

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Комп'ютерних наук і технологій
(повне найменування факультету)

Комп'ютерні системи та мережі
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалаврський
(ступінь вищої освіти)

на тему: РОЗРОБКА АПАРАТНО-ПРОГРАМНОГО КОМПЛЕКСУ
МОНІТОРИНГУ ТА СИТУАЦІЙНОЇ ОБІЗНАНОСТІ НА ОСНОВІ
ГЕОІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Виконав(ла): студент(ка) 4 курсу,
групи КНТз-512

Спеціальності

123 Комп'ютерна інженерія
(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерна інженерія
(назва освітньої програми (спеціалізації))

ОБРЄЗКОВ А.О.
(ПРИЗВИЩЕ та ініціали)

Керівник ТЯГУНОВА М.Ю.
(ПРИЗВИЩЕ та ініціали)

Рецензент РОМАНОВСЬКИЙ О.В.
(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет Комп'ютерних наук і технологій
Кафедра комп'ютерних систем та мереж
Ступінь вищої освіти бакалаврський
Спеціальність 123 Комп'ютерна інженерія
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерна інженерія
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри КУДЕРМЕТОВ Р.К.

«14» квітня 2026 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Обрезкова Антона Олександровича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка апаратно-програмного комплексу моніторингу та ситуаційної обізнаності на основі геоінформаційних технологій

керівник проєкту (роботи) к.т.н., доцент, ТЯГУНОВА М.Ю.

(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від «08» квітня 2026 року № 151

2. Строк подання студентом проєкту (роботи) 01.06.2026 р.

3. Вихідні дані до проєкту (роботи) опис предметної області, технології розробки клієнтської та серверної частини

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1) Аналіз предметної області;

2) Проєктування системи;

3) Реалізація системи;

4) Проєктування інтерфейсу користувача;

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4	ТЯГУНОВА М.Ю.		
Нормоконтроль	ДЬЯЧУК Т.С.		

7. Дата видачі завдання «14» квітня 2026 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Аналіз предметної області	до 18.04.2026	
2	Аналіз існуючих проблем та подібних рішень	до 20.04.2026	
3	Постановка задачі дипломного проєкту	до 22.04.2026	
4	Проектування загальної структури системи	до 24.04.2026	
5	Визначення функціональних та нефункціональних вимог	до 27.04.2026	
6	Опис функціонування системи та алгоритму її Роботи	до 02.05.2026	
7	Проектування архітектури програмного Забезпечення	до 07.05.2026	
8	Проектування бази даних та взаємодії компонентів	до 12.05.2026	
9	Реалізація системи	до 18.05.2026	
10	Проектування інтерфейсу користувача та Тестування системи	до 22.06.2026	
11	Оформлення пояснювальної записки та Проходження нормоконтролю	до 01.06.2026	
12	Проходження рецензування та перевірки на академічний плагіат	до 10.06.2026	

Студент(ка)

(підпис)

Антон ОБРЕЗКОВ

(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис)

Марія ТЯГУНОВА

(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи бакалаврського диплому:
108 с., 15 таблиць, 4 рисунки, 5 додатків, 30 джерел.

АПАРАТНО-ПРОГРАМНИЙ КОМПЛЕКС, ГІС-СИСТЕМИ,
ВЕБЗАСТОСУНОК, NODE.JS, POSTGRES SQL

Об'єктом розробки є апаратно-програмний комплекс моніторингу та ситуаційної обізнаності на основі геоінформаційних технологій.

Метою цієї роботи є спрощення процесу визначення та поширення інформації про події та об'єкти на визначених завданнях територіях.

У першому розділі аналізується предметна область, аналогічні системи, формулюється постановка проблеми та визначаються основні вимоги до програмного забезпечення.

У другому розділі описується операційна модель системи, варіанти використання та алгоритм роботи програмного забезпечення.

У третьому розділі розробляється архітектура системи, визначаються її апаратні та програмні компоненти, структура бази даних PostgreSQL.

У четвертому розділі описується інтерфейс користувача, основні екрани вебзастосунку та результати роботи системи.

В результаті було створено робочий прототип інформаційної системи, що дозволяє користувачам вводити та відображати інформацію про об'єкти на карті.

ЗМІСТ

Вступ.....	7
1 Аналіз апаратно-програмних комплексів моніторингу та ситуаційної обізнаності	10
1.1 Поняття та призначення систем моніторингу та ситуаційної обізнаності.....	10
1.2 Аналіз сучасних геоінформаційних систем та сервісів моніторингу	14
1.3 Архітектура апаратно-програмних комплексів моніторингу	17
1.4 Методи візуалізації та обробки геопросторових даних	20
1.5 Аналіз засобів реалізації веборієнтованих ГІС-систем.....	24
1.6 Постановка задачі дипломного проєкту	27
2 Проєктування системи.....	29
2.1 Визначення вимог до системи	29
2.2 Розробка структури системи та моделі взаємодії компонентів.....	33
2.3 Проєктування серверної частини комплексу	36
2.4 Проєктування клієнтської вебплатформи.....	42
2.5 Розробка системи авторизації та розмежування доступу	46
2.6 Реалізація механізму створення та керування шарами карти.....	51
2.7 Реалізація механізму додавання та редагування об'єктів на карті	55
2.8 Проєктування структури бази даних.....	59
2.9 Розробка механізму прикріплення медіафайлів до об'єктів.....	63
3 Реалізація та тестування апаратно-програмного комплексу	67
3.1 Вибір програмних засобів та технологій розробки	67
3.2 Реалізація серверної частини системи	70
3.3 Реалізація клієнтського інтерфейсу користувача	73
3.4 Інтеграція картографічного сервісу та ГІС-функціоналу	75
3.5 Реалізація системи ролей та керування доступом користувачів.....	77
3.6 Тестування функціональних можливостей системи	79
3.7 Аналіз результатів роботи та оцінка ефективності системи.....	81
4 Забезпечення безпеки, продуктивності та надійності апаратно-програмного комплексу	83
4.1 Аналіз загроз інформаційній безпеці геоінформаційної системи.....	83

4.2 Реалізація механізмів захисту користувачів та геоданих	85
4.3 Оптимізація продуктивності веборієнтованої ГІС системи	87
4.4 Забезпечення надійності та масштабованості системи	89
Висновки	92
Перелік джерел посилання	94
Додаток А UML-діаграми системи.....	98
Додаток Б Структура бази даних.....	99
Додаток В Фрагменти програмного коду	100
Додаток Г Інтерфейс користувача системи.....	101
Додаток Д Слайди презентації.....	102

ВСТУП

Актуальність теми дослідження зумовлена стрімким розвитком інформаційних технологій, геоінформаційних систем та засобів цифрового моніторингу, які активно використовуються у сфері безпеки, логістики, військового управління, цивільного захисту, екологічного контролю та управління інфраструктурою. У сучасних умовах зростає потреба у створенні програмних сервісів, здатних забезпечувати оперативний збір, обробку, візуалізацію та аналіз просторових даних у режимі реального часу. Особливо важливим є формування систем ситуаційної обізнаності, які дозволяють користувачам отримувати актуальну інформацію про об'єкти, події та зміни на визначеній території.

Сучасні геоінформаційні платформи на основі цифрових картографічних сервісів дозволяють реалізовувати багатофункціональні програмні рішення з підтримкою інтерактивної роботи із шарами даних, користувацькими мітками, мультимедійними вкладеннями та системами керування доступом. Такі системи активно використовуються в органах державного управління, військових структурах, службах реагування на надзвичайні ситуації, транспортних компаніях та аналітичних центрах. Разом із цим існує проблема створення спеціалізованих апаратно-програмних комплексів із закритим доступом, які забезпечують безпечну взаємодію між користувачами, контроль доступу до інформації та можливість адаптації системи під конкретні потреби організації.

Особливої актуальності набуває питання інтеграції ГІС технологій із сучасними вебтехнологіями та серверними рішеннями для реалізації захищених сервісів моніторингу територій та об'єктів. Використання інтерактивних картографічних платформ забезпечує можливість створення інформаційного середовища, у межах якого користувачі можуть додавати власні об'єкти на карту, редагувати шари, прикріплювати фотографії, формувати коментарі та здійснювати аналітичну роботу з просторовими даними. Це дозволяє значно підвищити ефективність прийняття рішень та покращити рівень ситуаційної обізнаності.

У сучасних умовах цифровізації суспільства та розвитку мережових технологій важливим є також питання захисту інформації та розмежування прав доступу між користувачами системи. Реалізація механізмів авторизації, створення ролей користувачів, обмеження доступу до окремих шарів інформації та забезпечення конфіденційності даних є необхідними складовими сучасних програмних комплексів моніторингу. Саме тому розробка апаратно-програмного комплексу моніторингу та ситуаційної обізнаності є актуальним завданням як з наукової, так і з практичної точки зору.

Метою дипломної роботи є розробка апаратно-програмного комплексу моніторингу та ситуаційної обізнаності на основі геоінформаційної системи з можливістю роботи із шарами, користувацькими об'єктами та механізмами керування доступом.

Об'єктом дослідження є процеси моніторингу, збору та обробки просторових даних у геоінформаційних системах.

Предметом дослідження є методи та засоби розробки комп'ютерних апаратно-програмних комплексів ситуаційної обізнаності із використанням інтерактивних картографічних сервісів та сучасних вебтехнологій.

Для досягнення поставленої мети у роботі необхідно виконати такі завдання:

- проаналізувати системи моніторингу та ситуаційної обізнаності та існуючі платформи;
- визначити вимоги до апаратно-програмного комплексу;
- розробити архітектуру системи моніторингу та ситуаційної обізнаності;
- реалізувати систему відповідно визначених вимог;
- провести тестування та оцінити ефективність та перспективи використання розробленої системи.

У процесі дослідження були використані методи системного аналізу, методи проектування інформаційних систем, методи об'єктно орієнтованого програмування, технології веброзробки, методи аналізу просторових даних, а також сучасні засоби створення геоінформаційних сервісів. Для реалізації

програмного комплексу можуть бути використані картографічні API, серверні технології, системи керування базами даних та засоби захисту інформації.

Практичне значення отриманих результатів полягає у можливості використання розробленого апаратно-програмного комплексу для моніторингу територій, управління інформаційними шарами, аналізу ситуаційної обстановки та організації захищеної взаємодії користувачів у межах єдиного інформаційного середовища. Розроблена система може бути адаптована для потреб органів місцевого самоврядування, служб безпеки, логістичних компаній, екологічного моніторингу та інших сфер діяльності, де необхідна оперативна робота з геопросторовою інформацією.

1 АНАЛІЗ АПАРАТНО-ПРОГРАМНИХ КОМПЛЕКСІВ МОНІТОРИНГУ ТА СИТУАЦІЙНОЇ ОБІЗНАНОСТІ

1.1 Поняття та призначення систем моніторингу та ситуаційної обізнаності

У сучасному інформаційному суспільстві особливого значення набувають системи моніторингу та ситуаційної обізнаності, які забезпечують збір, аналіз, обробку та відображення інформації про стан об'єктів, процесів і подій у реальному часі. Розвиток цифрових технологій, мережевих сервісів, геоінформаційних систем та засобів передачі даних сприяв активному впровадженню подібних систем у різних сферах діяльності людини. Вони використовуються у військовій сфері, логістиці, транспорті, цивільному захисті, охороні критичної інфраструктури, екологічному моніторингу, телекомунікаціях та системах управління містами.

Поняття моніторингу розглядається як процес безперервного спостереження за станом певного об'єкта або середовища з метою отримання актуальної інформації, її аналізу та своєчасного реагування на зміни. Основним завданням моніторингу є виявлення подій, відхилень або загроз, які можуть впливати на функціонування системи чи об'єкта спостереження. У технічних системах моніторинг забезпечує контроль параметрів обладнання, мереж, територій або інформаційних потоків, що дозволяє оперативно приймати рішення та мінімізувати ризики.

Ситуаційна обізнаність є більш комплексним поняттям і передбачає не лише отримання інформації, а й її інтерпретацію, оцінювання та прогнозування розвитку подій. У сучасних дослідженнях ситуаційна обізнаність визначається як здатність системи або користувача сприймати елементи навколишнього середовища, розуміти їх значення та прогнозувати подальший розвиток ситуації. Таким чином, система ситуаційної обізнаності повинна не тільки відображати дані, але й забезпечувати формування цілісної картини подій для підтримки процесу прийняття рішень.

Оснoву сучасних систем моніторингу та ситуаційної обізнаності становлять інформаційні технології, які забезпечують інтеграцію різних джерел даних у єдине інформаційне середовище. Джерелами інформації можуть бути датчики, камери спостереження, GPS трекери, геоінформаційні сервіси, мобільні пристрої, сервери баз даних та зовнішні інформаційні ресурси. Отримані дані обробляються програмними модулями та відображаються користувачеві у зручному вигляді, зокрема у формі інтерактивних карт, таблиць, графіків або інформаційних панелей [1].

Важливу роль у функціонуванні таких систем відіграють геоінформаційні технології. Саме інтеграція картографічних сервісів дозволяє реалізувати просторовий аналіз інформації та забезпечити візуалізацію об'єктів на карті. Використання ГІС технологій дає можливість працювати із шарами даних, додавати мітки, формувати маршрути, відображати зони ризику та аналізувати територіальні взаємозв'язки між об'єктами. У результаті користувач отримує більш повне уявлення про поточну ситуацію та може швидше приймати управлінські рішення.

Системи моніторингу та ситуаційної обізнаності мають складну багаторівневу структуру. Зазвичай вони включають рівень збору інформації, рівень передачі даних, серверний рівень обробки інформації та користувацький рівень візуалізації. На рівні збору інформації здійснюється отримання даних із різних джерел. Далі інформація передається каналами зв'язку на сервер, де виконується її аналіз, фільтрація, структуризація та збереження у базах даних. Після цього результати обробки відображаються користувачеві через вебінтерфейс або спеціалізоване програмне забезпечення (табл. 1.1) [2].

Таблиця 1.1 – Основні складові систем моніторингу та ситуаційної обізнаності

Складова системи	Характеристика	Призначення
Система моніторингу	Комплекс програмних та апаратних засобів для безперервного спостереження за об'єктами і процесами	Збір, передача та аналіз інформації у режимі реального часу
Ситуаційна обізнаність	Сукупність методів оцінювання поточної ситуації та прогнозування розвитку подій	Підтримка прийняття управлінських рішень
Геоінформаційна система (ГІС)	Система для роботи з просторовими даними та електронними картами	Візуалізація та аналіз об'єктів на карті
Картографічний сервіс	Онлайн або локальна платформа для відображення цифрових карт	Робота з шарами, маршрутами та мітками
Серверна частина	Програмний модуль для обробки, збереження та передачі інформації	Централізоване управління даними
База даних	Структуроване сховище інформації про об'єкти, координати та події	Збереження та швидкий доступ до даних
Користувацький інтерфейс	Вебінтерфейс або програмне середовище взаємодії користувача із системою	Перегляд, пошук та редагування інформації
Система авторизації	Механізм перевірки користувачів та їхніх прав доступу	Захист інформації та розмежування ролей
Інформаційні шари	Тематичні набори просторових даних	Організація та фільтрація інформації
Користувацькі об'єкти	Мітки, полігони, маршрути та інші елементи, створені користувачами	Відображення об'єктів спостереження
Мультимедійні вкладення	Фото, файли, відео та коментарі, пов'язані з об'єктами	Документування та деталізація подій
Аналітичний модуль	Програмні засоби для аналізу та оцінювання інформації	Формування звітів та прогнозування
Система оповіщення	Механізм автоматичного інформування користувачів про події	Оперативне реагування на загрози
Засоби безпеки	Комплекс методів захисту даних та мережевої взаємодії	Запобігання несанкціонованому доступу
Мережеві технології	Канали та протоколи передачі інформації	Обмін даними між компонентами системи
GPS та геолокація	Технології визначення координат об'єктів	Контроль переміщення та навігація
Хмарні сервіси	Віддалені платформи для обробки та зберігання інформації	Масштабованість та резервування даних
Мобільний доступ	Інтеграція системи з мобільними пристроями	Дистанційна робота користувачів

Особливістю сучасних систем є необхідність забезпечення багатокористувацької роботи та контролю доступу до інформації. У багатьох випадках дані мають обмежений доступ і можуть використовуватися лише певними категоріями користувачів. Саме тому важливими складовими систем ситуаційної обізнаності є механізми авторизації, автентифікації та розмежування прав доступу. Реалізація ролей користувачів дозволяє обмежувати доступ до окремих шарів інформації, редагування об'єктів або перегляду конфіденційних даних.

Сучасні системи моніторингу також повинні підтримувати можливість інтерактивної взаємодії з даними. Користувачі можуть створювати власні об'єкти на карті, додавати текстові коментарі, прикріплювати фотографії, редагувати інформацію та виконувати пошук за різними параметрами. Такий підхід значно підвищує ефективність роботи системи та дозволяє формувати актуальну базу просторових даних у реальному часі.

Однією з ключових характеристик систем ситуаційної обізнаності є оперативність. Інформація повинна оновлюватися максимально швидко, оскільки затримка в отриманні даних може призвести до втрати актуальності інформації та неправильного прийняття рішень. Особливо важливо це для систем військового моніторингу, аварійного реагування та управління критичною інфраструктурою. Не менш важливим є забезпечення надійності та безпеки функціонування системи. Сучасні програмні комплекси повинні бути захищені від несанкціонованого доступу, втрати даних та кібератак. Для цього використовуються механізми шифрування даних, захищені протоколи передачі інформації, резервне копіювання та системи аудиту дій користувачів [3].

1.2 Аналіз сучасних геоінформаційних систем та сервісів моніторингу

Сучасний розвиток інформаційних технологій сприяв активному поширенню геоінформаційних систем у різних сферах діяльності людини. Геоінформаційні системи забезпечують можливість збору, обробки, аналізу та візуалізації просторових даних, що дозволяє ефективно контролювати об'єкти та процеси у реальному часі. Сьогодні ГІС технології використовуються у сфері оборони, транспортній логістиці, містобудуванні, екологічному моніторингу, енергетиці, аграрному секторі, телекомунікаціях та системах управління надзвичайними ситуаціями.

Геоінформаційна система являє собою комплекс програмних, апаратних та інформаційних засобів, призначених для роботи з просторовими даними. Основною особливістю ГІС є поєднання електронної карти із базою даних, що дозволяє відображати інформацію у вигляді тематичних шарів. Такий підхід забезпечує можливість виконання просторового аналізу, побудови маршрутів, визначення координат, прогнозування ситуацій та підтримки управлінських рішень [4].

Одними з найпоширеніших сучасних геоінформаційних платформ є ArcGIS, QGIS, Google Maps Platform, OpenStreetMap, Mapbox та Leaflet. Кожна з цих систем має власні функціональні особливості, архітектуру та сферу застосування. Комерційні платформи орієнтовані на корпоративне використання та мають розширені аналітичні можливості, тоді як відкриті рішення забезпечують гнучкість інтеграції та адаптації під конкретні потреби користувачів.

Однією з найбільш відомих комерційних платформ є ArcGIS компанії Esri. Дана система використовується для професійного аналізу геопросторових даних, створення інтерактивних карт, роботи з великими обсягами інформації та організації багаторівневих систем моніторингу. ArcGIS підтримує інтеграцію з базами даних, хмарними сервісами та мобільними платформами. Система активно використовується державними установами, військовими структурами та великими

підприємствами для забезпечення ситуаційної обізнаності та управління територіями.

Важливе місце серед відкритих геоінформаційних систем займає QGIS. Це програмне забезпечення з відкритим кодом, яке підтримує роботу з просторовими шарами, картографічними сервісами, базами геоданих та аналітичними модулями. Головною перевагою QGIS є можливість безкоштовного використання та адаптації системи під конкретні завдання. Завдяки підтримці великої кількості плагінів система широко використовується у навчальних закладах, дослідницьких установах та проєктах із відкритими геоданими [5].

Одним із найбільш популярних сервісів вебкартографії є Google Maps Platform. Дана платформа забезпечує можливість інтеграції інтерактивних карт у вебсистеми та мобільні застосунки. Google Maps API дозволяє реалізувати функції визначення координат, побудови маршрутів, відображення користувацьких об'єктів, роботи з геолокацією та візуалізації даних у режимі реального часу. Основною перевагою сервісу є висока швидкість роботи, масштабованість та зручний інтерфейс для розробників. Саме тому Google Maps часто використовується як основа для створення систем моніторингу та ситуаційної обізнаності.

Відкритою альтернативою комерційним картографічним сервісам є OpenStreetMap. Даний проєкт представляє собою відкриту карту світу, яка наповнюється спільнотою користувачів. OpenStreetMap дозволяє використовувати картографічні дані без ліцензійних обмежень та інтегрувати їх у власні інформаційні системи. Завдяки відкритості та підтримці API сервіс активно використовується у системах моніторингу, навігації та кризового реагування.

Для реалізації сучасних веборієнтованих ГІС систем широко використовуються бібліотеки Leaflet та Mapbox. Leaflet є компактною JavaScript бібліотекою для створення інтерактивних карт у браузері. Вона підтримує роботу з шарами, мітками, полігонами, маршрутами та геоданими. Основною перевагою Leaflet є простота інтеграції та невисокі вимоги до ресурсів системи. У свою чергу

Марбох надає розширені можливості стилізації карт, підтримку 3D візуалізації та інструменти для роботи з великими масивами просторових даних. Сучасні сервіси моніторингу будуються на основі концепції Web GIS, що передбачає використання геоінформаційних систем через вебтехнології. Такий підхід дозволяє забезпечити віддалений доступ користувачів до картографічних даних та організувати багатокористувацьку роботу із системою. Web GIS підтримує інтеграцію з мобільними пристроями, GPS обладнанням, сенсорами та хмарними платформами.

У сучасних умовах особливого значення набувають системи ситуаційної обізнаності військового та безпекового призначення. Подібні системи забезпечують відображення оперативної обстановки, аналіз ризиків, контроль переміщення об'єктів та координацію дій користувачів. Геоінформаційні технології дозволяють інтегрувати дані із супутникових систем, безпілотних літальних апаратів, камер спостереження та сенсорних мереж у єдине інформаційне середовище.

Сучасні системи моніторингу також активно використовують хмарні технології та засоби обробки великих даних. Використання хмарних платформ забезпечує масштабованість системи, підвищення швидкості обробки інформації та можливість централізованого зберігання геопросторових даних. Перспективним напрямом розвитку є інтеграція ГІС технологій із засобами штучного інтелекту та машинного навчання, що дозволяє автоматизувати процес аналізу даних та прогнозування подій.

Аналіз сучасних геоінформаційних систем показує, що більшість платформ підтримують роботу із шарами даних, користувацькими об'єктами, мультимедійними вкладеннями та механізмами контролю доступу. Саме ці функції є ключовими для створення апаратно-програмного комплексу моніторингу та ситуаційної обізнаності. Водночас існує потреба у розробці спеціалізованих систем із закритим доступом, які можуть бути адаптовані під вимоги конкретної організації та забезпечувати високий рівень інформаційної безпеки.

1.3 Архітектура апаратно-програмних комплексів моніторингу

Сучасні апаратно-програмні комплекси моніторингу являють собою багатокомпонентні інформаційні системи, призначені для збору, передачі, обробки, аналізу та відображення даних про стан об'єктів, процесів або територій у режимі реального часу. Основною метою побудови таких комплексів є забезпечення безперервного контролю за інформаційним середовищем, оперативного реагування на зміни ситуації та підтримки процесу прийняття рішень. Архітектура подібних систем формується з урахуванням функціональних вимог, масштабу системи, кількості користувачів, типів джерел інформації та рівня захисту даних.

Архітектура апаратно-програмного комплексу визначає структуру взаємодії між його складовими елементами, принципи обробки інформації та механізми обміну даними. У більшості випадків сучасні системи моніторингу будуються за багаторівневим принципом, який забезпечує модульність, масштабованість та можливість інтеграції нових компонентів без суттєвої зміни всієї системи [6].

Типова архітектура комплексу моніторингу включає рівень збору даних, рівень передачі інформації, серверний рівень обробки даних та користувацький рівень взаємодії із системою. Кожен із цих рівнів виконує окремі функції та забезпечує стабільну роботу всього програмно-апаратного комплексу.

На рівні збору даних здійснюється отримання інформації із зовнішніх джерел. Джерелами можуть бути датчики, GPS трекери, камери відеоспостереження, геолокаційні сервіси, мобільні пристрої, мережеве обладнання або інші технічні засоби. У системах ситуаційної обізнаності особливу роль відіграють геопросторові дані, які містять координати об'єктів та дозволяють відображати їх на цифровій карті. Саме на цьому рівні формується первинний потік інформації, що надалі передається для обробки.

Другим рівнем є система передачі даних. Вона забезпечує зв'язок між пристроями збору інформації та центральним сервером. Для передачі інформації можуть використовуватися локальні мережі, Інтернет, бездротові технології, мобільні

мережі або супутникові канали зв'язку. Основними вимогами до даного рівня є висока швидкість передачі інформації, стабільність зв'язку та захист даних від несанкціонованого доступу.

Центральним елементом архітектури є серверний рівень. Сервер виконує функції приймання, аналізу, фільтрації, збереження та обробки інформації. Саме на серверному рівні реалізуються алгоритми роботи системи, логіка обробки подій та управління користувацьким доступом. Серверна частина може бути реалізована у вигляді локального серверного обладнання або хмарної інфраструктури. Використання хмарних технологій дозволяє забезпечити масштабованість системи, резервне копіювання даних та доступ користувачів із різних географічних точок [7].

Для зберігання інформації використовуються системи керування базами даних. База даних є важливою складовою архітектури, оскільки саме вона забезпечує структуроване зберігання координат, описів об'єктів, фотографій, коментарів, журналів подій та історії змін. У сучасних системах моніторингу часто використовуються реляційні та нереляційні бази даних, які забезпечують високу швидкість роботи із великими масивами інформації.

Особливе місце в архітектурі займає геоінформаційний модуль. Його призначення полягає у візуалізації даних на цифровій карті та роботі із просторовими шарами. Геоінформаційний модуль дозволяє відображати мітки, маршрути, полігони, зони ризику та інші об'єкти. Для реалізації подібних функцій використовуються картографічні сервіси та API, зокрема Google Maps, OpenStreetMap, Leaflet або Mapbox. Інтеграція картографічних технологій значно підвищує рівень ситуаційної обізнаності користувачів та дозволяє швидше оцінювати поточну ситуацію.

Користувацький рівень забезпечує взаємодію операторів або користувачів із системою. Найчастіше він реалізується у вигляді вебінтерфейсу або мобільного застосунку. Через користувацький інтерфейс здійснюється перегляд інформації, робота із шарами карти, додавання нових об'єктів, редагування даних, формування

звітів та отримання повідомлень про події. Важливою вимогою до користувацького інтерфейсу є простота використання, швидкодія та можливість адаптації до різних типів пристроїв.

У сучасних системах моніторингу важливим елементом архітектури є модуль безпеки. Його основним завданням є захист інформації, перевірка користувачів та контроль доступу до даних. Для цього використовуються механізми автентифікації, авторизації, шифрування даних та журналювання дій користувачів. У багатокористувацьких системах реалізується поділ ролей, який дозволяє визначати права доступу до певних функцій або інформаційних шарів.

Однією з характерних особливостей сучасних апаратно-програмних комплексів є підтримка роботи в реальному часі. Для цього використовуються технології асинхронного обміну даними, вебсокетти та механізми потокової передачі інформації. Завдяки цьому користувачі можуть отримувати актуальну інформацію без необхідності постійного оновлення сторінки або повторного запиту до сервера (рис. 1.1).

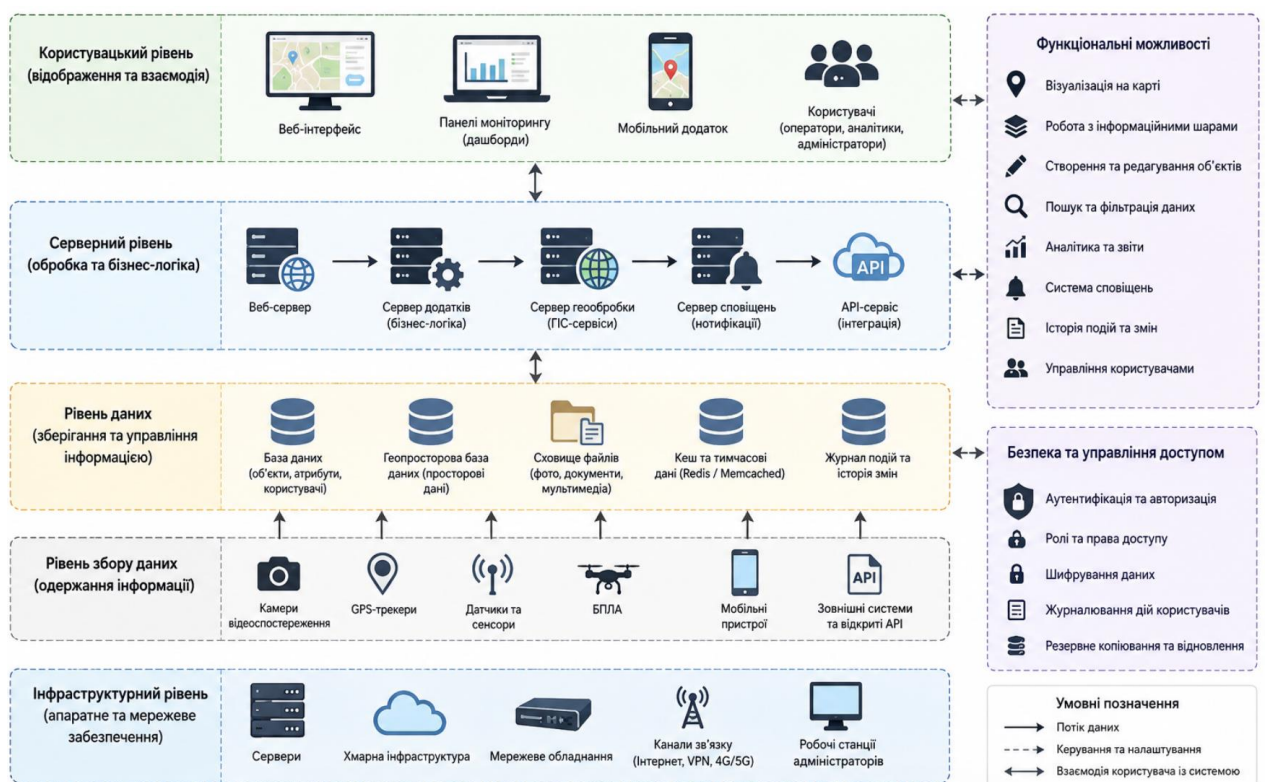


Рисунок 1.1 – Архітектура АПК моніторингу

Сучасні комплекси моніторингу також підтримують інтеграцію із зовнішніми інформаційними системами. Це дозволяє об'єднувати дані з різних джерел у єдиному середовищі та формувати більш повну картину ситуації. Інтеграція може здійснюватися через API інтерфейси, мережеві протоколи або обмін файлами даних.

Важливим напрямом розвитку архітектури апаратно-програмних комплексів є використання мікросервісного підходу. У такій архітектурі система поділяється на окремі незалежні модулі, кожен із яких виконує власну функцію. Це забезпечує спрощення масштабування, оновлення компонентів та підвищення стійкості системи до відмов.

1.4 Методи візуалізації та обробки геопросторових даних

У сучасних інформаційних системах особливе значення мають методи візуалізації та обробки геопросторових даних, оскільки саме вони забезпечують ефективне представлення інформації про об'єкти, події та процеси, пов'язані з певною територією. Геопросторові дані являють собою інформацію, яка містить координатну прив'язку до конкретного місця на поверхні Землі та дозволяє визначати розташування об'єктів у просторі. Такі дані активно використовуються у геоінформаційних системах, навігаційних сервісах, системах моніторингу, транспортній логістиці, військових інформаційних системах та системах ситуаційної обізнаності.

Основною метою візуалізації геопросторових даних є забезпечення зручного сприйняття інформації користувачем та підвищення ефективності аналізу просторових процесів. Візуальне представлення даних дозволяє швидко оцінювати поточну ситуацію, виявляти закономірності, аналізувати взаємозв'язки між об'єктами та приймати оперативні рішення. Для цього у сучасних системах використовуються електронні карти, тематичні шари, інтерактивні панелі, графіки,

схеми та тривимірні моделі [8].

Одним із найпоширеніших методів візуалізації є картографічне відображення даних. У цьому випадку інформація представляється у вигляді цифрової карти, на якій розміщуються об'єкти із прив'язкою до географічних координат. Карта може містити дороги, будівлі, інженерні мережі, маршрути, зони ризику, мітки користувачів та інші просторові елементи. Основною перевагою такого підходу є можливість одночасного аналізу великої кількості інформації у зрозумілому графічному вигляді.

Важливим методом організації геопросторових даних є використання тематичних шарів. Кожен шар містить окремий тип інформації, наприклад транспортну інфраструктуру, мережі зв'язку, системи відеоспостереження або місця розташування об'єктів моніторингу. Шарова структура дозволяє користувачеві вмикати або вимикати необхідні набори даних та формувати індивідуальне інформаційне середовище відповідно до поставлених завдань.

Для візуалізації об'єктів на карті використовуються різні графічні елементи. Найпростішими є маркери або мітки, що відображають координати окремих об'єктів. Для відображення територій або зон використовуються полігони, а для побудови маршрутів та напрямків руху застосовуються лінійні об'єкти. У сучасних системах також використовуються теплові карти, які дозволяють відображати інтенсивність подій або концентрацію певних показників на визначеній території.

Одним із важливих напрямів розвитку геоінформаційних технологій є тривимірна візуалізація. Використання 3D моделей дозволяє формувати більш реалістичне представлення об'єктів та середовища. Тривимірна візуалізація активно застосовується у містобудуванні, системах навігації, військових симуляторах та цифрових моделях місцевості. Завдяки використанню сучасних графічних технологій користувачі можуть отримувати більш детальну інформацію про рельєф місцевості, висоту будівель та просторове розташування об'єктів [9].

Для реалізації інтерактивної візуалізації у вебсистемах використовуються спеціалізовані картографічні бібліотеки та сервіси. Найбільш поширеними є

Leaflet, OpenLayers, Mapbox та Google Maps API. Ці інструменти дозволяють створювати інтерактивні карти із підтримкою масштабування, навігації, додавання користувацьких об'єктів, відображення шарів та роботи із геопросторовими даними у режимі реального часу.

Окрім візуалізації, важливе значення мають методи обробки геопросторових даних. Обробка даних включає збір інформації, її аналіз, фільтрацію, класифікацію, збереження та підготовку до відображення. У сучасних системах геодані можуть надходити із різних джерел, зокрема GPS пристроїв, датчиків, супутникових систем, безпілотних літальних апаратів та мобільних застосунків.

Одним із базових методів обробки є геокодування. Даний процес передбачає перетворення текстових адрес або описів місць у географічні координати. Зворотний процес називається зворотним геокодуванням і дозволяє отримувати адресу або опис об'єкта за його координатами.

Важливим методом аналізу є просторове фільтрування даних. Воно дозволяє здійснювати пошук об'єктів у певній зоні, визначати відстані між об'єктами, аналізувати маршрути та встановлювати територіальні взаємозв'язки.

Просторовий аналіз активно використовується у транспортних системах, логістиці, системах реагування на надзвичайні ситуації та військових інформаційних комплексах. У сучасних ГІС технологіях також застосовуються методи кластеризації геоданих. Кластеризація дозволяє групувати об'єкти залежно від їхнього розташування або характеристик. Це значно спрощує роботу із великою кількістю міток на карті та підвищує швидкість візуального аналізу інформації. Для забезпечення роботи систем у режимі реального часу використовуються методи потокової обробки даних. Інформація, що надходить із сенсорів або GPS пристроїв, миттєво аналізується сервером та відображається користувачам без затримок. У таких системах важливе значення мають оптимізація передачі даних, кешування інформації та використання асинхронних технологій обробки запитів (табл. 1.2).

Таблиця 1.2 – Методи візуалізації та обробки геопросторових даних

Метод візуалізації або обробки	Характеристика методу	Призначення у системах моніторингу
Картографічне відображення	Відображення даних на цифровій карті із координатною прив'язкою	Візуалізація об'єктів та подій у просторі
Тематичні шари	Розподіл інформації за окремими категоріями	Організація та фільтрація геоданих
Маркери та мітки	Графічні позначення координат об'єктів	Відображення місцезнаходження об'єктів
Полігони	Графічні області на карті	Позначення зон контролю або ризику
Лінійні об'єкти	Відображення маршрутів та напрямків руху	Аналіз переміщення та навігації
Теплові карти	Відображення інтенсивності подій кольоровими зонами	Аналіз концентрації даних на території
3D візуалізація	Тривимірне представлення просторових об'єктів	Аналіз рельєфу та інфраструктури
Геокодування	Перетворення адрес у географічні координати	Пошук та локалізація об'єктів
Зворотне геокодування	Визначення адреси за координатами	Ідентифікація місця розташування
Просторове фільтрування	Вибір об'єктів за територіальними параметрами	Пошук даних у визначеній зоні
Кластеризація об'єктів	Групування великої кількості міток	Оптимізація відображення даних
Буферний аналіз	Побудова зон навколо об'єктів	Визначення зон впливу або ризику
Просторовий аналіз	Аналіз взаємозв'язків між об'єктами	Підтримка прийняття рішень
Потокова обробка даних	Аналіз інформації у режимі реального часу	Оперативне оновлення даних
Кешування геоданих	Тимчасове збереження даних для прискорення доступу	Підвищення швидкодії системи
Інтерактивна візуалізація	Динамічна взаємодія користувача із картою	Масштабування та редагування даних
Обробка GPS даних	Аналіз координат рухомих об'єктів	Контроль маршрутів та навігації
Машинне навчання	Автоматичний аналіз та прогнозування подій	Виявлення аномалій та ризиків
Просторові бази даних	Спеціалізоване зберігання геоданих	Швидка обробка просторових запитів
API картографічних сервісів	Інтерфейси інтеграції картографічних платформ	Реалізація веборієнтованих ГІС систем

У сучасних системах ситуаційної обізнаності активно використовуються методи машинного навчання та штучного інтелекту для аналізу геопросторових даних. Алгоритми машинного навчання дозволяють прогнозувати розвиток подій, визначати потенційні загрози, аналізувати маршрути переміщення об'єктів та автоматично виявляти аномалії у просторових даних.

Не менш важливим є питання оптимізації зберігання геопросторових даних. Для цього використовуються спеціалізовані формати зберігання геоданих та просторові бази даних, зокрема PostgreSQL із розширенням PostGIS. Подібні системи забезпечують високу швидкість обробки просторових запитів та підтримують роботу із великими масивами географічної інформації.

1.5 Аналіз засобів реалізації веборієнтованих ГІС-систем

Сучасний розвиток інформаційних технологій та мережевих сервісів сприяв активному поширенню веборієнтованих геоінформаційних систем, які забезпечують віддалений доступ до просторових даних через веббраузер або мобільний застосунок. Веборієнтовані ГІС системи дозволяють організувати централізовану роботу із геоданими, забезпечити багатокористувацький доступ до інформації та реалізувати інтерактивну взаємодію користувачів із цифровими картами у режимі реального часу. Такі системи широко застосовуються у сфері моніторингу територій, логістики, транспортних сервісів, безпеки, екологічного контролю та управління інфраструктурою.

Основною особливістю веборієнтованих ГІС систем є використання клієнтсерверної архітектури. У такій архітектурі серверна частина забезпечує обробку, збереження та передачу геопросторових даних, а клієнтська частина реалізує інтерфейс взаємодії користувача із системою через веббраузер. Це дозволяє забезпечити доступ до інформації незалежно від операційної системи або типу пристрою [10].

Для створення веборієнтованих ГІС систем використовуються різноманітні програмні засоби, бібліотеки, фреймворки та картографічні сервіси. Вибір конкретних технологій залежить від функціональних вимог системи, масштабів проєкту, рівня безпеки та необхідної продуктивності.

Одним із найбільш поширених засобів реалізації веборієнтованих ГІС систем є Google Maps Platform. Дана платформа надає набір API для інтеграції інтерактивних карт у вебзастосунки. Google Maps API дозволяє реалізовувати функції відображення карт, побудови маршрутів, геолокації, роботи із маркерами, шарами та просторовими об'єктами. Основними перевагами сервісу є висока швидкодія, стабільність роботи та наявність великої кількості готових інструментів для розробників [11].

Популярною відкритою альтернативою є OpenStreetMap. Цей сервіс представляє собою відкриту картографічну платформу, яка дозволяє безкоштовно використовувати геодані у власних інформаційних системах. OpenStreetMap активно використовується у веборієнтованих ГІС завдяки відкритості, можливості модифікації карт та підтримці інтеграції із різними бібліотеками веброзробки.

Для створення клієнтської частини вебГІС систем широко застосовується бібліотека Leaflet. Вона є однією з найпопулярніших JavaScript бібліотек для роботи з інтерактивними картами. Leaflet забезпечує підтримку масштабування карти, роботи із шарами, мітками, маршрутами, полігонами та мультимедійними об'єктами. Основною перевагою бібліотеки є простота використання, невеликий розмір та висока швидкість роботи навіть на мобільних пристроях.

Ще одним сучасним засобом є Mapbox. Дана платформа надає розширені можливості стилізації карт та підтримує роботу із тривимірною графікою. Mapbox дозволяє створювати сучасні інтерфейси із високим рівнем візуалізації та підтримує роботу із великими обсягами геопросторових даних. Платформа активно використовується у навігаційних системах, транспортних сервісах та аналітичних вебплатформах [12].

Для реалізації складних геоінформаційних сервісів використовується

OpenLayers. Це потужна JavaScript бібліотека, яка підтримує роботу із великою кількістю картографічних форматів та геосервісів. OpenLayers забезпечує розширені функції аналізу геоданих, інтеграцію із WMS та WFS сервісами, а також підтримує роботу із геопросторовими стандартами.

Серверна частина веборієнтованих ГІС систем зазвичай реалізується із використанням сучасних серверних технологій та фреймворків. Для створення API та бізнеслогіки часто використовуються Node.js, Django, Flask, ASP.NET або Spring Boot. Ці технології дозволяють реалізовувати механізми авторизації, управління користувачами, обробку просторових запитів та інтеграцію із базами даних.

Для зберігання геопросторових даних найбільш поширеним рішенням є PostgreSQL із розширенням PostGIS. PostGIS забезпечує підтримку просторових типів даних, геометричних об'єктів та просторових запитів. Це дозволяє ефективно працювати із координатами, маршрутами, зонами та іншими геопросторовими елементами. У деяких системах також використовуються MongoDB або Firebase для роботи із нереляційними структурами даних та забезпечення роботи у режимі реального часу. Для організації взаємодії між клієнтом та сервером застосовуються REST API або GraphQL. REST API є найбільш поширеним підходом до побудови вебсервісів і дозволяє передавати дані між компонентами системи у форматі JSON. GraphQL забезпечує більш гнучку структуру запитів та дозволяє клієнту отримувати лише необхідну інформацію [13].

Важливу роль у реалізації вебГІС систем відіграють засоби безпеки та контролю доступу. У сучасних системах використовуються JWT токени, OAuth авторизація, HTTPS шифрування та механізми розмежування ролей користувачів. Це особливо важливо для систем моніторингу та ситуаційної обізнаності із закритим доступом. Для роботи у режимі реального часу застосовуються технології WebSocket та Socket.IO. Вони забезпечують миттєве оновлення інформації на карті без необхідності перезавантаження сторінки. Подібний підхід використовується у системах відстеження транспорту, моніторингу переміщення об'єктів та системах реагування на надзвичайні ситуації. У сучасних веборієнтованих ГІС системах

також активно використовуються хмарні сервіси. Платформи AWS, Google Cloud та Microsoft Azure дозволяють забезпечити масштабованість, резервне копіювання та стабільну роботу системи при великій кількості користувачів. Використання хмарних технологій значно спрощує розгортання та підтримку геоінформаційних сервісів.

Аналіз сучасних засобів реалізації веборієнтованих ГІС систем показує, що найбільш ефективними є комплексні рішення, які поєднують інтерактивні картографічні бібліотеки, серверні технології, просторові бази даних та засоби забезпечення безпеки. Саме такий підхід дозволяє створювати сучасні системи моніторингу та ситуаційної обізнаності із підтримкою багатокористувацької роботи, інтерактивної візуалізації та оперативної обробки геопросторових даних.

1.6 Постановка задачі дипломного проєкту

У сучасних умовах цифровізації інформаційного середовища та стрімкого розвитку геоінформаційних технологій особливої актуальності набуває створення ефективних систем моніторингу та ситуаційної обізнаності, здатних забезпечувати оперативний збір, аналіз і візуалізацію просторових даних у режимі реального часу. Існуючі програмні рішення часто орієнтовані на вузькоспеціалізовані завдання, мають обмежені можливості адаптації або потребують значних фінансових витрат для впровадження та підтримки. У багатьох випадках відсутня можливість створення закритих багатокористувацьких систем із власною структурою доступу та інтеграцією спеціалізованих функцій моніторингу.

Аналіз сучасних геоінформаційних платформ та сервісів показав, що для реалізації систем ситуаційної обізнаності необхідне поєднання засобів візуалізації просторових даних, серверної обробки інформації, механізмів роботи у режимі реального часу та системи керування користувацьким доступом. Особливе значення має забезпечення інтерактивної взаємодії користувачів із цифровою

картою, підтримка тематичних шарів, можливість створення власних об'єктів та інтеграція мультимедійної інформації.

У межах дипломного проєкту необхідно розробити апаратно-програмний комплекс моніторингу та ситуаційної обізнаності на основі веборієнтованої геоінформаційної системи. Розроблювана система повинна забезпечувати відображення цифрової карти, підтримку багатокористувацького режиму роботи, можливість створення та редагування просторових об'єктів, організацію роботи із шарами даних та реалізацію механізмів захисту інформації.

Основною задачею дипломного проєкту є створення програмного комплексу, який забезпечуватиме централізоване зберігання геопросторових даних, їх обробку та візуалізацію через вебінтерфейс. Система повинна підтримувати роботу із географічними координатами, мітками, маршрутами, полігонами та інформаційними шарами. Крім цього, необхідно забезпечити можливість додавання користувацьких коментарів, фотографій та іншої супровідної інформації до об'єктів моніторингу. Для реалізації поставленої мети необхідно вирішити низку технічних та програмних задач. Першочерговим етапом є аналіз сучасних методів побудови геоінформаційних систем та визначення оптимальної архітектури апаратно-програмного комплексу. Після цього необхідно виконати проєктування структури системи, визначити основні функціональні модулі та реалізувати механізми взаємодії між ними.

Одним із ключових завдань є розробка серверної частини системи, яка забезпечуватиме обробку запитів користувачів, збереження геопросторових даних та управління доступом до інформації. Для цього необхідно реалізувати базу даних із підтримкою просторових об'єктів та створити API для взаємодії між сервером і клієнтською частиною системи.

Не менш важливим завданням є створення клієнтської частини веборієнтованої ГІС системи. Інтерфейс користувача повинен забезпечувати інтерактивну роботу із картою, підтримку масштабування, фільтрацію інформаційних шарів, додавання нових об'єктів та перегляд детальної інформації

про них. Особлива увага повинна бути приділена швидкодії системи та зручності користувацького інтерфейсу.

Для забезпечення ситуаційної обізнаності система повинна підтримувати оновлення інформації у режимі реального часу. Це передбачає використання сучасних технологій передачі даних та реалізацію механізмів оперативного відображення змін на цифровій карті. У процесі розробки також необхідно реалізувати механізми автентифікації та авторизації користувачів. Система повинна підтримувати поділ ролей та забезпечувати контроль доступу до окремих функцій і шарів інформації. Це є особливо важливим для захищених систем моніторингу із багатокористувацьким режимом роботи.

Після завершення розробки необхідно провести тестування функціональних можливостей апаратно-програмного комплексу, перевірити стабільність роботи системи, швидкодію, коректність обробки геопросторових даних та ефективність механізмів захисту інформації.

Очікуваним результатом дипломного проєкту є створення працездатної веборієнтованої геоінформаційної системи моніторингу та ситуаційної обізнаності, яка забезпечуватиме інтерактивну роботу із просторовими даними, підтримку багатокористувацького режиму, відображення інформації у режимі реального часу та можливість адаптації під потреби конкретної організації або сфери застосування.

2 ПРОЄКТУВАННЯ СИСТЕМИ

2.1 Визначення вимог до системи

Формування функціональних вимог є одним із найважливіших етапів проєктування апаратно-програмного комплексу моніторингу та ситуаційної обізнаності. Саме на цьому етапі визначаються основні можливості системи, принципи її роботи, структура взаємодії між користувачами та програмними модулями, а також перелік функцій, які повинні бути реалізовані для забезпечення

ефективної роботи комплексу. Від правильності формування функціональних вимог залежить стабільність функціонування системи, її масштабованість, зручність використання та відповідність поставленим задачам.

У процесі аналізу сучасних геоінформаційних систем було встановлено, що програмний комплекс моніторингу повинен забезпечувати централізовану роботу із просторовими даними, підтримувати багатокористувацький режим роботи та надавати можливість оперативного оновлення інформації у режимі реального часу. Основою системи повинна стати інтерактивна цифрова карта, яка забезпечуватиме візуалізацію об'єктів моніторингу, інформаційних шарів та супровідних даних [14].

Однією з головних функціональних вимог є реалізація модуля авторизації та автентифікації користувачів. Система повинна підтримувати створення облікових записів, вхід користувачів до системи та перевірку прав доступу до інформації. Для забезпечення безпеки необхідно реалізувати механізми розмежування ролей користувачів. Адміністратор системи повинен мати можливість створювати нові облікові записи, змінювати права доступу та контролювати роботу користувачів. Звичайні користувачі повинні мати доступ лише до тих функцій та інформаційних шарів, які визначені їхніми ролями.

Важливою функціональною вимогою є підтримка роботи із цифровою картою. Користувач повинен мати можливість масштабувати карту, змінювати режим відображення, здійснювати навігацію по території та працювати із тематичними шарами. Система повинна підтримувати відображення різних типів об'єктів, зокрема міток, маршрутів, полігонів та зон контролю.

Для забезпечення ситуаційної обізнаності необхідно реалізувати механізм створення та редагування користувацьких об'єктів. Користувачі повинні мати можливість додавати нові об'єкти на карту, задавати координати, вказувати назву, опис та категорію об'єкта. Крім цього, система повинна підтримувати прикріплення мультимедійних матеріалів, зокрема фотографій та файлів.

Важливою вимогою є підтримка тематичних інформаційних шарів. Кожен шар повинен містити окрему категорію просторових даних та мати можливість

незалежного увімкнення або вимкнення. Подібний підхід дозволяє організувати структуру інформації та спростити аналіз даних користувачами.

Система повинна забезпечувати збереження геопросторових даних у централізованій базі даних. Для цього необхідно реалізувати механізми запису, редагування та видалення інформації про об'єкти моніторингу. База даних повинна підтримувати роботу із координатами, текстовими описами, часовими мітками та мультимедійними вкладеннями [15].

Однією з ключових функціональних вимог є забезпечення роботи системи у режимі реального часу. Усі зміни, які вносяться користувачами або надходять із зовнішніх джерел, повинні оперативно відображатися на карті без необхідності перезавантаження сторінки. Для цього необхідно використовувати сучасні технології обміну даними між сервером та клієнтською частиною системи.

Для підвищення ефективності роботи користувачів система повинна підтримувати функції пошуку та фільтрації даних. Користувачі повинні мати можливість здійснювати пошук об'єктів за назвою, категорією, координатами або іншими параметрами. Також необхідно реалізувати фільтрацію інформації за шарами, типами об'єктів або часовими характеристиками.

Важливою функціональною вимогою є реалізація журналу подій та історії змін. Система повинна фіксувати дії користувачів, зберігати інформацію про створення, редагування та видалення об'єктів, а також забезпечувати можливість перегляду історії змін для контролю цілісності даних.

У системі також необхідно реалізувати механізм сповіщень та повідомлень. Користувачі повинні отримувати інформацію про появу нових подій, зміни стану об'єктів або інші важливі повідомлення. Подібна функція є особливо важливою для систем ситуаційної обізнаності та оперативного реагування (табл. 2.1).

Для забезпечення стабільної роботи програмного комплексу необхідно передбачити підтримку резервного копіювання даних та механізми відновлення інформації у разі збою системи. Крім цього, система повинна бути оптимізована для роботи із великими обсягами геопросторових даних та підтримувати можливість подальшого масштабування.

Таблиця 2.1– Функціональні вимоги до системи моніторингу

№	Функціональна вимога	Характеристика вимоги	Призначення
1	Авторизація користувачів	Реалізація входу до системи за логіном і паролем	Захист доступу до інформації
2	Розмежування ролей	Поділ користувачів на адміністраторів та операторів	Контроль доступу до функцій системи
3	Робота з цифровою картою	Відображення інтерактивної карти	Візуалізація геопросторових даних
4	Масштабування карти	Зміна масштабу та навігація по карті	Зручність роботи користувача
5	Робота із шарами	Увімкнення та вимкнення тематичних шарів	Організація та фільтрація інформації
6	Додавання міток	Створення користувацьких об'єктів на карті	Позначення місць або подій
7	Редагування об'єктів	Зміна координат, опису та категорій	Оновлення інформації
8	Видалення об'єктів	Можливість видалення застарілих даних	Актуалізація інформації
9	Робота з маршрутами	Побудова та відображення маршрутів	Аналіз переміщення об'єктів
10	Відображення полігонів	Побудова зон контролю або ризику	Аналіз територій
11	Прикріплення фото	Додавання мультимедійних матеріалів	Документування подій
12	Система коментарів	Додавання текстових пояснень	Деталізація інформації
13	Збереження геоданих	Робота із просторовою базою даних	Централізоване зберігання інформації
14	Пошук об'єктів	Пошук за назвою або координатами	Швидкий доступ до інформації
15	Фільтрація даних	Відбір об'єктів за параметрами	Спрощення аналізу інформації
16	Робота у реальному часі	Оперативне оновлення інформації	Підтримка ситуаційної обізнаності
17	Журнал подій	Фіксація дій користувачів	Контроль змін у системі
18	Система сповіщень	Надсилання повідомлень користувачам	Оперативне інформування
19	Резервне копіювання	Створення копій бази даних	Захист від втрати інформації
20	Кросплатформеність	Робота на ПК, планшетах і смартфонах	Доступність системи
21	Інтеграція API	Підключення зовнішніх сервісів	Розширення функціональності
22	Захист даних	Використання шифрування та HTTPS	Інформаційна безпека
23	Масштабованість	Підтримка збільшення кількості користувачів	Розвиток системи
24	Адаптивний інтерфейс	Підлаштування під різні екрани	Зручність використання
25	Адміністрування системи	Керування користувачами та налаштуваннями	Контроль роботи комплексу

Важливою вимогою є забезпечення кросплатформеності системи. Користувачі повинні мати можливість працювати із системою через різні операційні системи та пристрої, включаючи персональні комп'ютери, планшети та мобільні телефони. Це забезпечується використанням веборієнтованої архітектури та адаптивного інтерфейсу користувача.

2.2 Розробка структури системи та моделі взаємодії компонентів

Розробка структури апаратно-програмного комплексу моніторингу та ситуаційної обізнаності передбачає визначення основних модулів системи, їхнього функціонального призначення та порядку взаємодії між ними. Практична реалізація системи доцільно виконується за клієнт-серверною архітектурою, оскільки саме такий підхід забезпечує централізоване збереження даних, контроль доступу користувачів, обробку запитів на сервері та відображення результатів через вебінтерфейс.

У межах дипломного проєкту структура системи формується як сукупність взаємопов'язаних компонентів: клієнтського вебінтерфейсу, серверної частини, бази даних, картографічного модуля, модуля авторизації, модуля керування шарами, модуля обробки користувацьких об'єктів, модуля мультимедійних вкладень та модуля журналювання дій. Такий поділ дозволяє зробити систему зрозумілою для розробки, тестування та подальшого розширення.

Клієнтська частина системи призначена для взаємодії користувача з апаратно-програмним комплексом. Вона реалізується у вигляді вебінтерфейсу, через який користувач отримує доступ до карти, інформаційних шарів, міток, полігонів, маршрутів і карток об'єктів. Через інтерфейс користувач може виконувати вхід до системи, переглядати доступні йому шари, створювати нові об'єкти, редагувати їхній опис, додавати коментарі та прикріплювати фотографії. Практично клієнтська частина може бути реалізована за допомогою HTML, CSS,

JavaScript та картографічної бібліотеки Leaflet або OpenLayers.

Серверна частина є центральним елементом системи. Вона приймає запити від клієнтського інтерфейсу, перевіряє права доступу користувача, звертається до бази даних, обробляє отриману інформацію та повертає результат назад у вебінтерфейс. На серверному рівні реалізується основна бізнес-логіка системи: авторизація користувачів, перевірка ролей, створення об'єктів, оновлення шарів, збереження мультимедійних матеріалів і фіксація змін у журналі подій. Для практичної реалізації серверної частини доцільно використати Node.js з Express, Django або Flask.

База даних забезпечує централізоване збереження всієї інформації, необхідної для роботи системи. У ній зберігаються облікові записи користувачів, ролі доступу, опис інформаційних шарів, координати об'єктів, геометрія полігонів і маршрутів, текстові коментарі, посилання на прикріплені файли та журнал дій. Для такої системи доцільно використовувати PostgreSQL із розширенням PostGIS, оскільки воно підтримує просторові типи даних і дозволяє виконувати геопросторові запити.

Практична структура бази даних може включати таблиці users, roles, layers, objects, comments, attachments та action_log.

Таблиця users містить дані користувачів, таблиця roles визначає рівні доступу, таблиця layers відповідає за тематичні шари, таблиця objects зберігає просторові об'єкти, comments використовується для текстових пояснень, attachments містить інформацію про прикріплені файли, а action_log фіксує всі важливі дії користувачів у системі.

Картографічний модуль відповідає за відображення геопросторових даних на цифровій карті. Він отримує від сервера набір об'єктів у форматі JSON або GeoJSON і відображає їх у вигляді міток, ліній або полігонів.

При зміні масштабу карти або виборі певного шару клієнтська частина надсилає запит на сервер, після чого система повертає тільки ті об'єкти, які відповідають заданим параметрам. Це дозволяє зменшити навантаження на

інтерфейс і підвищити швидкодію системи.

Модель взаємодії компонентів будується за послідовним принципом. Спочатку користувач відкриває вебінтерфейс і проходить авторизацію. Сервер перевіряє введені облікові дані та визначає роль користувача. Після успішного входу клієнтська частина надсилає запит на отримання доступних інформаційних шарів.

Сервер звертається до бази даних, вибирає дозволені шари відповідно до ролі користувача та передає їх у вебінтерфейс.

Далі користувач обирає потрібний шар, а система завантажує відповідні об'єкти на карту (рис. 2.1).

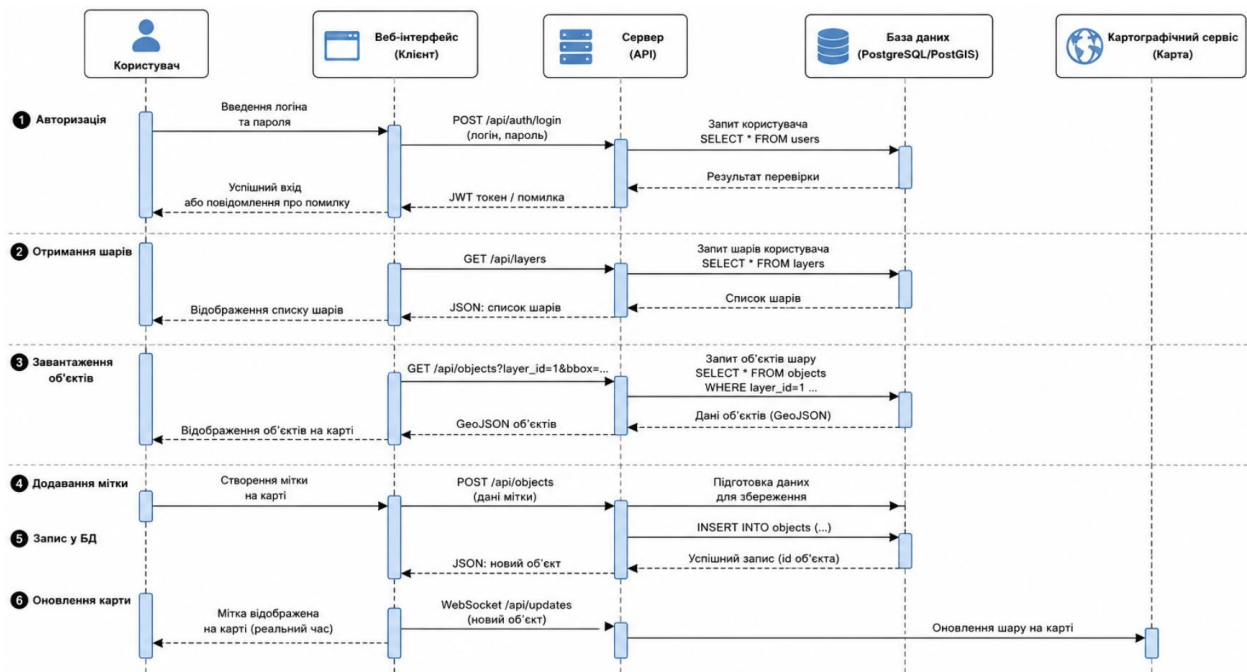


Рисунок 2.1 – UML-діаграма взаємодії компонентів системи моніторингу та ситуаційної обізнаності

Під час створення нового об'єкта користувач обирає місце на карті, вводить назву, опис, категорію та за потреби додає фото або коментар. Клієнтська частина формує запит і передає його на сервер. Сервер перевіряє права користувача, записує дані в базу, зберігає вкладення та створює запис у журналі подій. Після

цього оновлений об'єкт повертається на карту та стає доступним іншим користувачам, які мають відповідні права перегляду.

Для забезпечення роботи системи в режимі реального часу до структури доцільно включити модуль оновлення даних. Він може бути реалізований через WebSocket або періодичне оновлення запитів до сервера. У разі зміни об'єкта сервер надсилає повідомлення клієнтам, які працюють із відповідним шаром, і карта автоматично оновлюється без ручного перезавантаження сторінки.

Узагальнена модель взаємодії компонентів має такий вигляд: користувач взаємодіє з вебінтерфейсом, вебінтерфейс надсилає запити до API сервера, сервер перевіряє права доступу та обробляє логіку, база даних зберігає або повертає потрібні дані, картографічний модуль відображає результат на цифровій карті, а журнал подій фіксує виконані операції. Така модель забезпечує логічну послідовність роботи системи та дозволяє контролювати всі етапи обробки інформації.

2.3 Проєктування серверної частини комплексу

Серверна частина апаратно-програмного комплексу моніторингу та ситуаційної обізнаності є центральним елементом системи, оскільки саме вона забезпечує обробку запитів користувачів, керування доступом, збереження геопросторових даних, взаємодію з базою даних, передавання інформації до клієнтського інтерфейсу та підтримку оновлення даних у режимі реального часу. Якщо клієнтська частина відповідає переважно за зручне відображення карти та взаємодію користувача з інтерфейсом, то серверна частина виконує основну логіку роботи системи та забезпечує узгоджену роботу всіх компонентів комплексу.

У межах дипломного проєкту серверну частину доцільно проєктувати за принципом клієнт-серверної архітектури з використанням REST API. Такий підхід дозволяє відокремити інтерфейс користувача від логіки обробки даних, зробити

систему більш гнучкою, масштабованою та зручною для подальшого розвитку. Клієнтський вебінтерфейс у такій моделі надсилає запити до серверного API, сервер обробляє ці запити, звертається до бази даних, виконує перевірку прав доступу та повертає результат у форматі JSON або GeoJSON [16].

Основними завданнями серверної частини є приймання HTTP-запитів від клієнта, автентифікація користувачів, перевірка ролей, обробка просторових об'єктів, керування інформаційними шарами, збереження коментарів і мультимедійних вкладень, ведення журналу подій та передавання оновлень користувачам. Така структура дозволяє реалізувати не просто карту з мітками, а повноцінний програмний комплекс для моніторингу та ситуаційної обізнаності.

Для реалізації серверної частини може бути використано середовище Node.js із фреймворком Express.js. Такий варіант є доцільним, оскільки Node.js добре підходить для створення вебсервісів, підтримує асинхронну обробку запитів, легко інтегрується з базами даних та забезпечує можливість реалізації обміну даними в реальному часі через WebSocket. Альтернативно серверна частина може бути реалізована на Python з використанням Django або Flask, однак для даного проєкту Node.js є практичним рішенням завдяки простоті побудови API та зручній взаємодії з клієнтською частиною на JavaScript.

Серверна частина комплексу складається з декількох логічних модулів. Першим важливим модулем є модуль автентифікації та авторизації. Він відповідає за вхід користувачів до системи, перевірку логіна і пароля, формування токена доступу та визначення ролі користувача. Після успішної автентифікації користувач отримує токен, який надалі передається разом із запитами до сервера. Це дозволяє серверу визначати, хто саме виконує дію та чи має цей користувач право на доступ до певної інформації.

Другим важливим модулем є модуль керування користувачами та ролями. У системі можуть бути передбачені щонайменше три категорії користувачів: адміністратор, оператор і переглядач. Адміністратор має право створювати користувачів, змінювати їхні ролі, керувати шарами та переглядати журнал подій.

Оператор може створювати та редагувати об'єкти на карті, додавати коментарі й фото. Переглядач має право лише переглядати доступні йому шари та об'єкти без можливості внесення змін. Такий підхід забезпечує розмежування доступу та підвищує рівень інформаційної безпеки системи.

Окреме місце займає модуль роботи з інформаційними шарами. Він забезпечує створення, редагування, видалення та відображення тематичних шарів карти. Кожен шар може мати назву, опис, тип даних, колір відображення та перелік ролей або користувачів, яким дозволено доступ. Наприклад, один шар може містити об'єкти інфраструктури, інший — зони ризику, третій — точки спостереження або події. Сервер повинен повертати клієнту лише ті шари, до яких конкретний користувач має доступ.

Модуль обробки геопросторових об'єктів є одним із ключових у серверній частині. Він відповідає за створення, редагування, видалення та отримання об'єктів, які відображаються на цифровій карті. У разі використання PostgreSQL із розширенням PostGIS сервер може виконувати просторові запити, наприклад пошук об'єктів у межах певної області карти або визначення об'єктів поблизу заданої точки.

Для передавання геопросторових даних між сервером і клієнтом доцільно використовувати формат GeoJSON. Цей формат є зручним для роботи з картографічними бібліотеками, оскільки дозволяє описувати точки, лінії та полігони у стандартизованому вигляді. Клієнтська частина отримує від сервера набір об'єктів у форматі GeoJSON і відображає їх на карті за допомогою Leaflet або іншої картографічної бібліотеки. Важливим елементом серверної частини є модуль роботи з мультимедійними вкладеннями. Він забезпечує завантаження, збереження та прив'язку файлів до конкретних об'єктів на карті. Наприклад, користувач може створити мітку, додати до неї опис і прикріпити фотографію. Сервер приймає файл, перевіряє його тип і розмір, зберігає у файловому сховищі або хмарному середовищі, а в базі даних фіксує посилання на цей файл. Це дозволяє уникнути перевантаження бази даних великими файлами та забезпечити зручний доступ до

вкладень [17].

Модуль коментарів забезпечує можливість додавання текстових пояснень до об'єктів. Кожен коментар має бути пов'язаний із конкретним об'єктом, користувачем, датою створення та текстом повідомлення. Такий механізм дозволяє користувачам уточнювати інформацію, залишати службові примітки та вести історію обговорення певного об'єкта моніторингу. Для забезпечення контролю змін у системі необхідно реалізувати модуль журналювання. Він фіксує основні дії користувачів: вхід до системи, створення об'єкта, редагування даних, видалення запису, завантаження файлу, зміну ролей або доступу до шару. Журнал подій є важливим елементом безпеки, оскільки дозволяє відстежувати, хто і коли виконував певні дії в системі.

Серверна частина повинна також підтримувати механізм оновлення даних у режимі реального часу. Для цього може бути використано WebSocket. Якщо один користувач додає новий об'єкт або змінює інформацію на карті, сервер надсилає повідомлення іншим користувачам, які мають доступ до відповідного шару. У результаті карта оновлюється автоматично без перезавантаження сторінки. Це особливо важливо для систем ситуаційної обізнаності, де актуальність інформації має вирішальне значення.

Проектування серверного API передбачає визначення основних маршрутів, через які клієнтська частина буде взаємодіяти із сервером. Наприклад, для авторизації використовується маршрут POST /api/auth/login, для отримання шарів - GET /api/layers, для отримання об'єктів - GET /api/objects, для створення нового об'єкта - POST /api/objects, для редагування - PUT /api/objects/:id, для видалення - DELETE /api/objects/:id. Окремі маршрути створюються для коментарів, вкладень, користувачів та журналу подій (рис. 2.2).

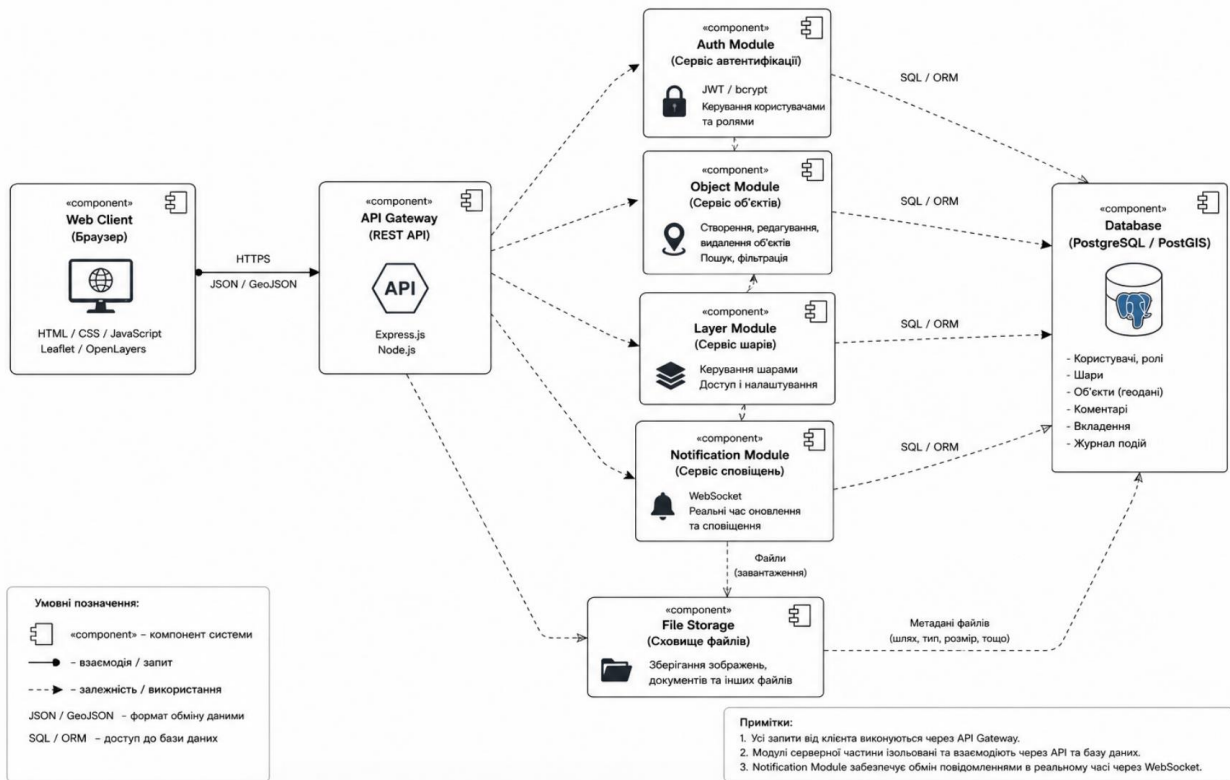


Рисунок 2.2 – Діаграма компонентів

З погляду безпеки серверна частина повинна використовувати захищене з'єднання HTTPS, хешування паролів, перевірку токенів доступу та обмеження прав користувачів. Паролі не повинні зберігатися у відкритому вигляді. Для їх захисту доцільно використовувати алгоритми хешування, наприклад bcrypt. Для автентифікації користувачів може застосовуватися JWT токен, який передається у заголовках запитів до сервера.

Для підвищення надійності серверна частина повинна передбачати обробку помилок. Якщо користувач вводить неправильні дані, не має доступу до певного шару або намагається виконати заборонену дію, сервер повинен повернути коректне повідомлення про помилку. Це дозволяє уникнути неконтрольованих збоїв та покращує зручність користування системою.

Апаратна складова апаратно-програмного комплексу моніторингу та ситуаційної обізнаності є основою функціонування всієї інформаційної системи та забезпечує виконання операцій зі збору, передачі, обробки, зберігання та відображення геопросторових даних. Від правильного вибору апаратних засобів

залежить продуктивність системи, її надійність, швидкодія та можливість масштабування відповідно до потреб користувачів [18].

До складу апаратної частини розробленого комплексу входять серверне обладнання, мережеві засоби передачі даних, клієнтські робочі станції користувачів, а також допоміжні засоби збору геопросторової інформації. Центральним елементом системи є сервер, на якому розміщується вебзастосунок, база даних та програмні модулі обробки інформації. Сервер забезпечує централізоване управління даними, виконання бізнес-логіки системи, зберігання інформації про користувачів, шари карти, географічні об'єкти та прикріплені мультимедійні матеріали.

Для забезпечення стабільної роботи програмного комплексу доцільним є використання сервера з багатоядерним процесором рівня Intel Xeon або AMD EPYC, оперативною пам'яттю обсягом не менше 8–16 ГБ та твердотільним накопичувачем SSD. Використання SSD-накопичувачів дозволяє значно підвищити швидкість обробки запитів до бази даних та скоротити час завантаження картографічної інформації. Операційною системою серверного обладнання може виступати Linux Ubuntu Server, яка забезпечує високу стабільність роботи, підтримку сучасних вебтехнологій та широкі можливості адміністрування.

Важливим елементом апаратної інфраструктури є мережева підсистема. Для забезпечення взаємодії між клієнтами та сервером використовуються мережеві маршрутизатори, комутатори та засоби захищеного доступу до мережі Інтернет. Передача даних між компонентами системи здійснюється через протоколи TCP/IP та HTTPS. Застосування захищених протоколів дозволяє мінімізувати ризики несанкціонованого доступу до інформації та забезпечує конфіденційність переданих даних [19].

Клієнтська частина апаратного комплексу представлена персональними комп'ютерами, ноутбуками, планшетами та мобільними пристроями користувачів. Основною вимогою до клієнтського обладнання є наявність сучасного веббраузера та доступу до мережі Інтернет. Оскільки система реалізована як веборієнтований

сервіс, користувачам не потрібно встановлювати спеціалізоване програмне забезпечення. Усі функції доступні через стандартний браузер, що значно спрощує експлуатацію системи та розширює коло потенційних користувачів.

Для реалізації функцій моніторингу можуть використовуватись додаткові технічні засоби збору інформації. До них належать GPS-приймачі, смартфони з підтримкою геолокації, цифрові фото- та відеокамери, безпілотні літальні апарати та інші пристрої, здатні передавати координати або мультимедійні дані. Отримана інформація може автоматично або вручну завантажуватися до системи та відображатися на інтерактивній карті.

З метою забезпечення безперервності роботи комплексу доцільним є використання засобів резервного копіювання та джерел безперебійного живлення. Наявність резервних копій бази даних дозволяє відновити працездатність системи у випадку технічних збоїв або втрати інформації. Використання джерел безперебійного живлення забезпечує коректне завершення роботи серверного обладнання та знижує ризик пошкодження даних під час аварійного відключення електроенергії.

Таким чином, апаратна складова розробленого комплексу формує технічну основу функціонування системи моніторингу та ситуаційної обізнаності, забезпечує стабільну роботу програмних модулів, підтримує багатокористувацький режим роботи та створює необхідні умови для подальшого розвитку і масштабування інформаційної системи.

2.4 Проєктування клієнтської вебплатформи

Проєктування клієнтської вебплатформи є важливим етапом розробки апаратно-програмного комплексу моніторингу та ситуаційної обізнаності, оскільки саме клієнтська частина забезпечує безпосередню взаємодію користувача із системою. Якщо серверна частина відповідає за збереження, обробку та захист

даних, то клієнтська вебплатформа виконує функцію візуального середовища, через яке користувач отримує доступ до цифрової карти, інформаційних шарів, об'єктів моніторингу, коментарів, вкладень та інструментів керування.

У межах дипломного проєкту клієнтську вебплатформу доцільно реалізувати як адаптивний вебзастосунок, що працює у браузері та не потребує встановлення додаткового програмного забезпечення на комп'ютер користувача. Такий підхід забезпечує доступність системи з різних пристроїв, зокрема персональних комп'ютерів, ноутбуків, планшетів і смартфонів. Практична реалізація клієнтської частини може виконуватися з використанням HTML, CSS, JavaScript, а також картографічної бібліотеки Leaflet або OpenLayers.

Основним елементом клієнтської вебплатформи є інтерактивна карта. Вона повинна займати центральну частину робочого екрана та забезпечувати користувачеві можливість масштабування, переміщення по території, вибору інформаційних шарів і перегляду об'єктів. На карті відображаються мітки, полігони, маршрути та інші геопросторові елементи, які надходять із серверної частини у форматі JSON або GeoJSON. Використання GeoJSON є доцільним, оскільки цей формат зручно обробляється вебкартографічними бібліотеками та дозволяє стандартизовано описувати просторові об'єкти [20].

Практична структура клієнтської вебплатформи може включати декілька основних зон інтерфейсу. Верхня панель призначається для відображення назви системи, інформації про поточного користувача, кнопок виходу із системи та доступу до налаштувань. Ліва бічна панель використовується для роботи з інформаційними шарами, фільтрами та категоріями об'єктів. Центральна частина екрана відводиться під карту. Права панель або спливаюче вікно може використовуватися для перегляду детальної інформації про вибраний об'єкт, додавання коментарів, перегляду фотографій та редагування атрибутів.

Першим практичним елементом клієнтської частини є сторінка авторизації. Вона повинна містити форму введення логіна та пароля, кнопку входу та механізм виведення повідомлень про помилки. Після введення облікових даних клієнтська

частина надсилає POST-запит на серверний маршрут авторизації. У разі успішної перевірки сервер повертає токен доступу, який зберігається на стороні клієнта та використовується для подальших запитів до API. Якщо користувач не авторизований, доступ до карти та службових функцій має бути заблокований.

Після входу користувач потрапляє на основну сторінку системи. На цьому етапі клієнтська вебплатформа виконує запит до сервера для отримання списку доступних інформаційних шарів. Сервер повертає лише ті шари, до яких користувач має право доступу відповідно до його ролі. Отримані шари відображаються у бічній панелі у вигляді списку з перемикачами. Користувач може вмикати або вимикати окремі шари, завдяки чому карта не перевантажується зайвою інформацією.

Для завантаження об'єктів певного шару клієнтська частина надсилає GET-запит до серверного API із зазначенням ідентифікатора шару та меж поточної області карти. Такий підхід дозволяє завантажувати не всі об'єкти одразу, а лише ті, які знаходяться у межах видимої частини карти. Це підвищує швидкодію системи та зменшує навантаження на браузер користувача. Отримані об'єкти у форматі GeoJSON відображаються на карті як мітки, лінії або полігони.

Особливу увагу під час проєктування клієнтської вебплатформи необхідно приділити механізму додавання нових об'єктів. Користувач із відповідними правами повинен мати можливість натиснути кнопку “Додати об'єкт”, вибрати точку на карті та заповнити форму з основними параметрами. До таких параметрів належать назва об'єкта, опис, тип, шар, статус, координати та додаткові примітки. Після заповнення форми клієнтська частина формує POST-запит до сервера та передає дані для збереження у базі даних [21].

Для підвищення практичної цінності системи необхідно реалізувати можливість прикріплення фотографій до об'єктів. На рівні інтерфейсу це може бути реалізовано через поле завантаження файлу у картці об'єкта. Після вибору фото клієнтська частина формує multipart/form-data запит і передає файл на сервер. Після успішного збереження файл відображається у картці об'єкта у вигляді

мініатюри або посилання для перегляду.

Картка об'єкта є одним із головних елементів клієнтської вебплатформи. Вона повинна відкриватися після натискання на мітку або інший елемент карти. У картці відображаються назва об'єкта, опис, координати, дата створення, автор, належність до шару, статус, коментарі та прикріплені файли. За наявності прав редагування користувач може змінити дані об'єкта або видалити його. Якщо користувач має лише право перегляду, кнопки редагування та видалення не повинні відображатися.

Клієнтська вебплатформа також повинна підтримувати пошук і фільтрацію. Пошук може здійснюватися за назвою об'єкта, типом, шаром або датою створення. Фільтрація дозволяє швидко відібрати потрібну інформацію серед великої кількості об'єктів. Наприклад, користувач може відобразити лише об'єкти певної категорії або лише ті, що були створені протягом визначеного періоду. Це особливо важливо для систем моніторингу, де кількість даних може швидко зростати.

Для підтримки ситуаційної обізнаності необхідно передбачити автоматичне оновлення карти. Практично це може бути реалізовано через WebSocket або періодичне опитування сервера. У разі додавання нового об'єкта або зміни існуючого сервер надсилає повідомлення клієнтській частині, після чого карта оновлюється без перезавантаження сторінки. Такий механізм дозволяє користувачам бачити актуальну інформацію майже одразу після її внесення до системи.

Важливим аспектом є адаптивність інтерфейсу. Оскільки система може використовуватися не лише на робочих станціях, а й на мобільних пристроях, інтерфейс повинен коректно відображатися на різних розмірах екрана. На великих екранах доцільно використовувати повноцінні бічні панелі, а на мобільних пристроях ці панелі можуть відкриватися у вигляді висувного меню. Карта при цьому повинна залишатися основним робочим простором [22].

Практична логіка роботи клієнтської вебплатформи може бути описана так: користувач відкриває вебсторінку, проходить авторизацію, отримує список

доступних шарів, обирає необхідні шари, система завантажує відповідні об'єкти з сервера, користувач переглядає карту, додає або редагує об'єкти, а всі зміни передаються на сервер і після збереження відображаються іншим користувачам. Такий сценарій забезпечує повний цикл роботи із геопросторовими даними.

З технічної точки зору клієнтська частина повинна містити окремі програмні модулі. До них належать модуль авторизації, модуль роботи з картою, модуль роботи з шарами, модуль об'єктів, модуль фільтрації, модуль коментарів, модуль завантаження файлів та модуль реального часу. Така структура спрощує супровід коду та дозволяє надалі розширювати систему без повної переробки інтерфейсу.

2.5 Розробка системи авторизації та розмежування доступу

Розробка системи авторизації та розмежування доступу є одним із ключових етапів створення апаратно-програмного комплексу моніторингу та ситуаційної обізнаності. Оскільки система передбачає роботу із геопросторовими даними, інформаційними шарами, користувацькими об'єктами, коментарями та мультимедійними вкладеннями, доступ до цих даних не може бути відкритим для всіх користувачів. Для забезпечення безпечної роботи комплексу необхідно реалізувати механізм перевірки користувача, визначення його ролі та обмеження доступу до функцій відповідно до наданих прав.

У межах дипломного проєкту система авторизації проєктується як окремий функціональний модуль серверної частини. Його основним призначенням є перевірка облікових даних користувача, формування токена доступу, контроль активної сесії та перевірка прав користувача під час виконання кожного запиту до серверного API. Такий підхід дозволяє забезпечити закритий доступ до системи та унеможливити перегляд або зміну даних неавторизованими особами.

Практично систему авторизації доцільно реалізувати на основі логіна, пароля та JWT токена. Користувач вводить логін і пароль у формі входу на клієнтській

вебплатформі. Після цього клієнтська частина надсилає запит на серверний маршрут авторизації. Сервер перевіряє наявність користувача в базі даних, порівнює введений пароль із хешованим паролем, що зберігається у системі, і у разі успішної перевірки формує токен доступу. Цей токен надалі використовується під час усіх запитів до API.

Паролі користувачів не повинні зберігатися у відкритому вигляді. Для їх захисту доцільно використовувати хешування за допомогою алгоритму bcrypt. Під час створення нового користувача пароль перетворюється на хеш, після чого саме цей хеш записується до бази даних. Під час входу до системи сервер не розшифровує пароль, а лише порівнює введені значення з наявним хешем. Це дозволяє зменшити ризики компрометації облікових даних у разі несанкціонованого доступу до бази даних.

Для практичної реалізації системи доступу в базі даних необхідно передбачити таблиці користувачів, ролей і прав доступу. Таблиця `users` зберігає інформацію про користувачів системи, таблиця `roles` містить перелік ролей, а таблиця `permissions` або `role_permissions` визначає, які дії дозволені кожній ролі. Така структура дозволяє гнучко змінювати права користувачів без внесення змін до програмного коду.

У системі доцільно передбачити три основні ролі: адміністратор, оператор і переглядач. Адміністратор має повний доступ до функцій системи, включаючи керування користувачами, створення шарів, зміну прав доступу, перегляд журналу подій і редагування будь-яких об'єктів. Оператор має право працювати з об'єктами на карті, створювати нові мітки, редагувати доступні йому об'єкти, додавати коментарі та прикріплювати фотографії. Переглядач має обмежений доступ і може лише переглядати дозволені шари та об'єкти без можливості їх редагування.

Практичне розмежування доступу повинно виконуватися не лише на рівні інтерфейсу, а й на рівні серверного API. Наприклад, якщо користувач із роллю переглядача не бачить кнопку “Видалити об'єкт” у вебінтерфейсі, це ще не гарантує безпеку. Такий користувач може спробувати вручну надіслати запит до

API. Тому сервер обов'язково повинен перевіряти роль користувача перед виконанням кожної критичної операції. Якщо прав недостатньо, сервер повертає відповідь із помилкою доступу.

Для цього у серверній частині реалізується middleware перевірки токена та ролі. Middleware є проміжним програмним шаром, який виконується перед основною логікою маршруту. Спочатку він перевіряє наявність JWT токена у заголовку запиту. Далі виконується перевірка дійсності токена. Якщо токен коректний, сервер отримує з нього ідентифікатор користувача та роль. Після цього перевіряється, чи має ця роль право виконувати відповідну дію. Лише після успішної перевірки запит передається до основного контролера.

Практична логіка роботи системи авторизації має такий вигляд. Користувач відкриває сторінку входу, вводить логін і пароль, клієнтська частина передає дані на сервер, сервер перевіряє обліковий запис у базі даних, порівнює пароль із хешем, формує JWT токен і повертає його клієнту. Після цього клієнт зберігає токен і додає його до кожного наступного запиту. Якщо токен відсутній, прострочений або підроблений, сервер блокує доступ до ресурсу.

Для зберігання токена на клієнтському боці можна використовувати httpOnly cookie або захищене локальне сховище. Більш безпечним підходом є використання httpOnly cookie, оскільки такий токен недоступний для JavaScript і краще захищений від XSS атак. Однак у навчальному проєкті може бути використане локальне сховище браузера, якщо додатково передбачити захист від небезпечного введення та коректну обробку сесії [23].

Окрему увагу слід приділити розмежуванню доступу до інформаційних шарів. У системі моніторингу різні групи користувачів можуть мати доступ до різних наборів даних. Наприклад, один користувач може бачити лише загальні об'єкти інфраструктури, інший має доступ до службових шарів, а адміністратор бачить усю інформацію. Для цього у базі даних необхідно передбачити таблицю `layer_access`, яка пов'язує шари з ролями або конкретними користувачами.

Під час запиту списку шарів сервер не повертає всі наявні шари, а виконує

вибірку лише тих, які дозволені для поточного користувача. Аналогічно під час запиту об'єктів сервер перевіряє, чи має користувач доступ до шару, у якому розташований об'єкт. Це дозволяє реалізувати не лише загальне розмежування ролей, а й більш детальний контроль доступу до конкретних даних.

Практично права доступу можна поділити на такі дії: перегляд об'єктів, створення об'єктів, редагування об'єктів, видалення об'єктів, керування шарами, додавання коментарів, завантаження файлів, перегляд журналу подій і керування користувачами. Кожна роль отримує набір дозволених дій. Наприклад, переглядач має право лише на перегляд, оператор має право створювати й редагувати об'єкти, а адміністратор має повний набір прав.

Приклад практичного розподілу прав доступу можна описати так: адміністратор має доступ до всіх модулів системи, оператор має доступ до карти, шарів, об'єктів, коментарів і вкладень, а переглядач має доступ тільки до карти та перегляду об'єктів. Така модель є зрозумілою, зручною для реалізації та достатньою для дипломного проєкту.

Під час реалізації авторизації також необхідно передбачити обробку помилок. Якщо користувач вводить неправильний пароль, система повинна повернути повідомлення про помилку входу. Якщо обліковий запис заблокований, сервер повинен заборонити доступ. Якщо користувач намагається виконати дію без відповідних прав, система повинна повернути статус 403 Forbidden. Якщо токен недійсний або відсутній, сервер повертає статус 401 Unauthorized.

Для підвищення безпеки доцільно реалізувати журналювання дій користувачів. У журналі повинні фіксуватися спроби входу до системи, успішні авторизації, помилки входу, створення та редагування об'єктів, зміна ролей, видалення даних і завантаження файлів. Це дозволяє адміністратору контролювати активність користувачів і виявляти потенційно небезпечні дії.

З технічної точки зору серверна частина може містити окремі маршрути для авторизації та керування доступом. Наприклад, `POST /api/auth/login` використовується для входу до системи, `POST /api/auth/logout` - для виходу, `GET`

/api/auth/me - для отримання інформації про поточного користувача, GET /api/users - для перегляду користувачів адміністратором, POST /api/users - для створення нового користувача, PUT /api/users/:id/role - для зміни ролі користувача.

У клієнтській частині система доступу проявляється через динамічне відображення елементів інтерфейсу. Якщо користувач не має права створювати об'єкти, кнопка додавання мітки не відображається. Якщо він не має права редагування, відповідні кнопки у картці об'єкта приховуються. Якщо користувач не має доступу до певного шару, цей шар не показується у списку. Проте остаточна перевірка завжди повинна виконуватися на сервері (табл. 2.2).

Таблиця 2.2 – Розмежування доступу користувачів системи

№	Роль користувача	Доступ до карти	Робота із шарами	Створення об'єктів	Редагування об'єктів	Видалення об'єктів	Коментарі та фото	Керування користувачами / журнал
1	Адміністратор	Повний доступ	Створення, редагування, видалення	Дозволено	Дозволено	Дозволено	Дозволено	Повний доступ
2	Оператор	Повний доступ	Перегляд доступних шарів	Дозволено	Дозволено	Обмежено	Дозволено	Частковий доступ
3	Переглядач	Перегляд карти	Перегляд доступних шарів	Заборонено	Заборонено	Заборонено	Лише перегляд	Заборонено
4	Гість	Обмежений перегляд	Лише публічні шари	Заборонено	Заборонено	Заборонено	Заборонено	Заборонено

Для реалізації захищеного доступу до файлів також необхідно перевіряти

права користувача перед відкриттям вкладень. Якщо фотографія прикріплена до об'єкта службового шару, її повинен бачити лише користувач, який має доступ до цього шару. Тому сервер не повинен віддавати прямі відкриті посилання на всі файли без перевірки прав доступу.

2.6 Реалізація механізму створення та керування шарами карти

Однією з ключових функцій апаратно-програмного комплексу моніторингу та ситуаційної обізнаності є підтримка роботи з інформаційними шарами карти. Саме шарова структура дозволяє організувати геопросторові дані за тематичними категоріями, забезпечити зручну візуалізацію об'єктів та реалізувати гнучке керування доступом користувачів до інформації. У межах дипломного проєкту механізм створення та керування шарами реалізується як окремий функціональний модуль системи, який забезпечує створення, редагування, видалення, відображення та фільтрацію тематичних шарів карти.

Практично інформаційний шар являє собою окремий набір геопросторових об'єктів, які мають спільне призначення або належать до однієї категорії даних. Наприклад, у системі можуть існувати шари “Інфраструктура”, “Маршрути”, “Зони ризику”, “Пункти спостереження”, “Події”, “Технічні об'єкти” або “Користувацькі мітки”. Кожен шар містить набір об'єктів, які відображаються на карті лише за умови його активації користувачем.

Реалізація механізму роботи із шарами починається зі створення структури бази даних. Для цього у системі передбачається таблиця `layers`, у якій зберігається основна інформація про шари карти. До структури таблиці входять ідентифікатор шару, назва, опис, тип шару, колір відображення, статус активності, дата створення та ідентифікатор користувача, який створив шар.

Практична структура таблиці `layers` може бути подана у такому вигляді:

- `id` — унікальний ідентифікатор шару;

- name — назва шару;
- description — опис шару;
- color — колір відображення об'єктів;
- type — тип шару;
- visibility — статус видимості;
- created_by — автор шару;
- created_at — дата створення.

Кожен геопросторовий об'єкт у таблиці objects повинен містити посилання на шар, до якого він належить. Для цього використовується зовнішній ключ layer_id. Такий підхід дозволяє швидко виконувати фільтрацію об'єктів за шарами та керувати їх відображенням на карті.

На серверному рівні реалізується окремий модуль Layer Service, який відповідає за роботу із шарами карти. Даний модуль забезпечує створення нового шару, отримання списку доступних шарів, оновлення параметрів шару, видалення шару та перевірку прав доступу користувача. Для взаємодії із клієнтською частиною використовується REST API.

Практично серверна частина повинна підтримувати такі API маршрути: GET /api/layers, POST /api/layers, PUT /api/layers/:id, DELETE /api/layers/:id.

Маршрут GET /api/layers використовується для отримання списку доступних шарів. Сервер перевіряє роль користувача та повертає лише ті шари, до яких він має доступ. Маршрут POST /api/layers використовується для створення нового шару. Право створення шарів повинні мати лише адміністратори або оператори із відповідними правами.

Під час створення нового шару користувач через вебінтерфейс вводить назву шару, опис, вибирає колір та тип відображення. Після цього клієнтська частина надсилає POST-запит на сервер. Сервер перевіряє права користувача, виконує валідацію введених даних та записує новий шар у базу даних.

Для редагування шару використовується маршрут PUT /api/layers/:id.

Користувач може змінити назву, опис, колір або статус видимості шару. Сервер перевіряє, чи має користувач право редагування, після чого оновлює відповідний запис у базі даних.

Видалення шару виконується через маршрут `DELETE /api/layers/:id`. Практично під час видалення шару система повинна перевіряти, чи існують об'єкти, пов'язані із цим шаром. Якщо шар містить об'єкти, система може або забороняти видалення, або виконувати каскадне видалення всіх пов'язаних даних. Для дипломного проєкту доцільніше реалізувати захист від випадкового видалення та вимагати попереднього очищення шару.

На клієнтському рівні шари відображаються у бічній панелі вебінтерфейсу. Кожен шар подається у вигляді окремого елемента списку з перемикачем видимості. Якщо користувач активує шар, клієнтська частина надсилає запит до сервера для отримання об'єктів, що належать до цього шару. Якщо шар вимикається, відповідні об'єкти видаляються з карти без перезавантаження сторінки. Для практичної реалізації механізму відображення шарів доцільно використовувати окремі `LayerGroup` у бібліотеці `Leaflet`. Кожен шар карти відповідає окремій групі об'єктів. При активації шару система додає `LayerGroup` на карту, а при деактивації видаляє його. Такий підхід дозволяє ефективно керувати великою кількістю об'єктів та підтримувати високу швидкодію вебінтерфейсу.

Особливу увагу необхідно приділити розмежуванню доступу до шарів. У системі можуть існувати відкриті та службові шари. Наприклад, оператори можуть бачити шари технічної інфраструктури, а переглядачі мають доступ лише до загальної інформації. Для реалізації цього механізму використовується таблиця `layer_access`, яка визначає відповідність між ролями користувачів та шарами. Практично механізм перевірки доступу працює таким чином: під час отримання списку шарів сервер виконує SQL-запит із перевіркою ролі користувача. Якщо користувач не має права доступу до певного шару, сервер не повертає цей шар у відповіді API. Таким чином приховується не лише сам шар, а й усі об'єкти, які до нього належать [9].

Для забезпечення швидкодії системи необхідно реалізувати механізм динамічного завантаження об'єктів. Якщо користувач переміщується по карті або змінює масштаб, клієнтська частина передає серверу межі видимої області карти. Сервер повертає лише ті об'єкти, які знаходяться в межах цієї області та належать до активних шарів. Це значно зменшує обсяг переданих даних та підвищує швидкість роботи системи.

У системі також необхідно реалізувати механізм стилізації шарів. Кожен шар може мати власний колір, тип маркера або стиль полігону. Наприклад, шари зон ризику можуть відображатися червоним кольором, маршрути — синім, а інфраструктурні об'єкти — зеленим. Інформація про стиль шару зберігається у базі даних і передається клієнтській частині разом із даними шару. Для підтримки ситуаційної обізнаності система повинна забезпечувати оновлення шарів у режимі реального часу. Якщо один користувач створює новий об'єкт у певному шарі, сервер через WebSocket повідомляє інших користувачів про зміну. Клієнтська частина автоматично оновлює відповідний LayerGroup та відображає новий об'єкт на карті.

Практично робота механізму шарів має такий вигляд: користувач входить до системи, сервер повертає список доступних шарів, користувач активує потрібні шари, клієнтська частина надсилає запит на отримання об'єктів, сервер повертає GeoJSON дані, а карта відображає їх у відповідних LayerGroup. Якщо користувач створює або редагує об'єкт, зміни зберігаються у базі даних та автоматично оновлюються для інших користувачів.

Для адміністрування шарів у вебінтерфейсі доцільно реалізувати окрему панель керування. У цій панелі адміністратор може створювати нові шари, редагувати їх параметри, змінювати права доступу та переглядати кількість об'єктів у кожному шарі. Це спрощує керування системою та забезпечує централізований контроль структури даних.

2.7 Реалізація механізму додавання та редагування об'єктів на карті

Однією з основних функцій апаратно-програмного комплексу моніторингу та ситуаційної обізнаності є можливість створення, редагування та видалення геопросторових об'єктів безпосередньо на цифровій карті. Саме механізм роботи з об'єктами забезпечує наповнення системи актуальною інформацією, підтримку ситуаційної обізнаності користувачів та формування інтерактивного інформаційного середовища. Практична реалізація даного механізму передбачає створення клієнтського інтерфейсу для роботи з картою, реалізацію серверних API маршрутів, інтеграцію з базою даних та підтримку оновлення інформації у режимі реального часу.

У межах дипломного проєкту об'єктом вважається будь-який елемент, який має координатну прив'язку та може бути відображений на карті. До таких об'єктів належать мітки, маршрути, полігони, точки спостереження, інфраструктурні елементи, зони ризику або користувацькі позначки. Кожен об'єкт повинен містити набір атрибутів, що описують його характеристики та дозволяють виконувати подальший аналіз інформації.

Для практичної реалізації механізму роботи з об'єктами необхідно створити структуру бази даних, яка забезпечуватиме збереження геопросторових даних та супровідної інформації. Основною таблицею є таблиця `objects`. У ній зберігаються ідентифікатор об'єкта, назва, опис, тип геометрії, координати, шар, дата створення, автор, статус та додаткові параметри.

Практична структура таблиці `objects` може містити такі поля:

- `id` — унікальний ідентифікатор об'єкта;
- `name` — назва об'єкта;
- `description` — текстовий опис;
- `geometry` — геометричні координати об'єкта;
- `object_type` — тип об'єкта;

- layer_id — належність до шару;
- status — поточний стан об'єкта;
- created_by — автор створення;
- created_at — дата створення;
- updated_at — дата редагування.

Для роботи з геометрією об'єктів доцільно використовувати PostgreSQL із розширенням PostGIS. Це дозволяє зберігати точки, лінії та полігони у спеціалізованому просторовому форматі та виконувати просторові запити до бази даних.

На клієнтському рівні механізм додавання об'єктів реалізується через інтерактивну взаємодію користувача із картою. Після натискання кнопки “Додати об'єкт” користувач обирає тип об'єкта: точка, маршрут або полігон. Далі користувач натискає на карту та задає координати нового елемента. Після завершення побудови система відкриває форму введення даних [29].

У формі створення об'єкта користувач вводить назву, опис, вибирає шар, категорію та статус об'єкта. За необхідності користувач може прикріпити фотографії або залишити коментар. Після підтвердження введених даних клієнтська частина формує POST-запит до серверного API та передає інформацію у форматі JSON або GeoJSON.

Практично серверна частина повинна підтримувати такі маршрути API: GET /api/objects, POST /api/objects, PUT /api/objects/:id, DELETE /api/objects/:id.

Маршрут POST /api/objects використовується для створення нового об'єкта. Сервер перевіряє токен користувача, визначає його роль, виконує валідацію вхідних даних та записує новий об'єкт у базу даних. Якщо дані коректні, сервер повертає створений об'єкт клієнтській частині, після чого він відображається на карті.

Для відображення об'єктів у вебінтерфейсі доцільно використовувати GeoJSON. Кожен об'єкт передається у вигляді Feature із геометрією та набором

властивостей. Наприклад, точковий об'єкт містить координати та інформацію про назву, тип, шар і статус. Картографічна бібліотека Leaflet отримує GeoJSON дані та автоматично відображає їх на карті [22].

Практична логіка створення об'єкта має такий вигляд: користувач натискає кнопку створення, обирає місце на карті, заповнює форму, клієнтська частина надсилає POST-запит, сервер перевіряє права доступу та зберігає дані, після чого об'єкт автоматично з'являється на карті для всіх користувачів із доступом до відповідного шару.

Редагування об'єктів є не менш важливою функцією системи. Для цього користувач обирає вже існуючий об'єкт на карті та відкриває його картку. Якщо користувач має право редагування, у картці відображається кнопка “Редагувати”. Після натискання система переводить об'єкт у режим редагування. У режимі редагування користувач може змінювати назву, опис, статус, координати або геометрію об'єкта. Для точкових об'єктів дозволяється переміщення мітки на карті. Для маршрутів і полігонів реалізується можливість редагування вершин геометрії. Після завершення редагування клієнтська частина надсилає PUT-запит до сервера із новими даними.

Маршрут `PUT /api/objects/:id` використовується для оновлення інформації про об'єкт. Сервер перевіряє права користувача, виконує валідацію нових параметрів та оновлює запис у базі даних. Дата останнього редагування автоматично змінюється у полі `updated_at`.

Особливу увагу необхідно приділити контролю доступу під час редагування об'єктів. Наприклад, оператор може редагувати лише ті об'єкти, які належать до дозволених йому шарів. Переглядач не повинен мати доступу до функції редагування. Усі перевірки виконуються на серверному рівні, навіть якщо клієнтський інтерфейс приховує кнопки редагування.

Практично механізм видалення об'єкта працює аналогічно. Після натискання кнопки “Видалити” система запитує підтвердження дії. Якщо користувач підтверджує видалення, клієнтська частина надсилає DELETE-запит до сервера.

Сервер перевіряє права доступу та видаляє об'єкт із бази даних. Для забезпечення стабільності системи необхідно реалізувати механізм журналювання змін. Після створення, редагування або видалення об'єкта сервер записує інформацію про виконану дію у таблицю журналу подій. У журналі зберігаються ідентифікатор користувача, тип дії, час виконання та ідентифікатор об'єкта. Це дозволяє контролювати зміни у системі та відновлювати історію роботи користувачів.

Для підтримки ситуаційної обізнаності система повинна забезпечувати оновлення об'єктів у режимі реального часу. Якщо один користувач створює або редагує об'єкт, сервер через WebSocket надсилає повідомлення іншим клієнтам. Клієнтська частина автоматично оновлює карту без необхідності перезавантаження сторінки.

Практично процес оновлення відбувається таким чином: сервер після зміни об'єкта формує повідомлення про оновлення, надсилає його через WebSocket усім клієнтам, які працюють із відповідним шаром, після чого клієнтська частина повторно завантажує оновлений об'єкт та відображає його на карті. Для підвищення зручності користувачів доцільно реалізувати систему кольорового позначення об'єктів. Наприклад, активні об'єкти можуть відображатися зеленим кольором, проблемні - червоним, а архівні - сірим. Інформація про стиль відображення може визначатися типом або статусом об'єкта.

Картка об'єкта повинна підтримувати перегляд прикріплених файлів і коментарів. Користувач може відкривати фотографії, додавати текстові пояснення та переглядати історію змін. Це робить систему не просто картою, а повноцінним середовищем документування та аналізу інформації. У практичній реалізації клієнтської частини механізм роботи з об'єктами доцільно поділити на окремі модулі: модуль створення об'єктів, модуль редагування геометрії, модуль картки об'єкта, модуль коментарів, модуль вкладень та модуль оновлення у реальному часі. Така структура спрощує підтримку коду та подальший розвиток системи.

2.8 Проєктування структури бази даних

Проєктування структури бази даних є одним із ключових етапів створення апаратно-програмного комплексу моніторингу та ситуаційної обізнаності, оскільки саме база даних забезпечує централізоване зберігання всієї інформації, яка використовується системою. До такої інформації належать облікові записи користувачів, ролі доступу, тематичні шари карти, геопросторові об'єкти, коментарі, мультимедійні вкладення, журнали подій та службові параметри системи.

У межах дипломного проєкту базу даних доцільно проєктувати як реляційну структуру з підтримкою просторових даних. Найбільш придатним рішенням для цього є PostgreSQL із розширенням PostGIS. Використання PostGIS дозволяє зберігати геометричні об'єкти, виконувати просторові запити, працювати з координатами, полігонами, маршрутами та точковими мітками. Це особливо важливо для системи, основним елементом якої є інтерактивна карта.

Структура бази даних повинна забезпечувати логічний зв'язок між користувачами, їхніми ролями, інформаційними шарами та об'єктами на карті. Кожен користувач має певну роль, кожен шар містить набір геопросторових об'єктів, а кожен об'єкт може мати коментарі та прикріплені файли. Така модель дозволяє підтримувати багатокористувацький режим роботи, розмежування доступу та збереження історії змін у системі. Основними таблицями бази даних є `users`, `roles`, `layers`, `layer_access`, `objects`, `comments`, `attachments` та `action_log`. Таблиця `users` призначена для зберігання облікових записів користувачів. Вона містить ідентифікатор користувача, логін, хеш пароля, електронну пошту, роль, дату створення та статус активності. Збереження саме хешу пароля, а не відкритого пароля, є обов'язковою умовою безпечної роботи системи.

Таблиця `roles` містить перелік ролей користувачів. У межах системи можуть бути передбачені ролі адміністратора, оператора, переглядача та гостя. Кожна роль визначає рівень доступу до функцій системи. Наприклад, адміністратор має повний

доступ до керування користувачами, шарами та журналом подій, оператор може створювати й редагувати об'єкти, переглядач має лише право перегляду інформації. Таблиця `layers` використовується для зберігання тематичних шарів карти. Кожен шар має власну назву, опис, тип, колір відображення, статус видимості, автора створення та дату створення. Наявність окремої таблиці шарів дозволяє структурувати просторові дані та вмикати або вимикати окремі категорії інформації у клієнтському інтерфейсі.

Для реалізації розмежування доступу до шарів передбачається таблиця `layer_access`. Вона встановлює зв'язок між ролями користувачів і шарами карти. Завдяки цій таблиці сервер може визначити, які шари доступні конкретному користувачеві залежно від його ролі. Це дозволяє не лише приховувати певні шари в інтерфейсі, а й блокувати доступ до них на рівні серверного API.

Центральною таблицею бази даних є `objects`. Вона зберігає інформацію про всі геопросторові об'єкти, які відображаються на карті. Кожен об'єкт має унікальний ідентифікатор, назву, опис, тип геометрії, координати, належність до шару, статус, автора створення, дату створення та дату останнього редагування. Поле `geometry` доцільно реалізувати як просторовий тип даних PostGIS, що дозволяє зберігати точки, лінії та полігони. Таблиця `comments` призначена для зберігання текстових пояснень до об'єктів. Кожен коментар пов'язується з конкретним об'єктом і користувачем, який його створив. Це дозволяє вести обговорення або уточнення інформації щодо певного об'єкта моніторингу.

Таблиця `attachments` використовується для зберігання інформації про мультимедійні вкладення. У ній фіксуються назва файлу, шлях до файлу, тип файлу, розмір, ідентифікатор об'єкта та користувач, який завантажив файл. Самі файли доцільно зберігати не безпосередньо у базі даних, а у файловому сховищі або хмарному середовищі, тоді як у таблиці `attachments` зберігається лише шлях або посилання на файл. Таблиця `action_log` забезпечує журналювання дій користувачів. У ній зберігаються відомості про створення, редагування, видалення об'єктів, вхід до системи, зміну прав доступу, завантаження файлів та інші важливі події. Журнал

подій дозволяє контролювати активність користувачів і підвищує рівень безпеки системи.

Логічна модель бази даних передбачає такі основні зв'язки: один користувач має одну роль, одна роль може належати багатьом користувачам, один шар може містити багато об'єктів, один об'єкт може мати багато коментарів і вкладень, одна роль може мати доступ до багатьох шарів, а один шар може бути доступним для декількох ролей. Така структура відповідає принципам нормалізації баз даних і дозволяє уникнути дублювання інформації. У практичній реалізації важливо передбачити використання первинних і зовнішніх ключів. Поле `id` у кожній таблиці виступає первинним ключем. Поле `role_id` у таблиці `users` є зовнішнім ключем до таблиці `roles`. Поле `layer_id` у таблиці `objects` є зовнішнім ключем до таблиці `layers`. Поля `object_id` у таблицях `comments` і `attachments` пов'язують коментарі та вкладення з конкретним об'єктом.

Для підвищення швидкодії системи необхідно використовувати індекси. Особливо важливим є створення просторового індексу для поля `geometry` у таблиці `objects`. Це дозволить швидко виконувати пошук об'єктів у межах певної області карти, визначати об'єкти поблизу заданої точки та фільтрувати інформацію за координатами. Також доцільно створити індекси для полів `layer_id`, `user_id`, `role_id` та `created_at`.

Практична структура бази даних може бути подана таким чином: `users` зберігає користувачів, `roles` описує ролі, `layers` містить тематичні шари, `layer_access` відповідає за права доступу до шарів, `objects` зберігає геопросторові об'єкти, `comments` містить коментарі, `attachments` зберігає інформацію про файли, а `action_log` фіксує всі важливі дії у системі (рис. 2.3).

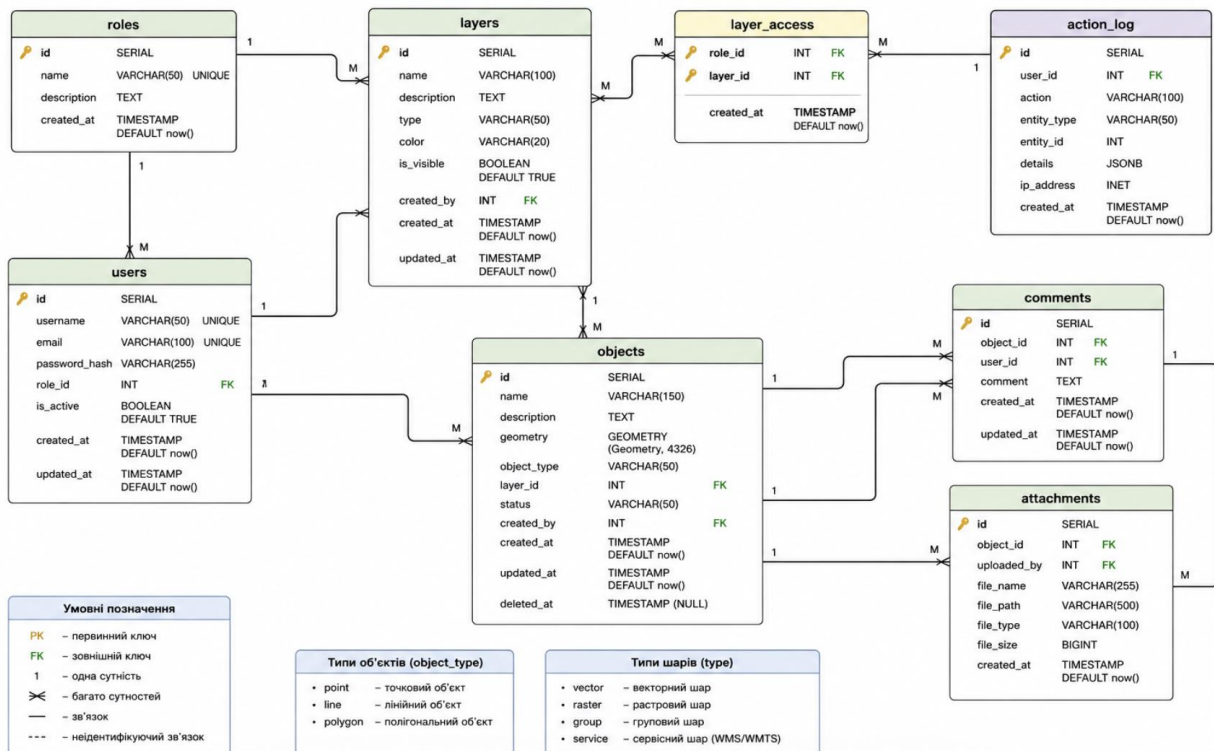


Рисунок 2.3 – ER-діаграма

При створенні бази даних також необхідно врахувати цілісність даних. Наприклад, об'єкт не може існувати без шару, коментар не може існувати без об'єкта, а вкладення повинно бути пов'язане з конкретним об'єктом. Для цього використовуються зовнішні ключі та обмеження цілісності. У разі видалення об'єкта можна передбачити каскадне видалення пов'язаних коментарів і вкладень або застосувати м'яке видалення через поле `deleted_at`.

Окрему увагу слід приділити збереженню історії змін. У системах моніторингу важливо знати, хто саме створив або змінив об'єкт. Тому у ключових таблицях доцільно використовувати поля `created_by`, `updated_by`, `created_at` та `updated_at`. Це забезпечує прозорість роботи системи та спрощує аудит дій користувачів.

2.9 Розробка механізму прикріплення медіафайлів до об'єктів

Розробка механізму прикріплення медіафайлів до об'єктів є важливою складовою апаратно-програмного комплексу моніторингу та ситуаційної обізнаності, оскільки вона дозволяє доповнювати геопросторові об'єкти фактичними матеріалами. У практичній роботі однієї лише мітки на карті або короткого текстового опису часто недостатньо для повного розуміння ситуації. Фотографії, документи, схеми або інші файли дають змогу підтвердити стан об'єкта, зафіксувати подію, уточнити обставини та створити більш повну інформаційну картку об'єкта.

У межах розроблюваної системи медіафайл розглядається як додатковий інформаційний матеріал, який прив'язується до конкретного об'єкта на карті. Такий підхід дозволяє користувачу не просто створити мітку із координатами, а сформуванати повноцінний запис із назвою, описом, географічним положенням, статусом, коментарями та прикріпленими файлами. Це особливо важливо для систем моніторингу, де необхідно документувати поточний стан об'єктів і зберігати підтверджувальні матеріали. Практична реалізація механізму прикріплення медіафайлів передбачає взаємодію трьох основних компонентів системи: клієнтської вебплатформи, серверної частини та сховища файлів. Клієнтська частина відповідає за вибір файлу користувачем і передавання його на сервер. Серверна частина виконує перевірку файлу, збереження, формування унікального імені, запис метаданих у базу даних і прив'язку файлу до об'єкта. Сховище файлів забезпечує фізичне збереження завантажених матеріалів.

На рівні клієнтського інтерфейсу механізм прикріплення файлів реалізується у картці об'єкта. Після створення або відкриття існуючого об'єкта користувач отримує доступ до блоку “Вкладення”. У цьому блоці розміщується кнопка вибору файлу, список уже прикріплених матеріалів і кнопка завантаження. Користувач може вибрати фотографію або документ зі свого пристрою, після чого клієнтська частина формує запит до серверного API.

Для передавання файлів від клієнта до сервера доцільно використовувати формат `multipart/form-data`. Саме цей формат дозволяє передавати не лише текстові поля, а й бінарні дані, зокрема фотографії або документи. Разом із файлом клієнтська частина передає ідентифікатор об'єкта, до якого потрібно прикріпити матеріал. Наприклад, якщо користувач додає фото до мітки на карті, у запиті передається `object_id` та сам файл.

На серверному рівні для завантаження файлів створюється окремий API маршрут. Практично він може мати такий вигляд:

```
POST /api/objects/:id/attachments
```

Цей маршрут приймає файл, перевіряє авторизацію користувача, визначає, чи має користувач право працювати з відповідним об'єктом, перевіряє параметри файлу та виконує його збереження. Якщо користувач не авторизований або не має доступу до шару, у якому розташований об'єкт, сервер повинен заблокувати завантаження.

Важливим етапом є перевірка типу та розміру файлу. У системі доцільно дозволити завантаження зображень у форматах JPEG, PNG, WebP, а також документів PDF або DOCX, якщо це передбачено функціональними вимогами. Для запобігання перевантаженню сервера необхідно встановити максимальний розмір одного файлу, наприклад 5 або 10 МБ. Якщо файл не відповідає вимогам, сервер повертає повідомлення про помилку.

Після успішної перевірки файл зберігається у файловому сховищі. У межах дипломного проєкту найпростішим варіантом є локальне серверне сховище, наприклад каталог `uploads`. Для кожного об'єкта можна створювати окрему папку або використовувати загальну структуру з унікальними іменами файлів. У промисловому варіанті така система може бути розширена за рахунок використання хмарного сховища, наприклад Amazon S3, Google Cloud Storage або іншого сумісного сервісу.

Щоб уникнути конфліктів імен, сервер повинен формувати унікальне ім'я файлу. Наприклад, до оригінальної назви можна додавати часову мітку,

випадковий ідентифікатор або UUID. Це дозволяє зберігати файли без ризику перезапису вже наявних матеріалів. Оригінальна назва файлу при цьому може зберігатися в базі даних окремо, щоб користувач бачив її в інтерфейсі. У базі даних для збереження інформації про вкладення використовується таблиця `attachments`. Вона містить ідентифікатор вкладення, ідентифікатор об'єкта, ідентифікатор користувача, оригінальну назву файлу, шлях до файлу, тип, розмір і дату завантаження. Сам файл не зберігається безпосередньо в базі даних, оскільки це може ускладнити масштабування системи та збільшити навантаження на СУБД. У базі зберігається лише службова інформація, яка дозволяє знайти файл у сховищі.

Практична структура таблиці `attachments` може бути такою: `attachments id, object_id, uploaded_by, file_name, file_path, file_type, file_size, created_at`.

Поле `object_id` забезпечує прив'язку вкладення до конкретного об'єкта на карті. Поле `uploaded_by` дозволяє визначити користувача, який завантажив файл. Поле `file_path` містить шлях до фізичного розташування файлу. Поля `file_type` і `file_size` використовуються для контролю типу та розміру вкладення.

Після збереження файлу сервер записує метадані у таблицю `attachments` і повертає клієнтській частині інформацію про завантажене вкладення. Клієнтський інтерфейс оновлює картку об'єкта та відображає новий файл у списку вкладень. Якщо файл є зображенням, доцільно показувати його у вигляді мініатюри. Якщо це документ, у картці може відображатися назва файлу та кнопка відкриття або завантаження.

Окремо потрібно реалізувати механізм перегляду вкладень. Коли користувач відкриває картку об'єкта, клієнтська частина надсилає запит на сервер для отримання списку вкладених файлів: `GET /api/objects/:id/attachments`.

Сервер перевіряє права доступу користувача до об'єкта і повертає список доступних вкладень. Якщо користувач не має права перегляду відповідного шару, список файлів не повертається. Це важливо, оскільки медіафайли можуть містити службову або чутливу інформацію.

Для видалення вкладення використовується окремий маршрут: `DELETE`

/api/attachments/:id.

Під час видалення сервер повинен перевірити права користувача, видалити запис із бази даних і за потреби видалити сам файл зі сховища. У системах із підвищеними вимогами до контролю доцільно реалізувати не фізичне, а м'яке видалення, коли файл залишається у сховищі, але позначається як неактивний.

Важливою вимогою є безпечний доступ до медіафайлів. Не рекомендується робити каталог завантажень повністю відкритим для всіх користувачів. Краще реалізувати контрольований доступ через серверний маршрут, який спочатку перевіряє права користувача, а вже потім передає файл. Це дозволяє гарантувати, що вкладення можуть переглядати лише ті користувачі, які мають доступ до відповідного об'єкта або шару.

Механізм прикріплення медіафайлів також повинен бути пов'язаний із журналом подій. Кожне завантаження, перегляд або видалення файлу може фіксуватися у таблиці `action_log`. Це дозволяє відстежувати дії користувачів і забезпечує контроль за змінами в системі. Наприклад, у журналі може зберігатися інформація про те, хто саме додав фото до об'єкта, коли це було зроблено та до якого об'єкта належить файл.

З практичної точки зору послідовність прикріплення медіафайлу виглядає так: користувач відкриває картку об'єкта, натискає кнопку додавання файлу, вибирає файл на своєму пристрої, клієнтська частина надсилає `multipart/form-data` запит на сервер, сервер перевіряє токен і права доступу, перевіряє тип і розмір файлу, зберігає файл у сховищі, записує метадані в базу даних, створює запис у журналі подій і повертає клієнту оновлений список вкладень. Для підвищення зручності користувачів доцільно передбачити в інтерфейсі попередній перегляд зображень перед завантаженням. Це дозволяє переконатися, що користувач вибрав правильний файл. Також варто реалізувати індикатор процесу завантаження, особливо якщо файл має великий розмір або система використовується через нестабільне з'єднання.

3 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ АПАРАТНО-ПРОГРАМНОГО КОМПЛЕКСУ

3.1 Вибір програмних засобів та технологій розробки

Для реалізації апаратно-програмного комплексу моніторингу та ситуаційної обізнаності було виконано аналіз сучасних програмних засобів, фреймворків, серверних технологій та геоінформаційних платформ, які можуть забезпечити стабільну роботу системи, підтримку геопросторових даних, багатокористувацький режим роботи та інтеграцію із вебтехнологіями. Основними критеріями вибору програмних засобів стали продуктивність, масштабованість, зручність інтеграції, підтримка роботи у режимі реального часу, наявність інструментів для роботи із картографічними сервісами та можливість реалізації механізмів інформаційної безпеки.

У процесі розробки системи було обрано веборієнтовану архітектуру, яка забезпечує доступ користувачів до функціоналу комплексу через браузер без необхідності встановлення додаткового програмного забезпечення. Такий підхід дозволяє реалізувати кросплатформеність системи та забезпечити підтримку роботи на персональних комп'ютерах, планшетах і мобільних пристроях.

Для реалізації клієнтської частини системи доцільно використовувати HTML5, CSS3 та JavaScript. HTML5 використовується для створення структури вебінтерфейсу, CSS3 забезпечує оформлення та адаптивний дизайн сторінок, а JavaScript реалізує динамічну взаємодію користувача із системою та інтерактивну роботу із картою. Використання сучасних вебтехнологій дозволяє створити зручний інтерфейс користувача та забезпечити високу швидкодію системи.

Для побудови інтерактивної карти було обрано бібліотеку Leaflet. Дана бібліотека є одним із найпоширеніших інструментів для створення веборієнтованих геоінформаційних систем. Leaflet підтримує роботу із картографічними шарами, мітками, маршрутами, полігонами та іншими геопросторовими об'єктами. Основними перевагами бібліотеки є невеликий

розмір, висока швидкість роботи, простота інтеграції та підтримка великої кількості картографічних сервісів. Використання Leaflet дозволяє реалізувати масштабування карти, додавання користувацьких об'єктів, відображення зон контролю та інтерактивну взаємодію із геопросторовими даними. Як джерело картографічних даних може використовуватись OpenStreetMap або Google Maps Platform. OpenStreetMap є відкритим картографічним сервісом, який дозволяє безкоштовно використовувати картографічні дані та інтегрувати їх у власні програмні рішення. Використання відкритої платформи забезпечує гнучкість налаштування системи та відсутність обмежень на модифікацію картографічних шарів [7]. Для реалізації серверної частини системи було обрано середовище Node.js. Дана технологія дозволяє створювати високопродуктивні серверні застосунки із підтримкою асинхронної обробки запитів та роботи у режимі реального часу. Node.js забезпечує швидку взаємодію між клієнтською та серверною частинами системи, що є особливо важливим для сервісів моніторингу та ситуаційної обізнаності. У якості серверного фреймворку використовується Express.js. Даний фреймворк забезпечує реалізацію REST API, маршрутизацію запитів, обробку даних користувачів та взаємодію із базою даних. Використання Express.js дозволяє спростити структуру серверного застосунку та забезпечити модульність програмного коду.

Для збереження інформації про користувачів, геопросторові об'єкти, шари карти та мультимедійні вкладення використовується система керування базами даних PostgreSQL із розширенням PostGIS. PostgreSQL є однією з найпотужніших реляційних баз даних, а розширення PostGIS забезпечує підтримку просторових типів даних та географічних запитів. Це дозволяє ефективно працювати із координатами, маршрутами, полігонами та іншими геопросторовими елементами.

Для забезпечення авторизації користувачів та захисту інформації використовується технологія JWT (JSON Web Token). JWT дозволяє реалізувати механізм автентифікації користувачів та контролю доступу до функцій системи. Після успішного входу до системи користувач отримує токен доступу, який

використовується для перевірки прав під час виконання запитів до серверної частини. Для передачі даних між сервером і клієнтом використовується REST API із форматом JSON. Такий підхід забезпечує стандартизований обмін інформацією між компонентами системи та спрощує інтеграцію із зовнішніми сервісами.

Для реалізації оновлення інформації у режимі реального часу використовуються WebSocket технології та бібліотека Socket.IO. Використання Socket.IO дозволяє миттєво передавати зміни між сервером та клієнтами без необхідності постійного оновлення сторінки. Це особливо важливо для систем ситуаційної обізнаності, де оперативність відображення інформації має критичне значення. У процесі розробки також використовуються сучасні засоби тестування та налагодження програмного забезпечення. Для перевірки API-запитів застосовується Postman, який дозволяє тестувати серверні маршрути, перевіряти передачу даних та контролювати коректність роботи REST API. Для розробки програмного коду може використовуватись середовище Visual Studio Code, яке підтримує роботу із JavaScript, Node.js та вебтехнологіями (табл. 3.1).

Таблиця 3.1 – Основні програмні засоби та технології розробки [2]

Програмний засіб або технологія	Призначення у системі
1	2
HTML5	Створення структури вебінтерфейсу
CSS3	Оформлення та адаптивний дизайн інтерфейсу
JavaScript	Реалізація інтерактивної логіки роботи системи
Leaflet	Відображення інтерактивної цифрової карти
OpenStreetMap	Джерело картографічних даних
Node.js	Реалізація серверної частини системи
Express.js	Побудова REST API та маршрутизація запитів
PostgreSQL	Збереження структурованих даних системи
PostGIS	Робота із геопросторовими даними та координатами
JWT (JSON Web Token)	Авторизація користувачів та контроль доступу
Socket.IO	Оновлення інформації у режимі реального часу
REST API	Передача даних між клієнтом і сервером
JSON	Формат обміну даними між компонентами
Postman	Тестування API та перевірка серверних запитів
Visual Studio Code	Середовище розробки програмного забезпечення
Git	Контроль версій та керування змінами коду

Продовження табл. 3.1

1	2
HTTPS	Захищена передача інформації мережею
Nginx	Вебсервер та проксі-сервер для розгортання системи
Docker	Контейнеризація та спрощення розгортання застосунку
Bootstrap	Адаптивне оформлення вебінтерфейсу
Multer	Завантаження та обробка мультимедійних файлів
Bcrypt	Хешування паролів користувачів
WebSocket	Двосторонній обмін даними у режимі реального часу
GeoJSON	Формат збереження геопросторових об'єктів
Firebase Storage (опціонально)	Зберігання мультимедійних вкладень та фотографій

Обраний комплекс програмних засобів дозволяє реалізувати сучасну веборієнтовану геоінформаційну систему із підтримкою багатокористувацького режиму роботи, інтерактивної карти, керування шарами, додавання користувацьких об'єктів та захищеної взаємодії між компонентами системи. Використання сучасних вебтехнологій та геоінформаційних сервісів забезпечує масштабованість системи, високу продуктивність та можливість подальшого розширення функціоналу програмного комплексу.

3.2 Реалізація серверної частини системи

Реалізація серверної частини апаратно-програмного комплексу моніторингу та ситуаційної обізнаності є одним із ключових етапів розробки системи, оскільки саме сервер забезпечує централізовану обробку даних, взаємодію із базою даних, підтримку багатокористувацького режиму роботи, керування доступом користувачів та обмін інформацією між клієнтською частиною і геоінформаційними сервісами. Серверна частина виступає основним ядром системи та реалізує логіку функціонування всіх програмних модулів. Для реалізації серверної частини системи було обрано платформу Node.js, яка дозволяє

створювати високопродуктивні вебзастосунки із підтримкою асинхронної обробки запитів та роботи у режимі реального часу. Використання Node.js забезпечує ефективну взаємодію між клієнтським інтерфейсом та сервером навіть при значній кількості одночасних користувачів. Додатково для організації серверної логіки використовується фреймворк Express.js, який спрощує побудову REST API та маршрутизацію HTTP-запитів. Архітектура серверної частини побудована за модульним принципом. Кожен функціональний елемент системи реалізується у вигляді окремого програмного модуля, що забезпечує зручність підтримки програмного коду та можливість подальшого масштабування системи.

Під час запуску серверного застосунку створюється центральний HTTP-сервер, який приймає запити від клієнтської частини системи. Після отримання запиту сервер виконує перевірку автентифікації користувача, аналізує права доступу та виконує відповідні операції із базою даних або геоінформаційними модулями. Для організації взаємодії між клієнтом та сервером використовується REST API. Такий підхід дозволяє стандартизувати передачу даних між компонентами системи та забезпечити простоту інтеграції із зовнішніми сервісами. Важливим елементом серверної частини є система авторизації та автентифікації користувачів. Для реалізації механізму захищеного входу використовується технологія JWT (JSON Web Token). Після успішного введення логіна та пароля сервер генерує унікальний токен доступу, який передається клієнту та використовується під час подальших запитів до системи. Для роботи із геопросторовими даними сервер взаємодіє із базою даних PostgreSQL та розширенням PostGIS. Серверний застосунок виконує обробку координат об'єктів, запис геометричних даних, формування просторових запитів та отримання інформації про мітки, маршрути та полігони.

Для підтримки роботи у режимі реального часу використовується бібліотека Socket.IO. Вона дозволяє реалізувати двосторонній обмін даними між сервером і клієнтами без необхідності постійного оновлення сторінки. Важливим компонентом серверної частини є модуль роботи із мультимедійними файлами.

Для завантаження фотографій та документів використовується бібліотека Multer, яка забезпечує приймання файлів через HTTP-запити та їх збереження на сервері або у хмарному сховищі. Для забезпечення стабільності роботи системи реалізовано механізм журналювання подій. Сервер фіксує дії користувачів, створення та редагування об'єктів, входи до системи, помилки та інші події. З метою забезпечення інформаційної безпеки серверна частина підтримує HTTPS-з'єднання та захищену передачу даних мережею.

Таблиця 3.2 – Основні модулі серверної частини системи

Модуль серверної частини	Призначення
Модуль авторизації	Перевірка користувачів та генерація JWT-токенів
REST API	Обмін даними між клієнтом та сервером
Модуль роботи із шарами	Керування інформаційними шарами
Модуль геопросторових об'єктів	Обробка міток, маршрутів та полігонів
Модуль бази даних	Взаємодія із PostgreSQL/PostGIS
Socket.IO	Робота системи у режимі реального часу
Модуль мультимедійних файлів	Завантаження та збереження фотографій
Модуль журналювання	Фіксація подій та дій користувачів
Модуль безпеки	Шифрування та контроль доступу
Модуль ролей користувачів	Розмежування прав доступу
Модуль фільтрації	Пошук та відбір об'єктів
Модуль резервного копіювання	Захист даних від втрати

У результаті реалізації серверної частини було створено централізовану систему обробки та збереження геопросторових даних, яка забезпечує підтримку багатокористувацького режиму роботи, інтерактивну взаємодію із картою, захищений доступ до інформації та оперативне оновлення даних у режимі реального часу. Реалізована архітектура дозволяє масштабувати систему, підключати нові модулі та інтегрувати додаткові сервіси моніторингу й ситуаційної обізнаності.

3.3 Реалізація клієнтського інтерфейсу користувача

Реалізація клієнтського інтерфейсу користувача є одним із найважливіших етапів створення апаратно-програмного комплексу моніторингу та ситуаційної обізнаності, оскільки саме через інтерфейс здійснюється взаємодія користувачів із системою, цифровою картою, інформаційними шарами та геопросторовими об'єктами. Основним завданням клієнтської частини є забезпечення зручного, швидкого та інтуїтивно зрозумілого доступу до функціональних можливостей програмного комплексу.

Клієнтський інтерфейс реалізовано як веборієнтований застосунок, що дозволяє користувачам працювати із системою через браузер без необхідності встановлення додаткового програмного забезпечення. Такий підхід забезпечує кросплатформеність системи та підтримку роботи на персональних комп'ютерах, планшетах і мобільних пристроях. Доступ до функціоналу системи здійснюється через захищене HTTPS-з'єднання, що забезпечує безпечну передачу інформації між користувачем та серверною частиною. Для реалізації інтерфейсу використано HTML5, CSS3 та JavaScript. HTML5 забезпечує формування структури сторінок, CSS3 відповідає за візуальне оформлення та адаптивний дизайн, а JavaScript реалізує інтерактивну взаємодію користувача із картою та серверною частиною системи. Використання сучасних вебтехнологій дозволяє створити динамічний інтерфейс із підтримкою швидкого оновлення даних без перезавантаження сторінки.

Основою клієнтської частини системи є інтерактивна цифрова карта, яка займає центральну частину робочої області вебзастосунку. Для реалізації картографічного функціоналу використовується бібліотека Leaflet, яка забезпечує підтримку масштабування карти, навігації, відображення геопросторових об'єктів та роботи із тематичними шарами. Використання Leaflet дозволяє інтегрувати картографічні сервіси OpenStreetMap та реалізувати функціонал створення міток, полігонів і маршрутів. Інтерфейс підтримує систему авторизації користувачів. Після відкриття вебзастосунку користувач переходить до сторінки входу, де

вводить логін та пароль. Після успішної автентифікації сервер генерує JWT-токен доступу, який використовується для подальшої взаємодії із серверною частиною системи. Інтерфейс автоматично адаптується до ролі користувача та відображає лише доступні функції й інформаційні шари.

Важливою складовою клієнтського інтерфейсу є панель керування шарами карти (табл. 3.3).

Таблиця 3.3 – Основні елементи клієнтського інтерфейсу

Елемент інтерфейсу	Призначення
Інтерактивна карта	Відображення геопросторових даних
Панель авторизації	Вхід користувачів до системи
Система шарів	Керування тематичними шарами
Мітки об'єктів	Відображення об'єктів моніторингу
Форма додавання об'єкта	Створення нових геопросторових об'єктів
Інформаційне вікно	Перегляд інформації про об'єкт
Модуль коментарів	Додавання текстових повідомлень
Мультимедійні вкладення	Прикріплення фотографій та файлів
Система пошуку	Пошук об'єктів за параметрами
Фільтрація даних	Відбір інформації за шарами та категоріями
Socket.IO	Оновлення інформації у режимі реального часу
Адаптивний дизайн	Підтримка мобільних пристроїв
HTTPS-з'єднання	Захищена передача даних
REST API	Взаємодія із серверною частиною
Кешування даних	Підвищення швидкодії системи

Користувач має можливість увімкнення або вимкнення окремих тематичних шарів, що дозволяє фільтрувати інформацію та формувати індивідуальне інформаційне середовище. Кожен шар містить окрему категорію просторових даних, наприклад маршрути, об'єкти моніторингу, зони контролю або спеціальні мітки. Для створення нових геопросторових об'єктів реалізовано інтерактивну форму додавання міток. Після вибору координат на карті користувач може вказати назву об'єкта, його опис, категорію, статус та прикріпити мультимедійні файли. Усі дані передаються на сервер через REST API та зберігаються у базі даних PostgreSQL/PostGIS.

Інтерфейс також підтримує перегляд детальної інформації про об'єкти. При натисканні на мітку відкривається інформаційне вікно, яке містить координати об'єкта, опис, фотографії, дату створення, коментарі користувачів та додаткові характеристики. Подібний підхід забезпечує швидкий доступ до інформації та підвищує ефективність ситуаційної обізнаності.

Для реалізації режиму реального часу використовується технологія WebSocket та бібліотека Socket.IO. Завдяки цьому всі зміни, внесені користувачами, миттєво відображаються у системі без необхідності перезавантаження сторінки. Якщо один користувач додає новий об'єкт або редагує існуючий, інші користувачі автоматично отримують оновлену інформацію. Особлива увага під час реалізації клієнтської частини була приділена адаптивному дизайну. Інтерфейс автоматично підлаштовується під різні розміри екрана та забезпечує коректне відображення елементів як на персональних комп'ютерах, так і на мобільних пристроях. Для реалізації адаптивності використовуються CSS Flexbox, Grid Layout та бібліотека Bootstrap.

Для підвищення швидкодії системи реалізовано кешування окремих елементів карти та асинхронне завантаження даних. Це дозволяє мінімізувати навантаження на сервер і забезпечити швидке відображення інформації навіть при значній кількості геопросторових об'єктів. У процесі реалізації клієнтського інтерфейсу також враховано вимоги інформаційної безпеки. JWT-токени зберігаються у захищеному середовищі браузера, а передача даних виконується через HTTPS-протокол. Додатково реалізовано перевірку коректності введених даних та захист від несанкціонованого доступу.

3.4 Інтеграція картографічного сервісу та ГІС-функціоналу

Інтеграція картографічного сервісу та геоінформаційного функціоналу є одним із ключових етапів розробки апаратно-програмного комплексу моніторингу

та ситуаційної обізнаності. Саме картографічна підсистема забезпечує відображення просторових даних, інтерактивну взаємодію користувачів із картою та реалізацію основних функцій ситуаційної обізнаності. У процесі реалізації системи було обрано веборієнтований підхід до інтеграції картографічного сервісу. Основою картографічного функціоналу стала бібліотека Leaflet, яка забезпечує створення інтерактивних карт у веббраузері та підтримує інтеграцію з різними джерелами геопросторових даних (табл. 3.4).

Таблиця 3.4 – Основні компоненти інтеграції картографічного сервісу та ГІС-функціоналу

Компонент	Призначення
Leaflet	Відображення інтерактивної карти
OpenStreetMap	Джерело картографічних даних
GeoJSON	Формат збереження геопросторових об'єктів
PostgreSQL	Збереження структурованих даних
PostGIS	Робота із просторовими даними
REST API	Передача геоданих між сервером і клієнтом
WebSocket	Оновлення даних у режимі реального часу
Socket.IO	Двосторонній обмін інформацією
Тематичні шари	Організація просторової інформації
Маркери та мітки	Відображення об'єктів моніторингу
Полігони	Формування зон контролю
Маршрути	Аналіз переміщення об'єктів
Спливаючі вікна	Перегляд інформації про об'єкти
Кешування тайлів	Підвищення швидкодії системи
HTTPS	Захищена передача геопросторових даних

Як джерело картографічних даних використовується сервіс OpenStreetMap. Дана платформа забезпечує відкритий доступ до цифрових карт та дозволяє інтегрувати геодані без необхідності придбання комерційної ліцензії. Під час інтеграції картографічного сервісу було реалізовано механізм відображення карти у центральній області вебінтерфейсу. Після завантаження сторінки система автоматично ініціалізує карту, задає початкові координати центру та рівень масштабування. Одним із основних елементів ГІС-функціоналу є підтримка

тематичних інформаційних шарів. Кожен шар містить окрему категорію просторових даних, наприклад об'єкти моніторингу, маршрути переміщення, зони ризику або службову інформацію.

У межах клієнтського інтерфейсу реалізовано можливість створення користувацьких об'єктів безпосередньо на карті. Користувач може додавати нові мітки, формувати маршрути, створювати полігони та визначати зони контролю. Для забезпечення інтерактивності реалізовано систему спливаючих інформаційних вікон. При натисканні на об'єкт користувач отримує доступ до детальної інформації, яка містить назву об'єкта, опис, координати, дату створення, фотографії та коментарі.

Для підтримки роботи у режимі реального часу використовується технологія WebSocket та бібліотека Socket.IO. У випадку зміни положення об'єкта або створення нової мітки інформація миттєво відображається у всіх підключених користувачів. У результаті інтеграції картографічного сервісу та ГІС-функціоналу було створено сучасне геоінформаційне середовище, яке забезпечує інтерактивну роботу із картою, підтримку багатокористувацького режиму та ефективне управління геопросторовими даними.

3.5 Реалізація системи ролей та керування доступом користувачів

Реалізація системи ролей та керування доступом користувачів є важливою складовою апаратно-програмного комплексу моніторингу та ситуаційної обізнаності, оскільки система працює із геопросторовими даними, службовою інформацією та багатокористувацьким режимом роботи. Основним завданням механізму контролю доступу є забезпечення безпечної взаємодії користувачів із системою, обмеження доступу до окремих функцій та запобігання несанкціонованим змінам інформації.

У процесі розробки програмного комплексу було реалізовано рольову модель

доступу, відповідно до якої кожен користувач отримує певний набір прав залежно від своєї ролі у системі. Подібний підхід дозволяє організувати структурований контроль доступу та забезпечити захист геопросторових даних від несанкціонованого використання.

Основою системи доступу є механізм автентифікації користувачів. Перед початком роботи із системою користувач повинен пройти процедуру авторизації шляхом введення логіна та пароля. Після успішної перевірки облікових даних серверна частина системи генерує JWT-токен доступу, який використовується для підтвердження особи користувача під час подальших запитів до серверної частини. Для захисту облікових записів паролі користувачів зберігаються у базі даних у вигляді хешів із використанням алгоритму bcrypt. Це забезпечує високий рівень інформаційної безпеки та унеможливорює отримання відкритих паролів навіть у випадку компрометації бази даних.

У системі реалізовано декілька основних ролей користувачів. Найвищий рівень доступу має адміністратор системи. Адміністратор отримує повний контроль над програмним комплексом та має можливість створювати нові облікові записи, змінювати ролі користувачів, керувати інформаційними шарами та контролювати роботу всієї системи (табл. 3.5).

Таблиця 3.5 – Основні ролі користувачів та права доступу

Роль користувача	Основні права доступу
Адміністратор	Повний доступ до системи та налаштувань
Оператор	Додавання та редагування об'єктів
Спостерігач	Перегляд інформації без редагування
Авторизація	Перевірка облікових даних користувача
JWT-токен	Підтвердження особи користувача
bcrypt	Захист паролів користувачів
Керування шарами	Контроль доступу до тематичних шарів
Журнал подій	Фіксація дій користувачів
REST API	Перевірка доступу до серверних маршрутів
HTTPS	Захищена передача інформації
Контроль сесій	Автоматичне завершення неактивних сесій
Розмежування прав	Обмеження доступу до функцій системи

Для контролю дій користувачів реалізовано систему журналювання подій. Серверна частина фіксує входи до системи, створення та редагування об'єктів, зміну прав доступу, видалення інформації та інші дії користувачів. Під час реалізації системи доступу особливу увагу було приділено захисту API-запитів. Усі серверні маршрути перевіряють JWT-токен користувача перед виконанням операцій. Для підвищення рівня безпеки також реалізовано автоматичне завершення сесії після закінчення терміну дії токена. У випадку бездіяльності користувача система вимагає повторної авторизації. Реалізована система ролей та керування доступом забезпечує безпечну роботу апаратно-програмного комплексу, підтримує багатокористувацький режим роботи та дозволяє ефективно контролювати доступ до геопросторових даних і функціональних можливостей системи.

3.6 Тестування функціональних можливостей системи

Тестування функціональних можливостей апаратно-програмного комплексу моніторингу та ситуаційної обізнаності є важливим етапом розробки системи, оскільки дозволяє перевірити коректність роботи програмних модулів, стабільність функціонування серверної та клієнтської частини, правильність обробки геопросторових даних та ефективність механізмів взаємодії між компонентами системи. У процесі тестування перевірялась робота всіх основних функціональних модулів системи, зокрема механізмів авторизації користувачів, відображення інтерактивної карти, роботи із тематичними шарами, створення та редагування геопросторових об'єктів, прикріплення мультимедійних файлів, передачі інформації у режимі реального часу та системи контролю доступу.

На початковому етапі було виконано тестування модуля авторизації користувачів. Перевірялась коректність введення логіна та пароля, генерація JWT-токенів, перевірка терміну дії сесії та обмеження доступу до функцій системи для

неавторизованих користувачів. Окрему увагу приділено тестуванню інтерактивної карти та механізму роботи із геопросторовими даними. Перевірялась швидкість завантаження карти, коректність масштабування, переміщення картою, відображення тематичних шарів та стабільність роботи картографічного сервісу при великій кількості об'єктів.

Для перевірки роботи у режимі реального часу було проведено тестування технології WebSocket та бібліотеки Socket.IO. У процесі тестування один користувач створював або редагував об'єкт на карті, а інші користувачі перевіряли швидкість оновлення інформації у власному інтерфейсі.

Таблиця 3.6 – Результати тестування функціональних можливостей системи

№	Функціональний модуль	Тип тестування	Результат
1	Авторизація користувачів	Перевірка логіна та пароля	Успішно
2	JWT-автентифікація	Перевірка токенів доступу	Успішно
3	Інтерактивна карта	Завантаження та масштабування	Успішно
4	Тематичні шари	Увімкнення та вимкнення шарів	Успішно
5	Геопросторові об'єкти	Створення та редагування міток	Успішно
6	Робота із маршрутами	Відображення маршрутів	Успішно
7	Полігони та зони	Формування зон контролю	Успішно
8	Мультимедійні файли	Завантаження фотографій	Успішно
9	REST API	Передача JSON-даних	Успішно
10	WebSocket	Оновлення інформації у реальному часі	Успішно
11	Socket.IO	Синхронізація між користувачами	Успішно
12	PostgreSQL/PostGIS	Збереження геопросторових даних	Успішно
13	Система ролей	Перевірка прав доступу	Успішно
14	HTTPS	Захищена передача даних	Успішно
15	Навантажувальне тестування	Одночасна робота користувачів	Успішно

Важливим етапом стало тестування системи ролей та керування доступом. Перевірялись права адміністратора, оператора та спостерігача. Було встановлено, що система коректно обмежує доступ до функцій відповідно до ролі користувача

та не дозволяє виконувати заборонені операції. Для оцінки стабільності серверної частини проводилось навантажувальне тестування. Під час тестування імітувалась одночасна робота декількох користувачів із картою, шарами та геопросторовими об'єктами. Для оцінки інформаційної безпеки системи проводилась перевірка захищеності HTTPS-з'єднання, перевірка JWT-токенів та тестування механізмів захисту від несанкціонованого доступу.

Після проведення тестування було встановлено, що апаратно-програмний комплекс коректно виконує основні функції моніторингу та ситуаційної обізнаності, забезпечує стабільну роботу із геопросторовими даними та підтримує багатокористувацький режим роботи.

3.7 Аналіз результатів роботи та оцінка ефективності системи

Після завершення реалізації та тестування апаратно-програмного комплексу моніторингу та ситуаційної обізнаності було проведено аналіз результатів роботи системи та оцінювання її ефективності. Основною метою аналізу є визначення рівня відповідності реалізованого програмного комплексу поставленим функціональним вимогам, оцінка стабільності роботи системи, швидкодії, ефективності обробки геопросторових даних та рівня забезпечення ситуаційної обізнаності користувачів.

У процесі експериментальної перевірки системи було встановлено, що реалізований програмний комплекс забезпечує стабільну роботу із цифровою картою, підтримує інтерактивну взаємодію користувачів із геопросторовими даними та дозволяє виконувати оперативне оновлення інформації у режимі реального часу.

Проведений аналіз показав, що використання бібліотеки Leaflet та сервісу OpenStreetMap забезпечує ефективне відображення геопросторових даних та підтримку роботи із великою кількістю об'єктів на карті.

Одним із ключових критеріїв оцінювання ефективності системи є швидкість обробки інформації. Результати тестування показали, що серверна частина системи на базі Node.js та PostgreSQL/PostGIS забезпечує швидке виконання запитів до бази даних та оперативну передачу інформації між клієнтською і серверною частинами.

Важливим показником ефективності є підтримка роботи у режимі реального часу. Використання технології WebSocket та бібліотеки Socket.IO дозволило реалізувати миттєве оновлення інформації на карті без необхідності перезавантаження сторінки (табл. 3.7).

Таблиця 3.7 – Оцінка ефективності роботи системи

Критерій оцінювання	Результат оцінювання	Характеристика
Швидкодія системи	Висока	Оперативна обробка геоданих та запитів
Робота у реальному часі	Успішно реалізовано	Миттєве оновлення інформації
Стабільність роботи	Висока	Відсутність критичних помилок
Захист інформації	Високий рівень	JWT та HTTPS-захист
Робота із картою	Стабільна	Коректне масштабування та навігація
Підтримка геоданих	Повна	Робота із мітками, маршрутами та полігонами
Багатокористувацький режим	Підтримується	Одночасна робота декількох користувачів
Робота із шарами	Ефективна	Фільтрація та керування даними
Інтерфейс користувача	Зручний	Інтерактивний та адаптивний дизайн
Масштабованість системи	Висока	Можливість розширення функціоналу
Інформаційна безпека	Висока	Контроль доступу та шифрування
Робота із файлами	Стабільна	Завантаження та перегляд мультимедіа
REST API	Стабільний	Коректна передача JSON-даних
Навантажувальна стійкість	Висока	Стабільна робота при навантаженні

Проведений аналіз результатів роботи показав, що розроблений апаратно-програмний комплекс повністю виконує поставлені функціональні завдання та забезпечує ефективну роботу із геопросторовими даними. Система підтримує інтерактивну роботу із картою, забезпечує оперативне оновлення інформації,

реалізує багатокористувацький режим роботи та підтримує високий рівень інформаційної безпеки.

У результаті оцінки ефективності встановлено, що реалізований програмний комплекс може використовуватись для задач моніторингу територій, ситуаційної обізнаності, координації дій користувачів та аналізу геопросторової інформації. Використання сучасних вебтехнологій, геоінформаційних сервісів та серверних рішень забезпечує можливість подальшого розвитку системи та її адаптації до потреб різних організацій і сфер діяльності.

4 ЗАБЕЗПЕЧЕННЯ БЕЗПЕКИ, ПРОДУКТИВНОСТІ ТА НАДІЙНОСТІ АПАРАТНО-ПРОГРАМНОГО КОМПЛЕКСУ

4.1 Аналіз загроз інформаційній безпеці геоінформаційної системи

У сучасних умовах активного розвитку інформаційних технологій та широкого використання геоінформаційних систем особливого значення набуває питання забезпечення інформаційної безпеки програмних комплексів моніторингу та ситуаційної обізнаності. Геоінформаційні системи працюють із великими обсягами просторових даних, службовою інформацією, координатами об'єктів та користувацькими даними, що робить їх потенційною ціллю для кібератак, несанкціонованого доступу та спроб порушення цілісності інформації.

Інформаційна безпека геоінформаційної системи являє собою комплекс організаційних, програмних та технічних заходів, спрямованих на забезпечення конфіденційності, цілісності та доступності даних. Основним завданням системи захисту є запобігання несанкціонованому доступу до інформації, блокування спроб втручання у роботу системи та захист геопросторових даних від модифікації або втрати [5]. Однією з основних загроз для геоінформаційних систем є несанкціонований доступ до серверної частини або бази даних. У випадку успішної атаки злоумисник може отримати доступ до координат об'єктів, службової

інформації, облікових записів користувачів та інших конфіденційних даних.

Важливою загрозою є компрометація облікових записів користувачів. Якщо система не використовує захищені механізми автентифікації або зберігає паролі у відкритому вигляді, це може призвести до викрадення облікових даних та отримання несанкціонованого доступу до функціоналу системи.

Суттєву небезпеку становлять атаки типу SQL Injection. Подібні атаки спрямовані на отримання доступу до бази даних шляхом підміни SQL-запитів через форми введення або API-запити (табл. 4.1).

Таблиця 4.1 – Основні загрози інформаційній безпеці геоінформаційної системи

Загроза	Характеристика загрози	Можливі наслідки
Несанкціонований доступ	Отримання доступу до системи сторонніми особами	Викрадення або зміна даних
Компрометація облікових записів	Викрадення логінів та паролів	Отримання контролю над системою
SQL Injection	Впровадження шкідливих SQL-запитів	Пошкодження бази даних
XSS-атаки	Впровадження шкідливого JavaScript-коду	Викрадення токенів доступу
CSRF-атаки	Виконання небажаних дій від імені користувача	Зміна або видалення інформації
DDoS-атаки	Перевантаження серверної частини	Відмова у доступі до системи
Перехоплення трафіку	Отримання інформації під час передачі	Викрадення конфіденційних даних
Втрата даних	Пошкодження або видалення інформації	Порушення роботи системи
Вірусне програмне забезпечення	Впровадження шкідливих програм	Пошкодження серверної інфраструктури
Помилки користувачів	Неправильні дії операторів системи	Пошкодження або зміна даних
Несанкціоноване редагування	Зміна геопросторової інформації	Втрата достовірності даних
Витік мультимедійних файлів	Несанкціонований доступ до вкладень	Розголошення службової інформації
Відсутність резервного копіювання	Неможливість відновлення даних	Повна втрата інформації
Порушення роботи API	Некоректна взаємодія компонентів	Нестабільна робота системи

Проведений аналіз загроз інформаційній безпеці показав, що геоінформаційні системи потребують комплексного підходу до захисту даних та контролю доступу користувачів.

Реалізація механізмів автентифікації, шифрування даних, резервного копіювання та контролю дій користувачів дозволяє значно підвищити рівень захищеності апаратно-програмного комплексу. Урахування сучасних кіберзагроз та використання засобів інформаційної безпеки забезпечує стабільну роботу системи моніторингу та ситуаційної обізнаності, знижує ризик несанкціонованого доступу до геопросторових даних та дозволяє підтримувати безпечне функціонування геоінформаційного середовища.

4.2 Реалізація механізмів захисту користувачів та геоданих

Реалізація механізмів захисту користувачів та геопросторових даних є одним із найважливіших етапів створення апаратно-програмного комплексу моніторингу та ситуаційної обізнаності. Оскільки система працює із конфіденційною інформацією, координатами об'єктів, службовими повідомленнями та мультимедійними файлами, необхідно забезпечити високий рівень інформаційної безпеки та захист від несанкціонованого доступу.

Основною метою реалізації механізмів захисту є забезпечення конфіденційності, цілісності та доступності інформації. Для досягнення цієї мети у системі реалізовано комплекс програмних та організаційних засобів захисту, які охоплюють процеси авторизації користувачів, шифрування даних, захисту API-запитів, контролю доступу та резервного копіювання інформації.

Одним із основних механізмів захисту є система автентифікації користувачів. Для підтвердження особи користувача використовується JWT (JSON Web Token). Після успішного введення логіна та пароля серверна частина системи генерує токен доступу, який використовується під час виконання подальших запитів до серверної

частини [28].

Для захисту облікових записів користувачів реалізовано механізм хешування паролів за допомогою алгоритму bcrypt. Паролі не зберігаються у відкритому вигляді, а перетворюються у спеціальні хеш-значення, що унеможлиблює їх відновлення навіть у випадку компрометації бази даних.

Важливим елементом захисту є використання HTTPS-протоколу для передачі інформації між клієнтською та серверною частинами системи. HTTPS забезпечує шифрування трафіку та захищає дані від перехоплення під час передачі мережею (табл. 4.2).

Таблиця 4.2 – Основні механізми захисту користувачів та геоданих

Механізм захисту	Призначення	Результат використання
JWT-автентифікація	Підтвердження особи користувача	Захищений доступ до системи
bcrypt	Хешування паролів	Захист облікових записів
HTTPS	Шифрування трафіку	Захищена передача інформації
SSL-сертифікати	Захист мережевого з'єднання	Захист від перехоплення даних
Параметризовані SQL-запити	Захист від SQL Injection	Безпечна робота із БД
Валідація введених даних	Перевірка коректності інформації	Зменшення ризику атак
Фільтрація HTML-коду	Захист від XSS-атак	Блокування шкідливих сценаріїв
Контроль ролей	Розмежування прав доступу	Захист службової інформації
Журналювання подій	Фіксація дій користувачів	Контроль роботи системи
Резервне копіювання	Захист від втрати даних	Відновлення інформації
Обмеження кількості запитів	Захист від DDoS-атак	Підвищення стабільності роботи
Socket.IO із авторизацією	Захист передачі даних у реальному часі	Безпечна синхронізація
Захист REST API	Контроль доступу до серверних маршрутів	Блокування несанкціонованих запитів
Кешування даних	Зменшення навантаження на сервер	Підвищення продуктивності

Для забезпечення цілісності геопросторових даних реалізовано механізм

журналювання подій. Усі дії користувачів фіксуються у спеціальному журналі, який містить інформацію про авторизацію, створення, редагування та видалення об'єктів. Важливим елементом захисту є система резервного копіювання бази даних. Резервні копії створюються автоматично через визначені проміжки часу та зберігаються на окремому сервері або у хмарному сховищі.

Для підвищення стійкості серверної частини до DDoS-атак реалізовано механізми обмеження кількості запитів від одного користувача, балансування навантаження та використання кешування даних. Після реалізації механізмів захисту було встановлено, що система забезпечує високий рівень інформаційної безпеки та дозволяє ефективно захищати користувачів і геопросторові дані від сучасних кіберзагроз.

4.3 Оптимізація продуктивності веборієнтованої ГІС системи

Оптимізація продуктивності веборієнтованої геоінформаційної системи є важливим етапом розробки апаратно-програмного комплексу моніторингу та ситуаційної обізнаності, оскільки ефективність роботи системи безпосередньо впливає на швидкість обробки геопросторових даних, стабільність функціонування сервісу та зручність роботи користувачів. Основною метою оптимізації продуктивності є зменшення навантаження на серверну частину, прискорення передачі даних між компонентами системи та підвищення швидкості обробки геопросторової інформації.

Одним із основних напрямів оптимізації є використання кешування даних. У процесі роботи геоінформаційної системи значна кількість запитів пов'язана із повторним отриманням однакових картографічних тайлів та геопросторових даних. Для підвищення продуктивності клієнтського інтерфейсу використовується асинхронне завантаження даних. Геопросторові об'єкти, шари та мультимедійні файли завантажуються поступово без блокування роботи інтерфейсу

користувача [29].

Для оптимізації роботи із базою даних PostgreSQL/PostGIS реалізовано індексацію геопросторових полів. Використання просторових індексів дозволяє значно прискорити виконання запитів, пов'язаних із пошуком об'єктів, побудовою маршрутів та аналізом геоданих (табл. 4.3).

Таблиця 4.3 – Основні методи оптимізації продуктивності веборієнтованої ГІС системи

Метод оптимізації	Призначення	Результат використання
Кешування тайлів	Зменшення кількості запитів до сервера	Прискорення завантаження карти
Асинхронне завантаження	Поступове отримання даних	Плавна робота інтерфейсу
GeoJSON	Оптимізоване збереження геоданих	Зменшення обсягу переданих даних
Gzip-компресія	Стиснення HTTP-відповідей	Прискорення передачі інформації
Просторові індекси PostGIS	Оптимізація геопросторових запитів	Підвищення швидкості пошуку
Node.js	Асинхронна обробка запитів	Підтримка великої кількості користувачів
WebSocket	Робота у режимі реального часу	Мінімальні затримки оновлення
Socket.IO	Синхронізація даних	Оперативний обмін інформацією
Адаптивне завантаження шарів	Відображення лише необхідних даних	Зменшення навантаження на клієнт
Масштабування фотографій	Оптимізація мультимедійних файлів	Зменшення розміру файлів
REST API оптимізація	Скорочення кількості запитів	Підвищення продуктивності
Балансування навантаження	Розподіл серверних ресурсів	Стабільна робота системи
Кешування запитів	Повторне використання результатів	Зменшення навантаження на БД
Обмеження кількості запитів	Захист серверної частини	Підвищення стабільності роботи

Для забезпечення швидкого оновлення інформації у режимі реального часу використовується технологія WebSocket та бібліотека Socket.IO. На відміну від

постійного опитування сервера, WebSocket забезпечує двосторонній обмін даними між сервером та клієнтом із мінімальними затримками.

Одним із важливих аспектів оптимізації є адаптивне завантаження картографічних шарів. Система завантажує лише ті шари та об'єкти, які знаходяться у межах поточної області перегляду користувача.

Після реалізації механізмів оптимізації було встановлено, що система забезпечує стабільну роботу навіть при значній кількості геопросторових об'єктів та одночасному підключенні декількох користувачів. У результаті проведеної оптимізації веборієнтована геоінформаційна система отримала високий рівень продуктивності, підтримку роботи у режимі реального часу та можливість масштабування функціоналу без критичного зниження швидкодії.

4.4 Забезпечення надійності та масштабованості системи

Забезпечення надійності та масштабованості веборієнтованої геоінформаційної системи є одним із ключових напрямів розробки апаратно-програмного комплексу моніторингу та ситуаційної обізнаності. У процесі експлуатації система повинна забезпечувати безперервну роботу, стабільне функціонування серверної та клієнтської частини, підтримку великої кількості користувачів та можливість подальшого розширення функціоналу без суттєвого зниження продуктивності.

Надійність геоінформаційної системи визначається її здатністю стабільно функціонувати протягом тривалого часу без критичних помилок, втрати даних або зупинки сервісу. Для забезпечення високого рівня надійності у програмному комплексі реалізовано комплекс програмних, серверних та організаційних рішень.

Одним із основних механізмів забезпечення надійності є використання модульної архітектури системи. Кожен функціональний компонент реалізований у вигляді окремого модуля, що дозволяє ізолювати помилки, спрощує супровід

програмного забезпечення та забезпечує можливість оновлення окремих частин системи без зупинки всього комплексу [30].

Для забезпечення стабільної роботи серверної частини використовується платформа Node.js, яка підтримує асинхронну модель обробки запитів та дозволяє ефективно працювати із великою кількістю одночасних підключень.

Важливим елементом надійності є використання системи резервного копіювання даних. Резервні копії бази даних PostgreSQL/PostGIS створюються автоматично через визначені проміжки часу та зберігаються на окремому сервері або у хмарному середовищі (табл. 4.4).

Таблиця 4.4 – Основні засоби забезпечення надійності та масштабованості системи

Засіб забезпечення	Призначення	Результат використання
Модульна архітектура	Ізоляція функціональних компонентів	Спрощення підтримки системи
Node.js	Асинхронна обробка запитів	Висока продуктивність
PostgreSQL/PostGIS	Зберігання геопросторових даних	Стабільна робота із БД
Резервне копіювання	Захист від втрати інформації	Відновлення даних
Журнування подій	Контроль роботи системи	Моніторинг помилок
Балансування навантаження	Розподіл серверних ресурсів	Підвищення стабільності
Docker	Контейнеризація сервісів	Підвищення відмовостійкості
REST API	Інтеграція компонентів системи	Масштабування функціоналу
Просторові індекси	Оптимізація геозапитів	Прискорення роботи із даними
WebSocket	Робота у режимі реального часу	Мінімальні затримки
Socket.IO	Синхронізація між користувачами	Стабільний обмін даними
Кешування тайлів	Зменшення навантаження на сервер	Прискорення роботи карти
Адаптивне завантаження	Оптимізація клієнтської частини	Стабільний вебінтерфейс
HTTPS	Захищена передача інформації	Безпечна робота системи
Обмеження кількості запитів	Захист від перевантаження	Стабільність серверної частини

Для контролю працездатності програмного комплексу реалізовано механізм журналювання подій та моніторингу серверної інфраструктури. Усі критичні події, помилки, спроби несанкціонованого доступу та дії користувачів фіксуються у журналі системи.

Масштабованість системи визначає її здатність підтримувати збільшення кількості користувачів, геопросторових об'єктів та функціональних модулів без суттєвого зниження продуктивності.

Для підтримки масштабованості бази даних PostgreSQL/PostGIS реалізовано систему індексації та оптимізації геопросторових запитів. Просторові індекси дозволяють ефективно працювати із великими обсягами геоданих.

У результаті проведеної оптимізації та впровадження механізмів масштабування веборієнтована геоінформаційна система отримала високий рівень відмовостійкості, підтримку багатокористувацького режиму роботи та можливість подальшого розвитку функціоналу без критичного зниження продуктивності програмного комплексу.

ВИСНОВКИ

У результаті виконання дипломної роботи було розроблено веборієнтовану геоінформаційну систему моніторингу та ситуаційної обізнаності, яка забезпечує ефективну роботу із геопросторовими даними, підтримує багатокористувацький режим роботи та дозволяє виконувати оперативний аналіз інформації у режимі реального часу. У процесі дослідження було проведено комплексний аналіз сучасних підходів до побудови геоінформаційних систем, досліджено архітектурні особливості веборієнтованих ГІС платформ та визначено основні принципи організації систем ситуаційної обізнаності. У роботі проаналізовано сучасні технології веброзробки, серверної обробки даних та геоінформаційного моделювання. На основі проведеного аналізу було обґрунтовано вибір програмних засобів та технологій розробки, серед яких Node.js, PostgreSQL/PostGIS, Leaflet, Socket.IO, REST API та OpenStreetMap. Використання зазначених технологій дозволило створити сучасний програмний комплекс із підтримкою інтерактивної роботи із картою, геопросторовими об'єктами та тематичними шарами.

У процесі розробки було реалізовано серверну частину системи, яка забезпечує обробку клієнтських запитів, роботу із базою даних, передачу геоданих та підтримку механізмів авторизації користувачів. Створено клієнтський вебінтерфейс користувача із підтримкою інтерактивної карти, масштабування, роботи із мітками, маршрутами та зонами контролю. Реалізовано інтеграцію картографічного сервісу та ГІС-функціоналу, що дозволило забезпечити ефективну роботу із просторовою інформацією.

Важливим результатом роботи стала реалізація системи ролей та керування доступом користувачів. У системі впроваджено JWT-автентифікацію, механізми розмежування прав доступу та захист облікових записів користувачів за допомогою алгоритму bcrypt. Це дозволило забезпечити безпечну взаємодію користувачів із програмним комплексом та обмежити доступ до функціоналу

відповідно до ролей користувачів.

У межах роботи було проведено тестування функціональних можливостей системи. Результати тестування підтвердили коректність роботи серверної та клієнтської частини, стабільність функціонування картографічного сервісу, ефективність роботи із геопросторовими даними та підтримку роботи у режимі реального часу. Проведений аналіз результатів роботи показав, що система забезпечує високу швидкодію, стабільність роботи та підтримку багатокористувацького режиму. Особливу увагу у дипломній роботі приділено питанням інформаційної безпеки геоінформаційної системи. Було проаналізовано основні загрози інформаційній безпеці, серед яких SQL Injection, XSS-атаки, CSRF-запити, DDoS-атаки та несанкціонований доступ до даних. Для забезпечення захисту користувачів та геопросторових даних реалізовано HTTPS-з'єднання, систему контролю доступу, журналювання подій, резервне копіювання та механізми захисту REST API.

У процесі дослідження також було реалізовано механізми оптимізації продуктивності веборієнтованої ГІС системи. Використання кешування даних, асинхронного завантаження інформації, WebSocket-з'єднань та просторових індексів PostGIS дозволило значно підвищити швидкодію системи та забезпечити ефективну роботу із великими обсягами геопросторової інформації.

Для забезпечення надійності та масштабованості програмного комплексу реалізовано модульну архітектуру системи, балансування навантаження, контейнеризацію сервісів та механізми резервного копіювання даних. Це забезпечило високу відмовостійкість системи та можливість подальшого розширення функціоналу без критичного зниження продуктивності.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Поморцева О. Є. Геоінформаційні системи в задачах моніторингу навколишнього середовища : конспект лекцій. Харків : ХНУМГ ім. О. М. Бекетова, 2022. 104 с. URL: https://eprints.kname.edu.ua/63161/1/2022_225Л_Конспект_Л_ГІС%20Моніторинг_укр3.pdf (дата звернення: 19.05.2026).
2. Спека А. Ю. Геоінформаційні системи та технології : навчальний посібник. Полтава : НУПП, 2024. URL: https://reposit.nupp.edu.ua/bitstream/PoltNTU/15213/1/Спека_2мБЗ.pdf (дата звернення: 19.05.2026).
3. Бондар О. І., Машков О. А., Пашков Д. П., Ващенко В. М. Моніторинг стану навколишнього середовища : підручник. Київ : ДЕА, 2018. 390 с. URL: <https://dglib.nubip.edu.ua/server/api/core/bitstreams/b5ed705f-85a8-4e7c-8d77-db2eea60ce9e/content> (дата звернення: 19.05.2026).
4. Андреев С. М., Жилін В. А. Застосування технологій геоінформаційних систем для побудови картографічних моделей залізничних сполучень. Системи управління, навігації та зв'язку. 2021. №1(63). С. 4–11. URL: <https://journals.nupp.edu.ua/sunz/article/view/2500> (дата звернення: 19.05.2026).
5. Коваленко Л. О., Костюк Ю. М. Основи проектування інформаційних систем : навчальний посібник. Київ : КПП ім. Ігоря Сікорського, 2021. URL: <https://ela.kpi.ua/server/api/core/bitstreams/8c0a4a7b-c8f5-4cb2-8d2b-15d9b7cb0d9d/content> (дата звернення: 19.05.2026).
6. Олійник А. О., Субботін С. О. Інформаційні технології та системи : навчальний посібник. Запоріжжя : ЗНТУ, 2020. URL: http://eir.zntu.edu.ua/bitstream/123456789/7073/1/Oliinyk_Information_systems.pdf (дата звернення: 19.05.2026).
7. PostgreSQL Global Development Group. PostgreSQL Documentation. URL: <https://www.postgresql.org/docs/> (дата звернення: 19.05.2026).
8. PostGIS Project Steering Committee. PostGIS Documentation. URL:

<https://postgis.net/documentation/> (дата звернення: 19.05.2026).

9. Leaflet Contributors. Leaflet JavaScript Library Documentation. URL: <https://leafletjs.com/reference.html> (дата звернення: 19.05.2026).

10. OpenStreetMap Foundation. OpenStreetMap Documentation. URL: https://wiki.openstreetmap.org/wiki/Main_Page (дата звернення: 19.05.2026).

11. Mozilla Foundation. MDN Web Docs: REST API. URL: <https://developer.mozilla.org/en-US/docs/Glossary/REST> (дата звернення: 19.05.2026).

12. Node.js Foundation. Node.js Documentation. URL: <https://nodejs.org/docs/latest/api/> (дата звернення: 19.05.2026).

13. Биков В. Ю., Спирін О. М. Цифрова трансформація освіти і науки: теорія і практика : монографія. Київ : Інститут цифровізації освіти НАПН України, 2022. 284 с. URL: <https://lib.iitta.gov.ua/731640/> (дата звернення: 25.05.2026).

14. Морзе Н. В., Вембер В. П., Гладун М. А. Інформаційні системи : навчальний посібник. Київ : Університет імені Бориса Грінченка, 2021. 324 с. URL: <https://elibrary.kubg.edu.ua/id/eprint/36513/> (дата звернення: 25.05.2026).

15. Бурачек В. Г., Железняк О. О. Геоінформаційні системи : навчальний посібник. Київ : КНУБА, 2020. 208 с. URL: <http://repository.knuba.edu.ua/handle/987654321/8142> (дата звернення: 25.05.2026).

16. Згуровський М. З., Панкратова Н. Д. Основи системного аналізу : підручник. Київ : Видавнича група BHV, 2020. 544 с. URL: <https://ela.kpi.ua/handle/123456789/49730> (дата звернення: 25.05.2026).

17. Longley P. A., Goodchild M. F., Maguire D. J., Rhind D. W. Geographic Information Systems and Science. 5th ed. Hoboken : Wiley, 2021. 544 p. URL: <https://www.wiley.com/en-us/Geographic+Information+Systems+and+Science%2C+5th+Edition-p-9781119676951> (дата звернення: 25.05.2026).

18. Heywood I., Cornelius S., Carver S. An Introduction to Geographical Information Systems. 5th ed. London : Pearson Education, 2019. 520 p. URL: <https://www.pearson.com/en-us/subject-catalog/p/introduction-to-geographical->

information-systems/P200000003481 (дата звернення: 25.05.2026).

19. Obe R., Hsu L. PostGIS in Action. 3rd ed. Shelter Island : Manning Publications, 2021. 520 p. URL: <https://www.manning.com/books/postgis-in-action-third-edition> (дата звернення: 25.05.2026).

20. Peterson M. P. Mapping in the Cloud. New York : Guilford Press, 2020. 398 p. URL: <https://www.guilford.com/books/Mapping-in-the-Cloud/Michael-Peterson/9781462545874> (дата звернення: 25.05.2026).

21. Stallings W. Network Security Essentials: Applications and Standards. 7th ed. Hoboken : Pearson, 2021. 816 p. URL: <https://www.pearson.com/en-us/subject-catalog/p/network-security-essentials-applications-and-standards/P200000003479> (дата звернення: 25.05.2026).

22. Sommerville I. Software Engineering. 10th ed. Boston : Pearson, 2020. 816 p. URL: <https://www.pearson.com/en-us/subject-catalog/p/software-engineering/P200000003192> (дата звернення: 25.05.2026).

23. Pressman R. S., Maxim B. R. Software Engineering: A Practitioner's Approach. 9th ed. New York : McGraw-Hill Education, 2020. 880 p. URL: <https://www.mheducation.com/highered/product/software-engineering-practitioner-s-approach-pressman-maxim/M9781260548008.html> (дата звернення: 25.05.2026).

24. Silberschatz A., Korth H. F., Sudarshan S. Database System Concepts. 7th ed. New York : McGraw-Hill, 2020. 1376 p. URL: <https://www.db-book.com/> (дата звернення: 25.05.2026).

25. Tanenbaum A. S., Wetherall D. J. Computer Networks. 6th ed. Boston : Pearson, 2021. 960 p. URL: <https://www.pearson.com/en-us/subject-catalog/p/computer-networks/P200000003146> (дата звернення: 25.05.2026).

26. Goodchild M. F. Geographic information systems and science: today and tomorrow // Annals of GIS. 2020. Vol. 26, № 1. P. 1–12. URL: <https://www.tandfonline.com/doi/full/10.1080/19475683.2020.1760064> (дата звернення: 25.05.2026).

27. Li S., Dragicevic S., Castro F. GIS-based systems for emergency

management and disaster monitoring // ISPRS International Journal of Geo-Information. 2021. Vol. 10, № 4. P. 215–229. URL: <https://www.mdpi.com/2220-9964/10/4/215> (дата звернення: 25.05.2026).

28. Zhang X., Wang J. Web-based geographic information systems and real-time spatial monitoring technologies // Journal of Geographical Systems. 2022. Vol. 24, № 3. P. 311–328. URL: <https://link.springer.com/article/10.1007/s10109-022-00367-4> (дата звернення: 25.05.2026).

29. Batty M. The Computable City: Histories, Technologies, Stories, Predictions. Cambridge : MIT Press, 2021. 504 p. URL: <https://mitpress.mit.edu/9780262045034/the-computable-city/> (дата звернення: 25.05.2026).

30. Chang K.-T. Introduction to Geographic Information Systems. 9th ed. New York : McGraw-Hill Education, 2020. 480 p. URL: <https://www.mheducation.com/highered/product/introduction-geographic-information-systems-chang/M9781260571068.html> (дата звернення: 25.05.2026).

ДОДАТОК А

UML-ДІАГРАМИ СИСТЕМИ

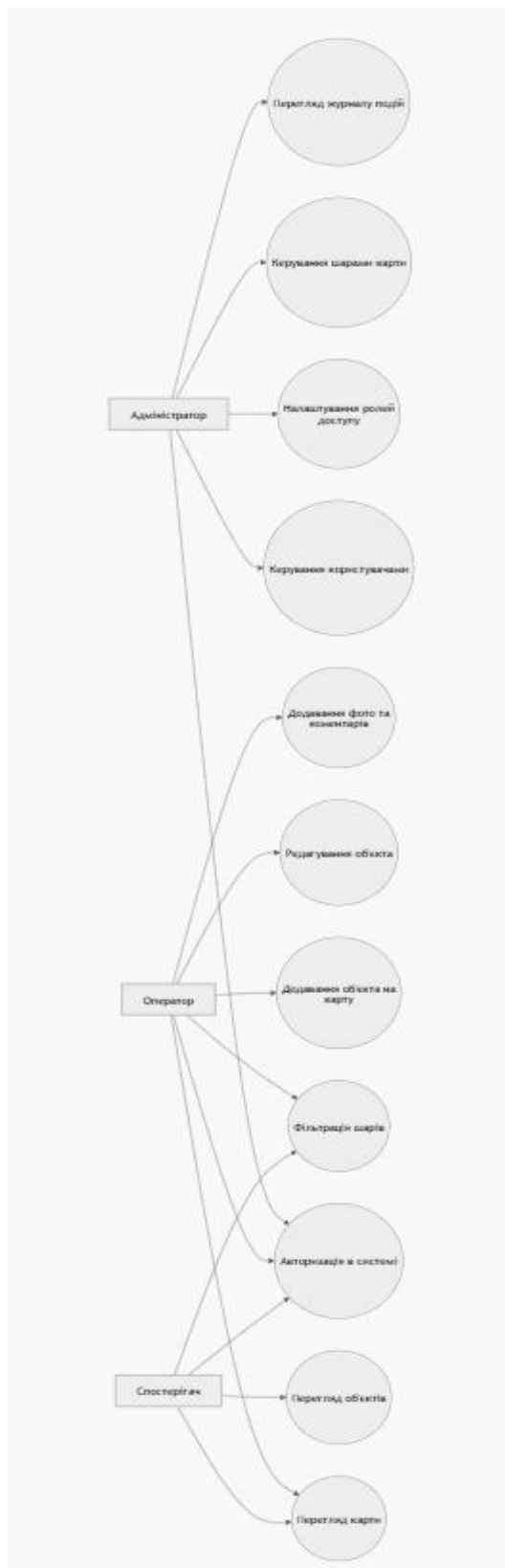


Рисунок А.1 – UML-діаграми системи

ДОДАТОК Б

СТРУКТУРА БАЗИ ДАНИХ

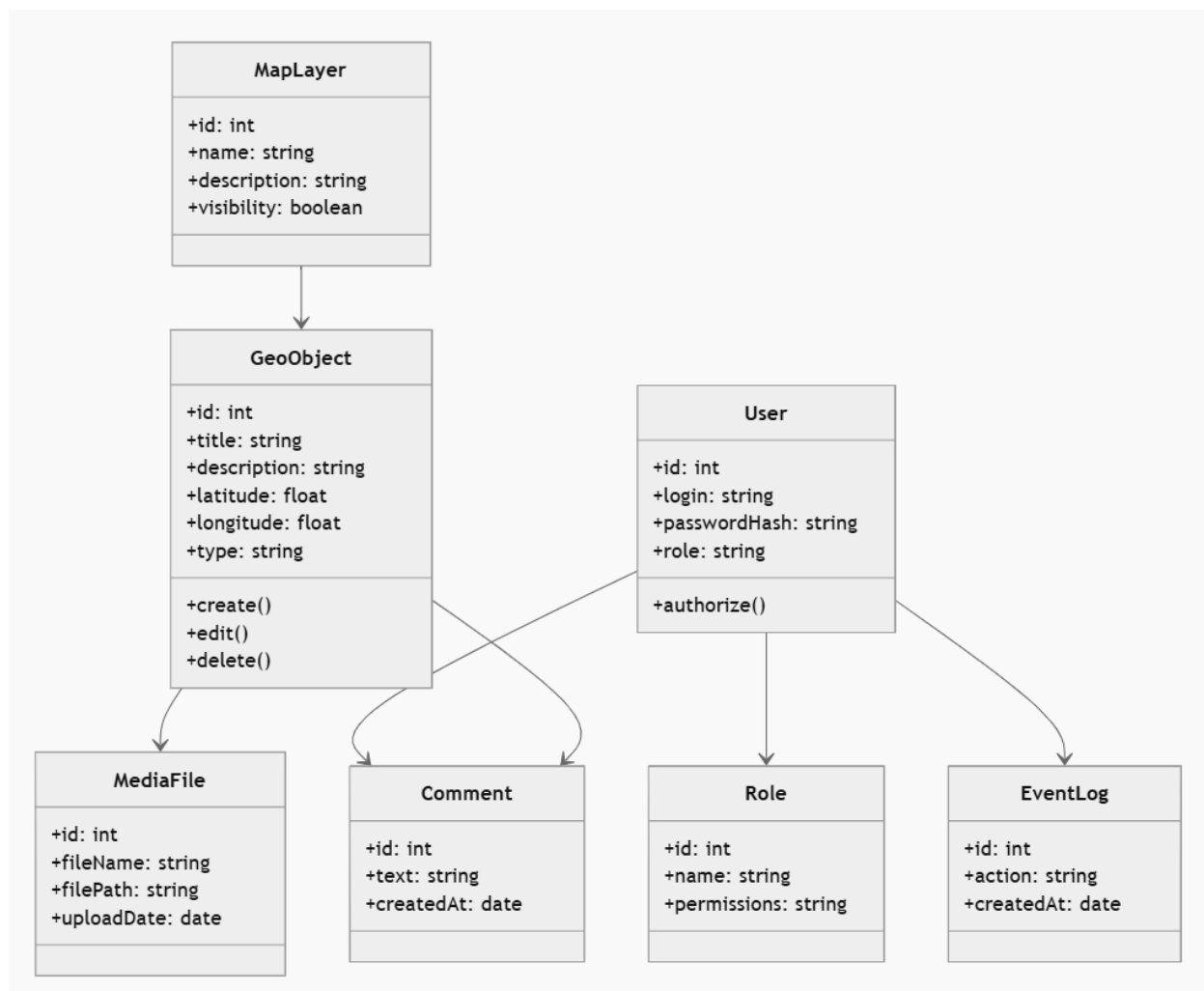


Рисунок Б.1 – Структура бази даних

ДОДАТОК В

ФРАГМЕНТИ ПРОГРАМНОГО КОДУ

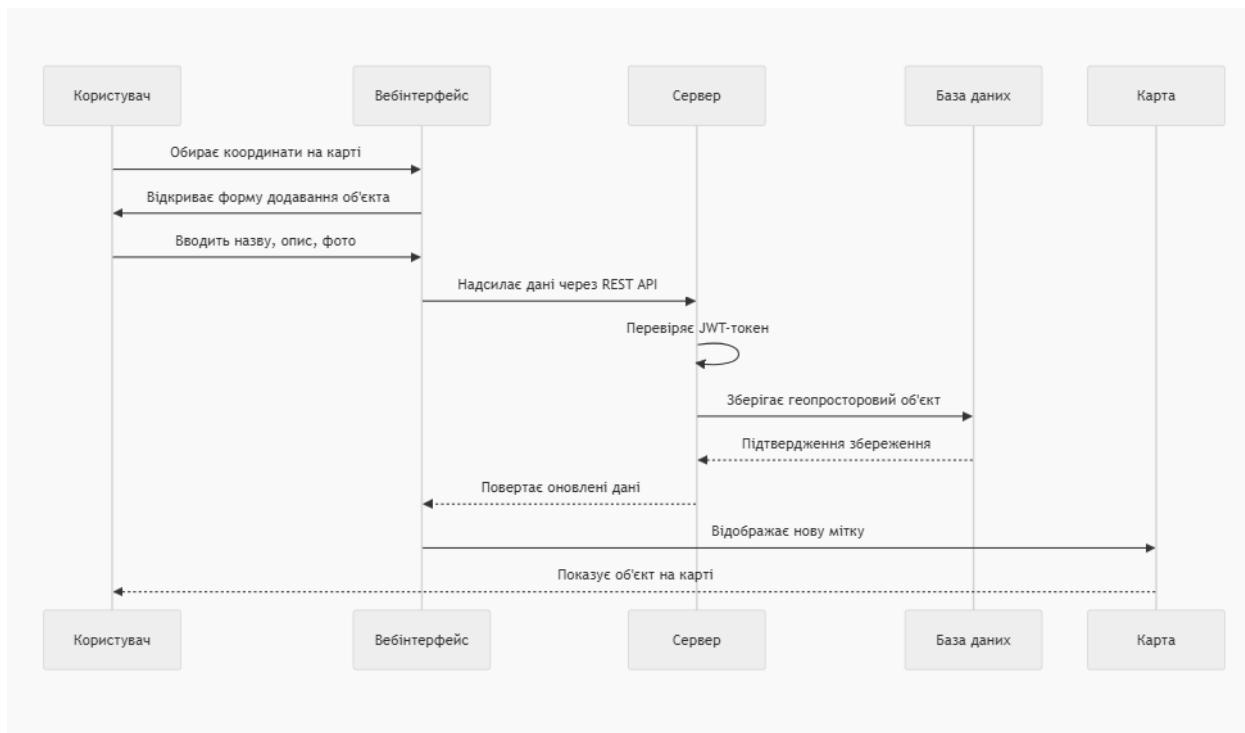


Рисунок В.1 – Фрагменти програмного коду

ДОДАТОК Г

ІНТЕРФЕЙС КОРИСТУВАЧА СИСТЕМИ

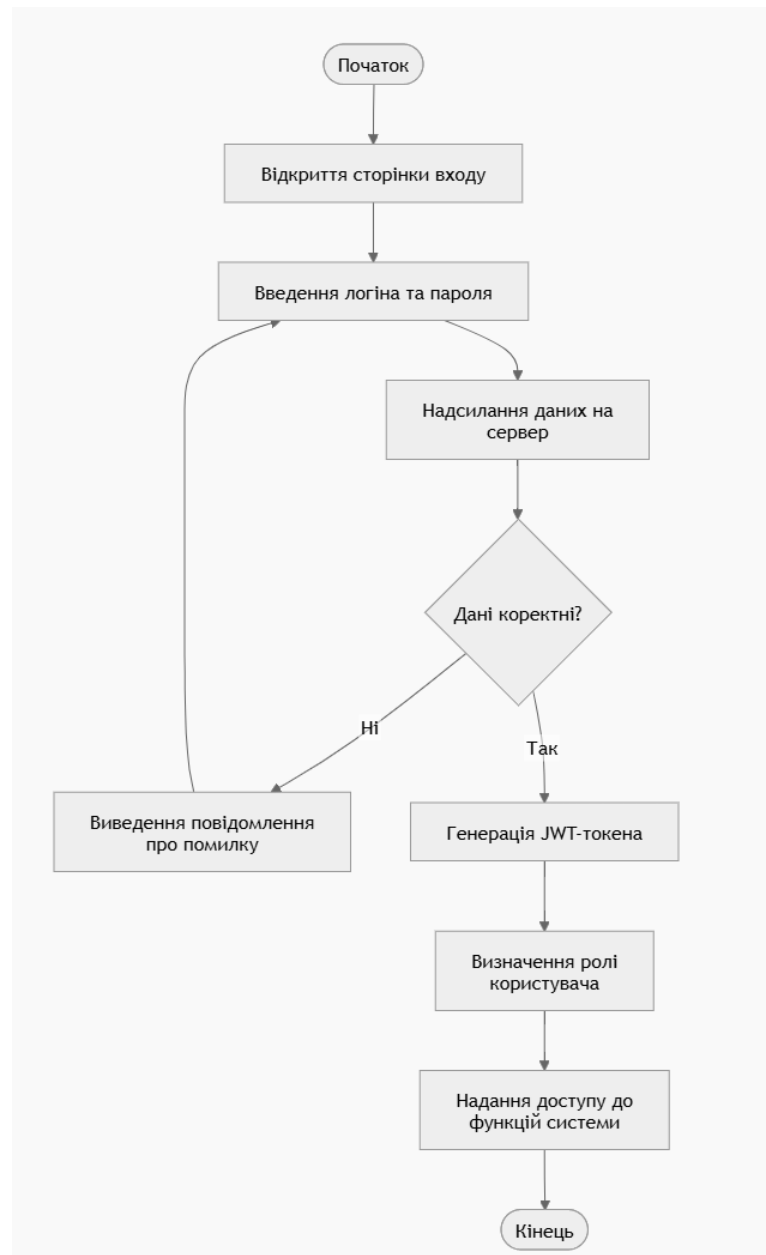


Рисунок Г.1 – Інтерфейс користувача системи

ДОДАТОК Д СЛАЙДИ ПРЕЗЕНТАЦІЇ

01

Розробка апаратно-програмного комплексу моніторингу та ситуаційної обізнаності на основі геоінформаційних технологій

Національний університет «Запорізька політехніка»
Факультет комп'ютерних наук і технологій
Кафедра комп'ютерних систем та мереж

Виконавець: Обрезков Антон Олександрович
Керівник: к.т.н., доцент Тягунова М.Ю.
2026 рік

Рисунок Д.1 – Слайд 1

02

Цілі і завдання

Цілі роботи	Ключові завдання
Цілі роботи Розробка апаратно-програмного комплексу моніторингу та ситуаційної обізнаності на основі ГІС із шарами, об'єктами та керуванням доступом.	1 проаналізувати існуючі ГІС-платформи 2 визначити вимоги до комплексу 3 спроектувати архітектуру системи 4 реалізувати вебплатформу 5 провести тестування
Об'єкт Процеси моніторингу, збору та обробки просторових даних у геоінформаційних системах.	
Предмет Методи та засоби розробки комп'ютерних АПК ситуаційної обізнаності із використанням вебтехнологій.	

Рисунок Д.2 – Слайд 2

Актуальність розробки

03

Потреба — швидко отримувати, перевіряти та поширювати просторову інформацію

Оперативність

Збір, обробка та візуалізація геоданих у режимі реального часу.

Закритий доступ

Ролі користувачів, авторизація та контроль доступу до шарів.

Ситуаційна картина

Єдине середовище для об'єктів, подій, коментарів і вкладень.

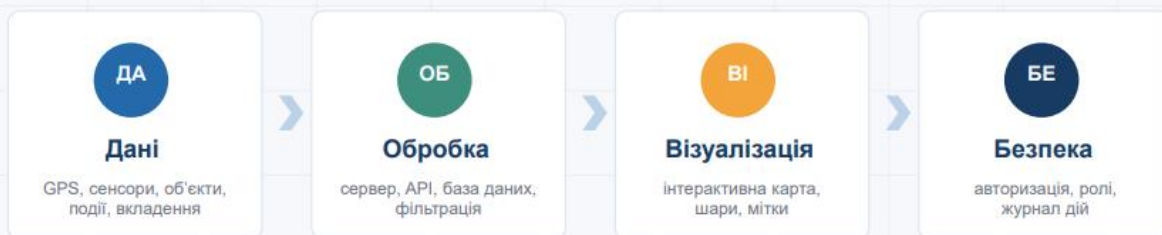
Результат: підвищення ефективності прийняття рішень та рівня ситуаційної обізнаності.

Рисунок Д.3 – Слайд 3

Основні складові системи моніторингу

04

Система поєднує дані, карту, користувацьку взаємодію та захист



У дипломі ці елементи подані як складові систем моніторингу та ситуаційної обізнаності: серверна частина, база даних, користувацький інтерфейс, авторизація, інформаційні шари, об'єкти, мультимедійні вкладки та засоби безпеки.

На основі табл. 1.1 дипломної роботи

Рисунок Д.4 – Слайд 4

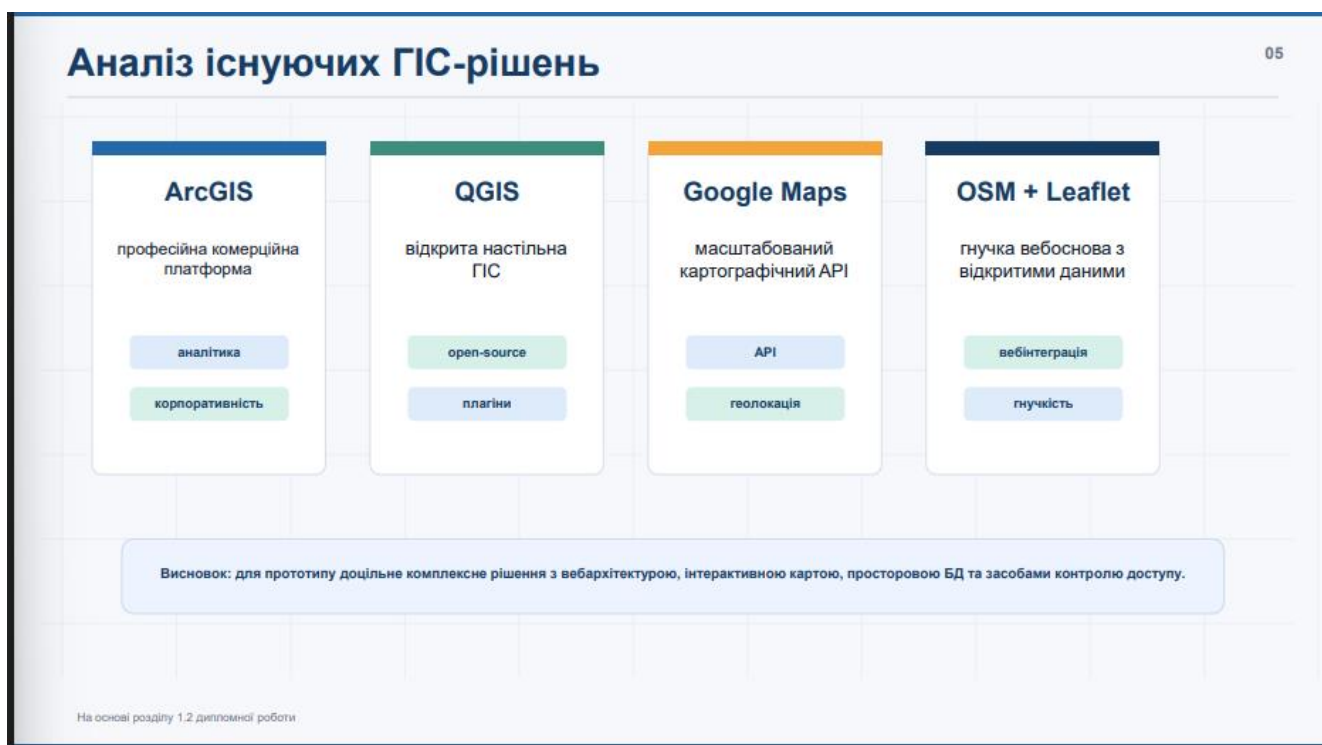


Рисунок Д.5 – Слайд 5



Рисунок Д.6 – Слайд 6

Архітектура АПК моніторингу

07

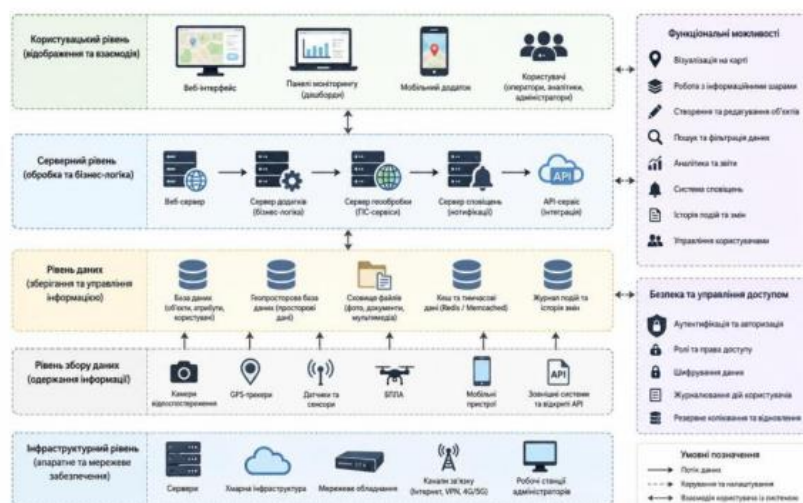


Рисунок 1.1 – Архітектура АПК моніторингу

Ключова ідея

Багаторівнева структура: інфраструктура, збір даних, сховище, серверна логіка та користувацький рівень.

Переваги

Модульність, масштабованість, можливість інтеграції нових джерел даних і сервісів.

Роль ГІС

Карта є центральним інтерфейсом для аналізу подій, шарів і об'єктів.

Рисунок Д.7 – Слайд 7

UML-модель взаємодії компонентів

08

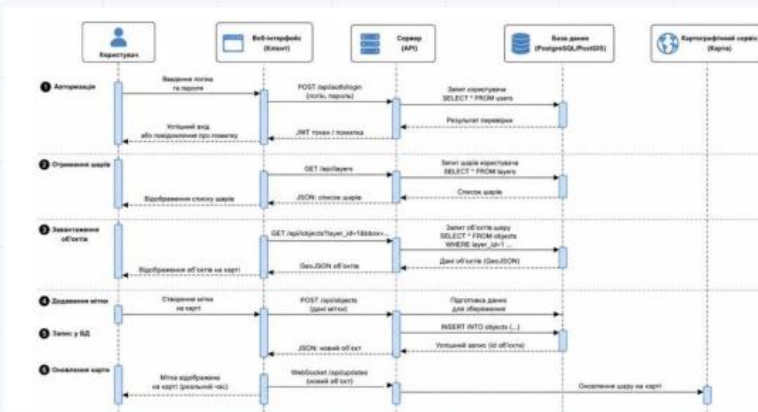


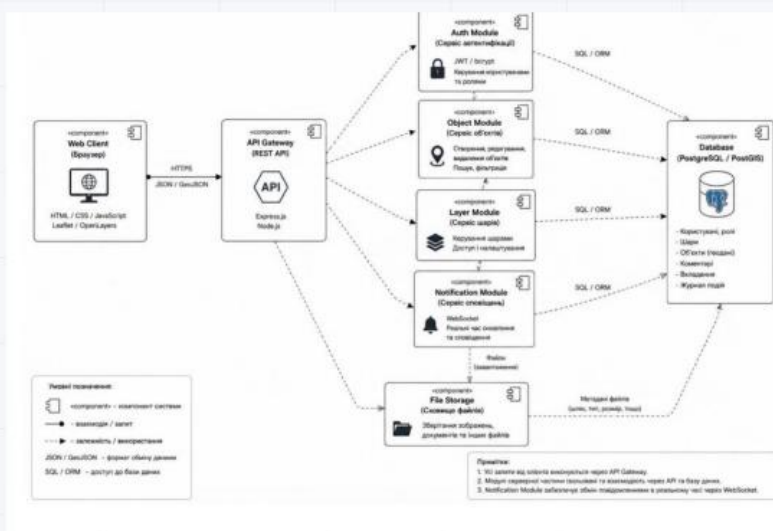
Схема показує, як запит користувача проходить через вебінтерфейс, сервер, базу даних та картографічний сервіс.

Рисунок 2.1 – UML-діаграма взаємодії компонентів системи

Рисунок Д.8 – Слайд 8

Діаграма компонентів та серверне API

09



Основні маршрути API

POST /api/auth/login

GET /api/layers

GET /api/objects

POST /api/objects

PUT /api/objects/id

DELETE /api/objects/id

API зв'язує клієнт, модулі авторизації, об'єкти, шари, повідомлення, файлове сховище та PostgreSQL/PostGIS.

Рисунок 2.2 – Діаграма компонентів

Рисунок Д.9 – Слайд 9

Структура бази даних

10

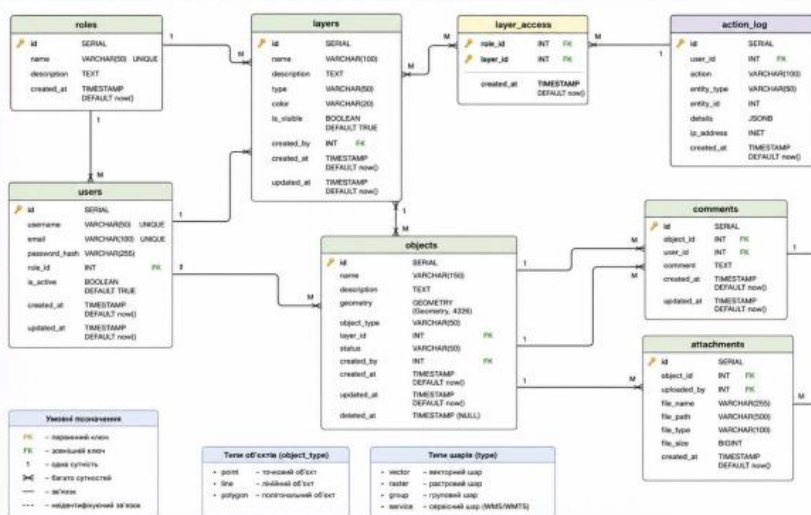


Рисунок 2.3 – ER-діаграма

Сутності

users, roles, layers, layer_access, objects, comments, attachments, action_log.

Зв'язки

Об'єкт належить шару; коментарі та вкладення пов'язані з конкретним об'єктом.

Цілісність

Зовнішні ключі, обмеження та журнал подій підтримують контроль змін.

Рисунок Д.10 – Слайд 10



Рисунок Д.11 – Слайд 11



Рисунок Д.12 – Слайд 12



Рисунок Д.13 – Слайд 13

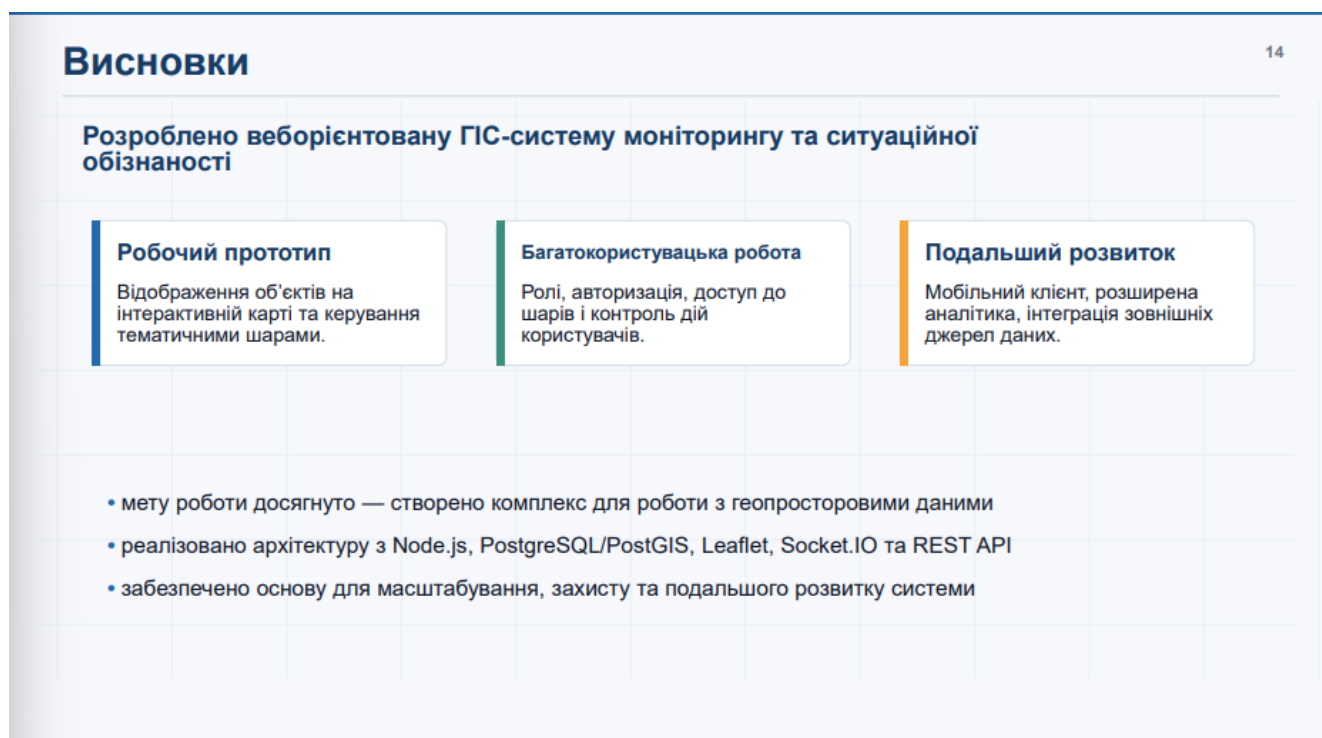


Рисунок Д.14 – Слайд 14