – годинна заробітна плата програміста – 200 грн/год;
5. B – коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі – 1,3;
6. k – коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності – 1,25;
7. $C_{мв}$ – вартість машино-години ЕОМ – 5,8 грн/год;
8. B_k – число виконавців – 1 людина;
9. F_p – місячний фонд робочого часу (при 40 годинному робочому тижні) – 176 годин.

Примітка:

У 2023 році заробітна плата програміста кваліфікації Junior Software Engineer, який працює з мовою програмування JavaScript, становить 600$ [24] на місяць. Враховуючи те, що програміст працює в середньому 22 дні на місяць при чинному курсі Національного банку України 1 долар = 38 гривень, заробітна плата програміста за годину виходить 200 грн/год.

Вихідна вартість машино-години включає в себе витрати на електроенергію, доступ до мережі інтернет, вартість амортизації приладу.

Зарядний блок живлення сучасного ЕОМ, в даному випадку – комп’ютер та монітор, розрахований на потужність до 135 Вт.

Тариф на електроенергію: 4,32 грн/кВт·год [25].

Ціна за годину:

$$4{,}32 * 0{,}135 \text{ кВт} = 0{,}58 \text{ грн/год}$$

Тариф інтернет зв'язку: 190 грн/місяць

Ціна за годину:

$$190 / (176 \text{ год.}) \approx 1{,}08 \text{ грн/год.}$$

Погодинна амортизація компютера за умови початкової вартості 40 000 грн., ліквідаційної вартості – 10 000 грн, терміну корисного використання – 4 роки прямолінійним методом розраховується відповідно:

$$(40000 \text{ грн} – 5000 \text{ грн}) / (12 \text{ міс} \cdot 4 \text{ роки} \cdot 176 \text{ годин}) \approx 4{,}14 \text{ грн}$$

Сумарна вартість машино-години ЕОМ складає:

$$0{,}58 + 1{,}08 + 4{,}14 = 5{,}8 \text{ грн/год.}$$

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки ПЗ може бути розрахована на основі системи моделей з різною точністю оцінки.

Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t = t_o + t_u + t_a + t_n + t_{отл} + t_д, \text{ люд-годин,} \qquad (3.1)$$

- де t_o-витрати праці на підготовку й опис поставленої задачі (приймається 45 людино-годин);

t_u - витрати праці на дослідження алгоритму рішення задачі;

t_a- витрати праці на розробку блок-схеми алгоритму;

$t_п$-витрати праці на програмування по готовій блок-схемі;

$t_{отл}$-витрати праці на налагодження програми на ЕОМ;

$t_д$ - витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у програмному забезпеченні, яке розробляється.

Умовне число операторів (підпрограм):

$$Q=q\cdot C(1+p) \qquad (3.2)$$

де:

- q - передбачуване число операторів (850);

- C - коефіцієнт складності програми (1,15);

- p - коефіцієнт корекції програми в ході її розробки (0,2).

Звідси умовне число операторів в програмі:

$$Q = 850\cdot 1{,}25\cdot(1 + 0{,}2) = 1275$$

Витрати праці на вивчення опису задачі t_u визначаються з урахуванням уточнення опису і кваліфікації програміста:

$$t_u=\frac{Q\cdot B}{(75...85)K}, \text{ людино-годин} \qquad (3.3)$$

де:

- B - коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі (1,3);

- k - коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності(1,25).

Приймемо збільшення витрат праці внаслідок недостатнього опису завдання не більше 50% (B = 1,3). З урахуванням коефіцієнта кваліфікації k = 1,25 отримуємо витрати праці на вивчення опису завдання:

$$t_u = (1275 \cdot 1,3) / (75 \cdot 1,25) = 17,68 \text{ людино-годин}$$

Витрати праці на розробку алгоритму рішення задачі визначаються за формулою:

$$t_a = \frac{Q}{(20...25)K} \quad (3.4)$$

де Q – умовне число операторів програми;

K – коефіцієнт кваліфікації програміста.

Підставивши відповідні значення в формулу, отримаємо:

$$t_a = 1275 / (20 \cdot 1,25) = 51 \text{ людино-годин}$$

Витрати праці на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20...25)K} \quad (3.5)$$

Підставивши відповідні значення в формулу, отримаємо:

$$t_n = 1275 / (20 \cdot 1,25) = 51 \text{ людино-годин.}$$

Витрати праці на налагодження програми на ЕОМ:

- За умови автономного налагодження одного завдання:

$$t_{отл} = \frac{Q}{(4...5)K} \quad (3.6)$$

$$t_{отл} = 1275 / (4 \cdot 1,25) = 255 \text{ людино-годин}$$

- За умови комплексного налагодження завдання:

$$t^{к}_{отл}=1,5\cdot t_{отл} \quad (3.7)$$

$$t^{к}_{отл}= 1,5 \cdot 255 = 382,5 \text{ людино-годин}$$

Витрати праці на підготовку документації визначаються за формулою:

$$t_{д}=t_{др}+t_{до} \quad (3.8)$$

- де $t_{др}$-трудомісткість підготовки матеріалів і рукопису.
- $t_{до}$ - трудомісткість редагування, печатки й оформлення документації:

$$t_{\text{др}} = \frac{Q}{(15\ldots20)K} \quad (3.9)$$

$$t_{др}= 1275/ (15 \cdot 1,25) = 68 \text{ людино-годин.}$$

- $t_{до}$ - трудомісткість редагування, печатки й оформлення документації:

$$t_{до}=0,75\cdot t_{др} \quad (3.10)$$

$$t_{до} = 0,75 \cdot 68 = 51 \text{ людино-годин.}$$

Розрахуємо витрати праці на підготовку документації за формулою (3.8):

$$t_{д}= 68+ 51= 119 \text{ людино-годин.}$$

Повертаючись до формули (3.1), отримаємо повну оцінку трудомісткості

розробки програмного забезпечення:

t = 45 + 17,68 + 51 + 51 + 255 + 119 = 538,68 людино-години

У результаті отримано, що в загальній складності необхідно 538,68 людино-годин для розробки даного програмного забезпечення.

3.2. Розрахунок витрат на створення програми

Витрати на створення ПЗ $К_{ПО}$ включають витрати на заробітну плату виконавця програми $З_{ЗП}$і витрат машинного часу, необхідного на налагодження програми на ЕОМ:

$$К_{ПО}=З_{ЗП}+З_{МВ}, \text{ грн}, \quad (3.11)$$

де $З_{ЗП}$– заробітна плата виконавців, яка визначається за формулою:

$$З_{ЗП}=t\cdot С_{ПР}, \text{ грн}, \quad (3.12)$$

де: *t* - загальна трудомісткість, людино-годин

$С_{ПР}$ - середня годинна заробітна плата програміста, грн/година

З урахуванням того, що середня годинна зарплата програміста становить 200 грн / год, отримуємо:

$$З_{ЗП}= 538,68 \cdot 200 = 107736 \text{ грн.}$$

Вартість машинного часу, необхідного для налагодження програми на ЕОМ, визначається за формулою:

$$З_{МВ} = t_{отл} \cdot СМ, \text{ грн,} \quad (3.13)$$

де: $t_{отл}$ - трудомісткість налагодження програми на ЕОМ, год

- $С_{мч}$ - вартість машино-години ЕОМ, грн/год (5,8 грн/год).

Підставивши в формулу (3.13) відповідні значення, визначимо вартість необхідного для налагодження машинного часу:

$$З_{МВ} = 255 \cdot 5{,}8 = 1479 \text{ грн.}$$

Звідси витрати на створення програмного продукту:

$$К_{ПО} = 107736 + 1479 = 109\,215 \text{ грн.}$$

Очікуваний період створення ПЗ:

$$Т = \frac{t}{В_к \cdot F_р}, \text{ мес.} \quad (3.14)$$

де:

- $В_к$- число виконавців (дорівнює 1);

- $F_р$ - місячний фонд робочого часу (при 40-годинному робочому тижні (Fр = 176 годин).

Звідси очікуваний період створення ПЗ:

$$Т = 538{,}68 / 1 \cdot 176 \approx 3{,}06 \text{ міс.}$$

Висновки: трудомісткість розробленого веб-застосунку для інтернет-магазину з продажу гелієвих кульок становить 538,68 людино-години. Обчислена вартість роботи по створенню програми дорівнює 109,215гривень. Визначений затрачений час на створення програмного забезпечення становить 3,06 місяці.

ВИСНОВКИ

В цій роботі досягнуто основну мету, яка полягала в розробці програмного забезпечення для візуалізації даних агромоніторингу, яке сприятиме підвищенню ефективності сільськогосподарського виробництва через точне землеробство.

Програма являє собою систему, що реалізує автоматизований збір, обробку і візуалізацію даних, отриманих з різних джерел агромоніторингу, включаючи супутники, дрони, метеостанції та сенсори. Веб-застосунок забезпечує зручний інтерфейс для користувачів, що дозволяє швидко та ефективно аналізувати агродані та приймати обґрунтовані рішення.

База даних, яка забезпечує зберігання інформації, є іменованою сукупністю даних, організованих за певними правилами, що включає загальні принципи опису, зберігання і маніпулювання даними. Програма призначена для забезпечення діяльності сільськогосподарських підприємств і надає можливість виконувати наступні дії:

- додавання, видалення і редагування інформації про сільськогосподарські угіддя, метеостанції, дрони та сенсори;

- перегляд інформації про різні джерела даних агромоніторингу;

- перегляд і друк інформації про агротехнічні заходи, метеорологічні дані, індекси рослинності та інші важливі параметри;

- здійснення пошуку необхідної інформації про стан полів, рівень вологості ґрунту, температуру та інші параметри;

- здійснення аналізу даних для прийняття рішень щодо внесення добрив, зрошення та інших агротехнічних заходів;

- можливість входу в систему з різними рівнями доступу до даних: адміністратор, агроном, технічний персонал;

- можливість зміни користувача у ході роботи програми;

- здійснення контролю введених даних: перевірка на відповідність типів, на введення обов'язкових полів даних, а також, на введення тільки можливих значень, прочитуваних із необхідних таблиць;

- можливість перегляду інформації з таблиць в режимі реального часу.

Актуальність поставленої задачі обумовлюється зростаючим попитом на такі програмні продукти, що надають можливість ефективного управління сільськогосподарськими ресурсами через точне землеробство, електронне зберігання даних про агротехнічні заходи, аналіз агроданих, отримання звітів і формування супутньої ділової документації.

Розроблене програмне забезпечення інформаційної системи призначене для застосування в будь-яких сільськогосподарських підприємствах з подібним родом діяльності і схожими функціональними вимогами.

Створений веб-застосунок дозволить оптимізувати та спростити дії по веденню агроданих; скоротити час на прийняття рішень; підвищити ефективність діяльності сільськогосподарських підприємств шляхом електронного ведення документації з агромоніторингу і можливістю аналізу наявних даних та надання необхідних звітів і супутньої ділової документації.

Структура програми являє собою клієнтський додаток, написаний мовою програмування JavaScript з використанням фреймворку Vue.js та Quasar, що взаємодіє з серверною частиною на Python з використанням Flask. База даних розроблена на платформі PostgreSQL.

В «Економічному розділі» визначено трудомісткість розробки програмного забезпечення (610,29 людино-години), підраховані витрати на створення програмного забезпечення (134961 грн.) і гаданий період розробки (3,63 міс.).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Що таке Vue.js? Внутрішня будова Vue.js. https://lemon.school/blog/shho-take-vue-js (дата звернення 15.05.2024)
2. Vue.js як JavaScript-фреймворк https://foxminded.ua/vue-js/ (дата звернення 15.05.2024)
3. Що таке Leaflet https://uk.wikipedia.org/wiki/Leaflet (дата звернення 15.05.2024)
4. Getting Started with Leaflet.js and React: Rendering a Simple Map https://coderlessons.com/tutorials/veb-razrabotka/uznaite-leafletjs/leafletjs-kratkoe-rukovodstvo (дата звернення 15.05.2024)
5. Технології візуалізації даних https://etk.lntu.edu.ua/pluginfile.php/25016/mod_resource/content/1/%D0%B7%D0%B0%D0%B9%D1%86%D0%B5%D0%B2.pdf (дата звернення 02.06.2024)
6. **EO Browser (https://apps.sentinel-hub.com/eo-browser/)** (дата звернення 02.06.2024)
7. **Crop Monitor (https://cropmonitor.org/)** (дата звернення 02.06.2024)
8. **Crop Monitor Exploring Tool (**https://cropmonitortools.org/tools/cmet/**)** (дата звернення 02.06.2024)
9. **AgriSense**(https://agrisense.iot.ubidots.com/app/dashboards/public/dashboard/Xke58VrfGkZvrNH5Nl-A6oLfGVM1oMlJrzNRPwk8oHA?datePicker=true) (дата звернення 02.06.2024)
10. Як створити веб-додаток: типи, переваги, принцип роботи https://wezom.com.ua/ua/blog/kak-sozdat-veb-prilozhenie (дата звернення 02.06.2024)
11. ОБРОБКА ТА ВИКОРИСТАННЯ МУЛЬТИСПЕКТРАЛЬНИХ ЗОБРАЖЕНЬ В АГРОМОНІТОРИНГУ https://ir.lib.vntu.edu.ua/bitstream/handle/123456789/31832/%D0%91%D1%96%D0%BB%D0%B8%D0%BD%D1%81%D1%8C%D0%BA%D0%B8%D0%B9.pdf?sequence=1 (дата звернення 07.06.2024)

12. GPS-моніторинг в агросекторі: сільгосптехніка як на долоні https://www.tehkontrol.ua/gps-monitoring-u-agrosectori.html (дата звернення 02.06.2024)

13. Flask & Vue. Завантаження файлів https://blog.avislab.com/flask-vue/example10/ (дата звернення 07.06.2024)

14. Сучасний підручник з JavaScript https://uk.javascript.info/ (дата звернення 07.06.2024)

15. Impact of Climate change on food systems https://www.researchgate.net/publication/337566958_Impact_of_Climate_change_on_food_systems (дата звернення 07.06.2024)

16. Climate Change and Land ipcc https://www.ipcc.ch/srccl/ (дата звернення 07.06.2024)

17. The Environmental Role of Protein Crops in the New Common Agricultural Policy https://www.europarl.europa.eu/RegData/etudes/etudes/join/2013/495856/IPOL-AGRI_ET(2013)495856_EN.pdf (дата звернення 12.06.2024)

18. Agriculture, The Path to Smart Farming: Innovations and Opportunities in Precision Agriculture https://www.mdpi.com/2077-0472/13/8/1593 (дата звернення 13.06.2024)

19. Data analytics for crop management: a big data view https://journalofbigdata.springeropen.com/articles/10.1186/s40537-022-00668-2 (дата звернення 14.06.2024)

20. What Is Data Visualization? Definition, Examples, And Learning Resources https://www.tableau.com/learn/articles/data-visualization (дата звернення 15.06.2024)

21. Applications of Remote Sensing in Precision Agriculture: A Review https://www.mdpi.com/2072-4292/12/19/3136 (дата звернення 15.06.2024)

22. What is Notepad++ https://notepad-plus-plus.org/ (дата звернення 15.06.2024)

23. Основи роботи з фреймворком Express.js https://foxminded.ua/express-js/ (дата звернення 15.06.2024)

24. Економ Salary statistics JavaScript. URL: https://builtin.com/salaries/dev-engineer/data-architect (дата звернення: 20.06.23)

25. Економ Тарифи на електроенергію для населення. https://index.minfin.com.ua/ua/tariff/electric/ (дата звернення 20.06.2024)

Додаток А

ЛІСТИНГ ПРОГРАМИ

MapView.vue:

```
<template>
 <div class="container">
  <!-- Бокова панель для відображення деталей маркера та коментарів -->
  <div class="sidebar" :class="{ 'sidebar-open': selectedMarker }">
   <!-- Секція даних маркера, видима, коли вибрано маркер -->
   <div v-if="selectedMarker" class="marker-data">
    <!-- Вкладки для перемикання між інформацією та коментарями -->
    <div class="tabs">
     <button @click="activeTab = 'info'" :class="{ active: activeTab === 'info' }">Інформація</button>
     <button @click="activeTab = 'comments'" :class="{ active: activeTab === 'comments'
}">Коментарі</button>
    </div>
    <!-- Відображення інформації про маркер, коли активна вкладка "Інформація" -->
    <div v-if="activeTab === 'info'">
     <h2>Інформація про ділянку</h2>
     <div class="info-box">
      <p><strong>Площа:</strong> {{ selectedMarker.area }} га</p>
      <p><strong>Температура:</strong> {{ selectedMarker.weather.temp }}°C</p>
      <p><strong>Опади:</strong> {{ selectedMarker.weather.precipitation }} мм</p>
      <p><strong>NDVI/EVI:</strong> {{ selectedMarker.num }}</p>
      <p><strong>NRI/DSWI:</strong> {{ selectedMarker.num2 }}</p>
      <p><strong>NDVI:</strong> {{ selectedMarker.NDVI.join('/') }}</p>
      <p><strong>EVI:</strong> {{ selectedMarker.EVI.join('/') }}</p>
      <p><strong>NRI:</strong> {{ selectedMarker.NRI.join('/') }}</p>
      <p><strong>DSWI:</strong> {{ selectedMarker.DSWI.join('/') }}</p>
      <p><strong>Ґрунт (поверхня):</strong> {{ selectedMarker.soil.surface }}°C</p>
      <p><strong>Ґрунт (10см):</strong> {{ selectedMarker.soil['10cm'] }}°C</p>
      <p><strong>Вологість ґрунту:</strong> {{ selectedMarker.soil.moisture }}%</p>
      <p><strong>Останнє оновлення:</strong> {{ selectedMarker.date }}</p>
     </div>
    </div>
    <!-- Відображення коментарів, коли активна вкладка "Коментарі" -->
    <div v-else-if="activeTab === 'comments'">
     <h2>Коментарі</h2>
     <div class="comments-list">
      <!-- Відображення списку коментарів -->
      <div v-for="(comment, index) in selectedMarker.comments" :key="index" class="comment">
       <strong>{{ comment.name }}</strong>
```

```
            <p>{{ comment.comment }}</p>
            <small>{{ comment.date }}</small>
          </div>
        </div>
        <div class="comment-form">
          <input v-model="commentName" placeholder="Ваше ім'я" />
          <textarea v-model="newComment" placeholder="Додайте коментар..."></textarea>
          <button @click="addComment" class="add-comment-btn">Додати коментар</button>
        </div>
      </div>
      <!-- Кнопка для закриття інформаційного блоку -->
      <button @click="closeMarkerInfo" class="close-btn">Закрити</button>
    </div>
    <!-- Привітальне повідомлення, якщо маркер не вибрано -->
    <div v-else class="welcome-message">
      <h2>Ласкаво просимо!</h2>
      <p>Виберіть маркер на карті, щоб побачити детальну інформацію.</p>
    </div>
  </div>
  <!-- Контейнер для карти -->
  <div class="map-container">
    <div id="map">
      <!-- Компонент карти з налаштуванням зума та центруванням -->
      <l-map :zoom="zoom" :center="center" style="height: 100%;" @click="closeMarkerInfo">
        <l-tile-layer :url="url" :attribution="attribution"></l-tile-layer>
        <!-- Додавання маркерів на карту -->
        <l-marker v-for="(marker, index) in markers" :key="index" :lat-lng="marker.position"
@click="selectMarker(marker, index)">
          <l-icon :icon-url="getMarkerIcon(index)" :icon-size="[25, 41]" :icon-anchor="[12, 41]"></l-icon>
        </l-marker>
      </l-map>
    </div>
    <!-- Легенда карти з поясненням кольорів маркерів -->
    <div class="map-legend">
      <h3>Пояснення міток:</h3>
      <div v-for="(color, index) in ['red', 'blue', 'green', 'orange', 'yellow', 'violet', 'grey', 'black']" :key="color"
class="legend-item">
        <img :src="getMarkerIcon(index)" alt="marker" width="20" height="30">
        <span>Ділянка {{ index + 1 }}</span>
      </div>
    </div>
    <!-- Легенда з поясненням термінів -->
```

```
      <div class="info-legend">
        <h3>Пояснення термінів:</h3>
        <p><strong>NDVI:</strong> Нормалізований відносний індекс рослинності</p>
        <p><strong>EVI:</strong> Розширений вегетаційний індекс</p>
        <p><strong>NRI:</strong> Нормалізований індекс червоного краю</p>
        <p><strong>DSWI:</strong> Індекс водного стресу</p>
        <p><strong>Ґрунт (поверхня):</strong> Температура поверхні ґрунту</p>
        <p><strong>Ґрунт (10см):</strong> Температура ґрунту на глибині 10 см</p>
        <p><strong>Вологість ґрунту:</strong> Відсоток вологи в ґрунті</p>
      </div>
    </div>
  </div>
</template>
<script>
import { LMap, LTileLayer, LMarker, LIcon } from '@vue-leaflet/vue-leaflet';
import 'leaflet/dist/leaflet.css';
export default {
  name: "MapView",
  components: {
    LMap,
    LTileLayer,
    LMarker,
    LIcon,
  },
  data() {
    return {
      zoom: 13, // Зум карти
      center: this.getCenter() || [48.5320, 35.0350], // Центрування карти, за замовчуванням Дніпро, Україна
      url: "https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png", // URL плитки карти
      attribution: '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a>
contributors', // Атрибуція карти
      markers: [], // Масив маркерів на карті
      selectedMarker: null, // Вибраний маркер
      selectedMarkerIndex: -1, // Індекс вибраного маркера
      activeTab: 'info', // Активна вкладка (інформація або коментарі)
      commentName: '', // Ім'я для нового коментаря
      newComment: '', // Новий коментар
    };
  },
  mounted() {
    this.fetchData(); // Завантаження даних при монтуванні компонента
    if (!this.getCenter()) {
```

```
    this.getCurrentLocation(); // Отримання поточного місцезнаходження, якщо не збережено центр
   }
  },
  methods: {
   async fetchData() {
    try {
     const response = await fetch("http://localhost:3000/data"); // Запит на отримання даних
     const data = await response.json();
     this.markers = data; // Присвоєння отриманих даних до маркерів
    } catch (error) {
     console.error("Error fetching data:", error); // Обробка помилки при завантаженні даних
    }
   },
   getCurrentLocation() {
    if (navigator.geolocation) {
     navigator.geolocation.getCurrentPosition(position => {
      this.center = [position.coords.latitude, position.coords.longitude]; // Встановлення центру карти на
поточне місцезнаходження
      localStorage.setItem('mapCenter', JSON.stringify(this.center)); // Збереження центру в localStorage
     }, error => {
      console.error("Error getting current location:", error); // Обробка помилки при отриманні
місцезнаходження
     });
    } else {
     console.error("Geolocation is not supported by this browser."); // Відображення помилки, якщо геолока


ція не підтримується
    }
   },
   getCenter() {
    const savedCenter = localStorage.getItem('mapCenter'); // Отримання збереженого центру з localStorage
    return savedCenter ? JSON.parse(savedCenter) : null; // Парсинг центру або повернення null, якщо не
знайдено
   },
   selectMarker(marker, index) {
    this.selectedMarker = marker; // Встановлення вибраного маркера
    this.selectedMarkerIndex = index; // Встановлення індексу вибраного маркера
    this.activeTab = 'info'; // Встановлення активної вкладки на "інформацію"
   },
   closeMarkerInfo() {
    this.selectedMarker = null; // Скидання вибраного маркера
    this.selectedMarkerIndex = -1; // Скидання індексу вибраного маркера
```

```
  },
  getMarkerIcon(index) {
    const colors = ['red', 'blue', 'green', 'orange', 'yellow', 'violet', 'grey', 'black']; // Кольори маркерів
    const color = colors[index % colors.length]; // Вибір кольору маркера за індексом
    return `https://raw.githubusercontent.com/pointhi/leaflet-color-markers/master/img/marker-icon-2x-${color}.png`; // Повернення URL іконки маркера
  },
  async addComment() {
    if (this.commentName.trim() && this.newComment.trim()) {
      try {
        const [lat, lon] = this.selectedMarker.position; // Отримання координат вибраного маркера
        const response = await fetch(`http://localhost:3000/comments/${lat}/${lon}`, {
          method: 'PUT',
          headers: {
            'Content-Type': 'application/json',
          },
          body: JSON.stringify({
            name: this.commentName,
            comment: this.newComment
          }),
        });

        if (!response.ok) {
          throw new Error('Failed to add comment'); // Помилка при додаванні коментаря
        }
        const result = await response.json();
        // Оновлення локального стану
        this.selectedMarker.comments = result.data.comments;
        this.commentName = '';
        this.newComment = '';
      } catch (error) {
        console.error('Error adding comment:', error); // Обробка помилки при додаванні коментаря
        alert('Failed to add comment. Please try again.'); // Повідомлення про помилку
      }
    } else {
      alert('Будь ласка, введіть ваше ім\'я та коментар.'); // Повідомлення про необхідність заповнення полів
    }
  },
 },
};
</script>
```

```
<style scoped>
.container {
  display: flex;
  height: 100vh;
  position: relative;
}
.sidebar {
  width: 0;
  overflow-x: hidden;
  transition: 0.5s;
  background-color: #f8f9fa;
  box-shadow: 2px 0 5px rgba(0,0,0,0.1);
}
.sidebar-open {
  width: 300px;
}
.map-container {
  flex-grow: 1;
  height: 100%;
  position: relative;
}
#map {
  height: 100%;
}
.marker-data {
  padding: 20px;
}
h2 {
  color: #333;
  margin-bottom: 15px;
  text-align: center;
}
.info-box, .comments-list {
  background-color: #ffffff;
  border: 1px solid #e0e0e0;
  border-radius: 5px;
  padding: 15px;
  margin-bottom: 15px;
}
.info-box p, .comment {
  margin: 5px 0;
}
```

```
.close-btn, .add-comment-btn {
  display: block;
  width: 100%;
  padding: 10px;
  background-color: #007bff;
  color: white;
  border: none;
  border-radius: 5px;
  cursor: pointer;
  font-size: 16px;
  margin-top: 10px;
}
.welcome-message {
  padding: 20px;
  text-align: center;
}
.welcome-message p {
  color: #6c757d;
}
.tabs {
  display: flex;
  margin-bottom: 15px;
}
.tabs button {
  flex: 1;
  padding: 10px;
  border: none;
  background-color: #e9ecef;
  cursor: pointer;
}
.tabs button.active {
  background-color: #007bff;
  color: white;
}
.comment-form input, .comment-form textarea {
  width: 100%;
  margin-bottom: 10px;
  padding: 5px;
  border: 1px solid #ced4da;
  border-radius: 5px;
}
.comment-form textarea {
```

```
  height: 100px;
}
.comment {
  border-bottom: 1px solid #e0e0e0;
  padding-bottom: 10px;
  margin-bottom: 10px;
  text-align: left;
}
.comment strong {
  display: block;
}
.comment small {
  color: #6c757d;
}
.map-legend, .info-legend {
  position: absolute;
  background-color: rgba(255, 255, 255, 0.8);
  padding: 10px;
  border-radius: 5px;
  max-width: 200px;
  z-index: 1000;
}
.map-legend {
  top: 10px;
  left: 10px;
}
.info-legend {
  top: 10px;
  right: 10px;
  max-width: 300px;
}
.legend-item {
  display: flex;
  align-items: center;
  margin-bottom: 5px;
}
.legend-item img {
  margin-right: 10px;
}
.info-legend h3, .map-legend h3 {
  margin-top: 0;
  margin-bottom: 10px;
```

```
}
.info-legend p {
  margin: 5px 0;
  font-size: 12px;
}
</style>
```

App.vue:

```
<template>
  <div id="app">
    <!-- Шапка з заголовком -->
    <header class="header">
      <div class="overlay">
        <h1>Агромоніторинг</h1>
      </div>
    </header>
    <!-- Компонент карти -->
    <MapView />
  </div>
</template>
<script>
import MapView from './components/MapView.vue';
export default {
  name: 'App', // Назва компонента
  components: {
    MapView // Підключення компонента MapView
  }
}
</script>
<style>
@import url('https://fonts.googleapis.com/css2?family=Roboto:wght@700&display=swap');
#app {
  font-family: 'Avenir', Helvetica, Arial, sans-serif; /* Базові шрифти */
  -webkit-font-smoothing: antialiased; /* Згладжування шрифтів для WebKit */
  -moz-osx-font-smoothing: grayscale; /* Згладжування шрифтів для Firefox */
  text-align: center; /* Центрування тексту */
  color: #2c3e50; /* Колір тексту */
  margin-top: 0; /* Відступ зверху */
}
.header {
  position: relative;
```

```
  background-image: url('@/assets/3.jpg'); /* Фонове зображення, скоригуйте шлях за потреби */
  background-size: cover; /* Масштабування зображення, щоб покрити всю область */
  background-position: center; /* Центрування фону */
  height: 175px; /* Висота заголовка */
  display: flex; /* Використання Flexbox для центрування */
  justify-content: center; /* Горизонтальне центрування */
  align-items: center; /* Вертикальне центрування */
  color: white; /* Колір тексту білий */
}
.overlay {
  background: rgba(0, 0, 0, 0.5); /* Напівпрозорий чорний фон */
  padding: 20px; /* Внутрішні відступи */
  width: 100%; /* Ширина 100% */
  display: flex; /* Використання Flexbox для центрування */
  justify-content: center; /* Горизонтальне центрування */
  align-items: center; /* Вертикальне центрування */
}
.header h1 {
  margin: 0; /* Відсутність зовнішніх відступів */
  font-size: 48px; /* Розмір шрифту */
  font-weight: bold; /* Жирний шрифт */
  font-family: 'Roboto', sans-serif; /* Користувацький шрифт */
  text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.4); /* Тінь для тексту */
  letter-spacing: 2px; /* Відстань між літерами */
}
</style>
```

Index.html:

```
<!DOCTYPE html>
<html lang="">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,initial-scale=1.0">
    <link rel="icon" href="<%= BASE_URL %>favicon.ico">
    <title><%= htmlWebpackPlugin.options.title %></title>
  </head>
  <body>
    <noscript>
      <strong>We're sorry but <%= htmlWebpackPlugin.options.title %> doesn't work properly without
JavaScript enabled. Please enable it to continue.</strong>
    </noscript>
```

```
    <div id="app"></div>
  </body>
</html>
```

Index.js:

```
const express = require('express'); // Імпорт бібліотеки Express для створення сервера
const cors = require('cors'); // Імпорт бібліотеки CORS для дозволу крос-доменних запитів
const fs = require('fs'); // Імпорт бібліотеки File System для роботи з файловою системою
const bodyParser = require('body-parser'); // Імпорт бібліотеки Body-Parser для парсингу тіла запитів
const app = express(); // Ініціалізація Express додатку
const port = 3000; // Порт, на якому буде запущено сервер
app.use(cors()); // Використання CORS для дозволу крос-доменних запитів
app.use(bodyParser.json()); // Використання Body-Parser для парсингу JSON запитів
const dataFile = __dirname + '/data.json'; // Шлях до файлу з даними
// Ендпоінт для отримання всіх даних
app.get('/data', (req, res) => {
  res.sendFile(dataFile); // Відправка файлу з даними у відповідь
});
// Ендпоінт для отримання даних за координатами
app.get('/data/:lat/:lon', (req, res) => {
  const lat = parseFloat(req.params.lat); // Отримання широти з параметрів запиту
  const lon = parseFloat(req.params.lon); // Отримання довготи з параметрів запиту
  fs.readFile(dataFile, 'utf8', (err, data) => {
    if (err) {
      return res.status(500).json({ message: 'Error reading data file' }); // Обробка помилки читання файлу
    }
    const jsonData = JSON.parse(data); // Парсинг даних з файлу
    const entry = jsonData.find(item => item.position[0] === lat && item.position[1] === lon); // Пошук запису
за координатами
    if (!entry) {
      return res.status(404).json({ message: `Data with position [${lat}, ${lon}] not found` }); // Обробка випадку,
коли дані не знайдено
    }
    res.json(entry); // Відправка знайденого запису у відповідь
  });
});
// Ендпоінт для додавання нових коментарів
app.put('/comments/:lat/:lon', (req, res) => {
  const lat = parseFloat(req.params.lat); // Отримання широти з параметрів запиту
  const lon = parseFloat(req.params.lon); // Отримання довготи з параметрів запиту
  const { comment, name } = req.body; // Отримання коментаря та імені з тіла запиту
  const currentDate = new Date().toLocaleString("uk-UA"); // Отримання поточної дати та часу
```

```
  fs.readFile(dataFile, 'utf8', (err, data) => {
    if (err) {
      return res.status(500).json({ message: 'Error reading data file' }); // Обробка помилки читання файлу
    }
    let jsonData = JSON.parse(data); // Парсинг даних з файлу
    const entryIndex = jsonData.findIndex(item => item.position[0] === lat && item.position[1] === lon); // Пошук індексу запису за координатами
    if (entryIndex === -1) {
      return res.status(404).json({ message: `Data with position [${lat}, ${lon}] not found` }); // Обробка випадку, коли дані не знайдено
    }
    // Додавання нового коментаря
    jsonData[entryIndex].comments.push({
      name: name,
      comment: comment,
      date: currentDate
    });
    // Запис оновлених даних у файл
    fs.writeFile(dataFile, JSON.stringify(jsonData, null, 2), 'utf8', (err) => {
      if (err) {
        return res.status(500).json({ message: 'Error writing data file' }); // Обробка помилки запису файлу
      }
      res.json({
        message: `New comment for position [${lat}, ${lon}] has been added`, // Повідомлення про успішне додавання коментаря
        data: jsonData[entryIndex] // Відправка оновлених даних у відповідь
      });
    });
  });
});
// Запуск сервера на вказаному порті
app.listen(port, () => {
  console.log(`Server running at http://localhost:${port}`); // Лог повідомлення про успішний запуск сервера
});
```

Data.json:

```
[
  {
    "position": [
      48.532,
      35.035
    ],
```

```
"area": 25.72,
"num": 6432,
"num2": 1626,
"NDVI": [
  0.09,
  0.23,
  0.68
],
"EVI": [
  0.1,
  0.16,
  0.69
],
"NRI": [
  0.07,
  0.28,
  0.53
],
"DSWI": [
  0.66,
  0.89,
  1.64
],
"weather": {
  "temp": 20.1,
  "precipitation": 7.67
},
"soil": {
  "surface": 32.6,
  "10cm": 27.1,
  "moisture": 11
},
"date": "jun 9, 2024",
"comments": [
  {
    "name": "Влад ",
    "comment": "Круто",
    "date": "28.06.2024, 13:28:18"
  },
  {
    "name": "Ден ",
    "comment": "1",
```

```
      "date": "28.06.2024, 13:54:13"
    }
  ]
},
{
  "position": [
    48.5375,
    35.027
  ],
  "area": 28.1,
  "num": 7014,
  "num2": 1754,
  "NDVI": [
    -0.04,
    0.51,
    0.78
  ],
  "EVI": [
    -0.08,
    0.47,
    0.87
  ],
  "NRI": [
    -0.11,
    0.47,
    0.6
  ],
  "DSWI": [
    0.74,
    1.15,
    2.1
  ],
  "weather": {
    "temp": 20.1,
    "precipitation": 7.67
  },
  "soil": {
    "surface": 32.6,
    "10cm": 27.1,
    "moisture": 11
  },
  "date": "jun 9, 2024",
```

```
    "comments": [
      {
        "name": "Влад ",
        "comment": "Круто1",
        "date": "28.06.2024, 13:28:35"
      }
    ]
  }
]
```

Додаток Б

ВІДГУК КЕРІВНИКА ЕКОНОМІЧНОГО РОЗДІЛУ

ДОДАТОК В

Перелік файлів на диску

ПЕРЕЛІК ДОКУМЕНТІВ НА МАГНІТНОМУ НОСІЇ

Ім’я файлу	Опис
Пояснювальні документи	
Кваліфікаційна робота.docx	Пояснювальна записка до кваліфікаційної роботи. Документ Word.
Кваліфікаційна робота.pdf	Пояснювальна записка до кваліфікаційної роботи в форматі PDF
Програма	
програма.rar	Архів. Містить коди програми і скомпільовану програму
Презентація	
презентація.ppt	Презентація кваліфікаційної роботи