ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «МОДЕЛЬ БАЗИ ДАНИХ ДЛЯ ВІДОБРАЖЕННЯ МІСЦЕ РОЗТАШУВАННЯ ВАНТАЖУ»

на здобуття освітнього ступеня бакалавра
зі спеціальності 126 Інформаційні системи та технології
освітньо-професійної програми Інформаційні системи та технології

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Єгор НІКІФОРОВ

(підпис) Ім'я, ПРІЗВИЩЕ здобувача

Виконав:
здобувач вищої освіти
група ІСД-42 Єгор НІКІФОРОВ

Керівник:
науковий ступінь, вчене звання Віктор САГАЙДАК
доктор філософії

Рецензент:
науковий ступінь, вчене звання _____
Ім'я, ПРІЗВИЩЕ

Київ – 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інформаційних систем та технологій

Ступінь вищої освіти Бакалавр

Спеціальність Інформаційні системи та технології

Освітньо-професійна програма Інформаційні системи та технології

ЗАТВЕРДЖУЮ
Завідувач кафедрою ІСТ

_____ Каміла СТОРЧАК
«_____» _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Нікіфоров Єгор Ігорови
(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: Модель бази даних для відображення місцерозташування вантажу

керівник кваліфікаційної роботи Сагайдак Віктор, доктор філософії,
(Ім'я, ПРІЗВИЩЕ науковий ступінь, вчене звання)
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «___» 02.2025р. №145

2. Строк подання кваліфікаційної роботи «24» травня 2025 р.

3. Вихідні дані до кваліфікаційної роботи: науково-технічна література, структура логістичної мережі, системи GPS-моніторингу та їх API для інтеграції з БД.

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

Аналіз предметної області та постановка задачі

Огляд існуючих рішень та створення вимог до системи

Проєктування бази даних: концептуальна модель, логічна модельта, фізична модель

5. Перелік графічного матеріалу: *презентація*

1. Теоретична частина
2. Логістична структура
3. Проектування БД

6. Дата видачі завдання «27» лютого 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	02.03-05.03.25	
2	Аналіз наукової та технічної літератури з питань теми кваліфікаційної роботи.	03.03-15.03.25	
3	Розробка концептуальної моделі бази даних.	15.03-21.03.25	
4	Розробка логічної та фізичної моделей бази даних.	21.03-27.03.25	
5	Інтеграція бази даних із системами моніторингу.	27.03-07.04.25	
6	Тестування та налагодження системи.	07.04-25.04.25	
7	Оформлення роботи: вступ, висновки, реферат	25.04-05.05.25	
8	Розробка демонстраційних матеріалів	05.05-12.05.25	

Здобувач вищої освіти _____ Єгор НІКІФОРОВ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник
кваліфікаційної роботи _____ Віктор САГАЙДАК
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина бакалаврської роботи:60 сторінок, 30 рисунків, 30 джерел.

Об'єкт дослідження – інформаційна система моніторингу вантажів.

Предмет дослідження – методи та моделі управління базами даних для зберігання та обробки даних про місцезнаходження вантажів.

Мета роботи – розробка моделі бази даних для ефективного відображення та управління інформацією про переміщення вантажів.

Методи дослідження:

1. Аналіз літературних джерел та нормативних документів – вивчення міжнародних стандартів з управління логістичними даними, а також сучасних підходів до проектування та використання баз даних у логістиці.
2. Методи якісного та кількісного аналізу – вивчення моделей оптимізації баз даних для зберігання та обробки інформації про місцезнаходження вантажу, оцінка ефективності різних схем структуризації даних.
3. Емпіричні методи – аналіз реальних кейсів впровадження інформаційних систем моніторингу вантажів у логістичних компаніях.
4. Методи моделювання – використання програмних засобів для розробки концептуальної, логічної та фізичної моделей бази даних, а також тестування її працездатності.'
5. Порівняльний аналіз – зіставлення ефективності різних методів проектування та організації баз даних для забезпечення швидкого та точного відображення місцезнаходження вантажу.

Галузь використання – інформаційні системи та технології.

БАЗА ДАНИХ, ЛОГІСТИКА, GPS-МОНІТОРИНГ, ГЕОЛОКАЦІЯ, СИСТЕМА УПРАВЛІННЯ ПЕРЕВЕЗЕННЯМИ, МОДЕЛЮВАННЯ, ІНФОРМАЦІЙНА СИСТЕМА

ABSTRACT

Textual Part of the Bachelor's Thesis: 60 pages, 30 figures, 30 reference.

Object of Research – cargo monitoring information system.

Subject of Research – methods and models of database management for storing and processing data on cargo location.

Objective of the Thesis – to develop a database model for efficient representation and management of information on cargo movement.

Research Methods:

1. Analysis of literary sources and regulatory documents – study of international standards for logistics data management and modern approaches to database design and use in logistics.
2. Qualitative and quantitative analysis methods – investigation of database optimization models for storing and processing cargo location data, evaluation of the efficiency of various data structuring schemes.
3. Empirical methods – analysis of real-life cases of implementing cargo monitoring information systems in logistics companies.
4. Modeling methods – use of software tools to develop conceptual, logical, and physical models of the database, as well as testing its functionality.
5. Comparative analysis – comparison of the effectiveness of various database design and organization methods to ensure fast and accurate display of cargo location.

Field of Application – information systems and technologies.

Keywords: DATABASE, LOGISTICS, GPS MONITORING, GEOLOCATION, TRANSPORTATION MANAGEMENT SYSTEM, MODELING, INFORMATION SYSTEM

ЗМІСТ

ВСТУП

Актуальність дослідження. У сучасному світі логістика є важливою складовою економічної діяльності, і ефективне управління перевезеннями відіграє ключову роль у зниженні витрат, покращенні сервісу та мінімізації ризиків. Одним із головних аспектів сучасної логістики є контроль місцезнаходження вантажу в реальному часі. Відсутність автоматизованих систем моніторингу може призводити до втрат, затримок у доставці та фінансових збитків. Використання інформаційних систем для відстеження вантажів дозволяє підвищити прозорість логістичних процесів, покращити управління транспортними потоками та оптимізувати маршрути перевезень. Оскільки логістичні компанії працюють у конкурентному середовищі, впровадження інформаційних систем моніторингу стає важливим для їх успішної діяльності.

Об'єкт дослідження – інформаційна система моніторингу вантажів.

Предмет дослідження – методи та моделі управління базами даних для зберігання та обробки даних про місцезнаходження вантажів.

Мета роботи – розробка моделі бази даних для ефективного відображення та управління інформацією про переміщення вантажів.

Наукові завдання:

– Дослідити проблеми управління перевезеннями та відстеження вантажу у логістичних компаніях.

– Проаналізувати сучасні інформаційні системи, що використовуються для моніторингу вантажоперевезень.

– Розробити концептуальну, логічну та фізичну моделі бази даних для системи моніторингу.

– Визначити методи зберігання та обробки інформації про переміщення вантажів.

– Інтегрувати базу даних із системами GPS-моніторингу та картографічними сервісами.

– Виконати тестування працездатності системи та оцінити її ефективність у логістичних процесах.

– Запропонувати рекомендації щодо подальшого вдосконалення системи для підвищення її продуктивності та надійності.

Методи дослідження:

1. Аналіз літературних джерел та нормативних документів – вивчення стандартів управління логістикою, сучасних підходів до проєктування баз даних у сфері транспорту та моніторингу перевезень.

2. Методи якісного та кількісного аналізу – вивчення моделей оптимізації баз даних для обробки та відображення інформації про місцезнаходження вантажу.

3. Емпіричні методи – аналіз реальних кейсів впровадження систем моніторингу вантажів у логістичних компаніях.

4. Методи моделювання – використання програмних засобів для створення концептуальної, логічної та фізичної моделей бази даних.

5. Порівняльний аналіз – зіставлення ефективності різних методів проєктування та організації баз даних у логістиці.

Практичне значення одержаних результатів: Розроблена модель бази даних дозволить ефективно управляти інформацією про місцезнаходження вантажів, що сприятиме підвищенню ефективності логістичних процесів. Інтеграція бази даних із GPS-моніторингом та картографічними сервісами дасть змогу отримувати актуальну інформацію про статус і розташування вантажів у реальному часі. Запропоновані рішення можуть бути використані логістичними компаніями для оптимізації маршрутів перевезень, зниження витрат та підвищення рівня обслуговування клієнтів.

Апробація:

VI Міжнародну науково-технічну конференцію «Сучасний стан та перспективи розвитку IoT», 15 травня 2025 року, ДУІКТ – «Модель бд для відображення місцерозташування вантажу»

РОЗДІЛ 1. АНАЛІЗ І ПОСТАНОВКА ЗАДАЧІ

1.1 Мета та функціональні вимоги інформаційної системи

Метою дипломної роботи є розробка моделі бази даних, яка дозволяє ефективно зберігати, обробляти та візуалізувати інформацію про місцезнаходження вантажів у режимі реального часу. Система має забезпечувати надійне зберігання маршрутів перевезення, оперативне оновлення координат GPS, а також інтеграцію з картографічними сервісами для зручного відображення шляху переміщення вантажу. Особлива увага приділяється структурі даних, їх оптимізації для швидкої вибірки, а також безпеці доступу до інформації залежно від ролі користувача [1].

Корисність розробленої системи полягає в наступному:

– оперативний контроль вантажу – дозволяє в реальному часі відстежувати місцезнаходження вантажу, що знижує ризики його втрати або крадіжки;

– оптимізація логістичних процесів – завдяки зібраній інформації можна аналізувати маршрути та оптимізувати перевезення, зменшуючи витрати пального та часу;

– підвищення прозорості та звітності – система забезпечує збереження історії переміщень вантажів, що спрощує аналітику та звітність для компаній;

– інтеграція з іншими сервісами – підтримка GPS-моніторингу та API дозволяє з’єднати систему з мобільними додатками, CRM, ERP або картографічними сервісами;

– масштабованість і адаптивність – система побудована таким чином, що її можна розширювати під потреби різних компаній (від малого бізнесу до великої логістичної корпорації);

– підвищення якості обслуговування клієнтів – клієнти можуть отримувати повідомлення про статус доставки, що підвищує довіру до логістичного сервісу.

Для відображення місцерозташування вантажу, потрібно зробити:

– підключити систему до джерела GPS-даних (трекер або API від логістичної платформи);

– створити базу даних для зберігання координат та супровідної інформації;

– розробити механізм періодичного оновлення місцезнаходження у базі;

– реалізувати серверну логіку для обробки запитів (REST API) ;

– додати інтеграцію з картографічним сервісом (Google Maps, OpenStreetMap) ;

– створити інтерфейс для відображення маркера з координатами вантажу;

– реалізувати авторизацію користувачів для захисту даних;

– забезпечити фільтрацію та сортування вантажів за статусом, датою, маршрутом;

– додати систему сповіщень у разі відхилення від маршруту або зупинки;

– протестувати систему на надійність, точність та швидкодію.

1.2 Логістика та її проблеми

Логістика – це наука, практика та система організації процесів транспортування, зберігання, обліку і управління матеріальними та інформаційними потоками. Вона охоплює повний цикл – від постачання сировини до доставки готової продукції кінцевому споживачеві. Однією з головних проблем є відсутність своєчасного та достовірного інформування про місцезнаходження вантажів, що ускладнює контроль за переміщенням та створює ризики затримок. Затримки у доставці, своєю чергою, часто спричиняються не лише неефективним плануванням маршрутів, а й браком взаємодії між логістичними ланками та недостатнім використанням сучасних IT-рішень. Додаткову загрозу становлять втрати та пошкодження вантажів під час

перевезення, що може бути наслідком людського фактору, технічних збоїв або поганої координації. Багато підприємств досі користуються паперовими носіями або застарілими системами управління, що знижує рівень автоматизації процесів та не дозволяє оперативно реагувати на зміни. Ускладнює ситуацію також відсутність прозорості логістичних операцій, що не дозволяє керівництву компаній отримувати повну картину поточних процесів і приймати виважені управлінські рішення [2]. Найпоширеніші з перелічених проблем узагальнені у таблиці 1.1.

Таблиця 1.1

Основні проблеми логістичної галузі

Проблема	Опис
Відсутність актуальної інформації	Інформація про вантаж оновлюється із запізненням або вручну, що ускладнює контроль.
Затримки у доставці	Виникають через неефективне планування маршрутів або технічні проблеми.
Втрати та пошкодження вантажів	Через відсутність моніторингу вантажі можуть бути втрачені або пошкоджені.
Низький рівень автоматизації	Використання застарілих методів управління знижує ефективність.
Непрозорість логістичних процесів	Слабка інтеграція систем ускладнює аналітику та прийняття рішень.

Однією з ключових проблем сучасної логістичної галузі є відсутність або недостатній рівень цифровізації основних бізнес-процесів. У багатьох логістичних компаніях досі застосовуються застарілі підходи до обліку, планування та відстеження вантажів, що базуються на ручному введенні даних, паперових документах та ізольованих системах. Така фрагментованість ускладнює оперативне реагування на зміни в логістичному ланцюзі, призводить до затримок, дублювання інформації, а також помилок у передачі даних між учасниками логістичних процесів.

У цьому контексті важливою є повноцінна цифровізація логістики – впровадження інформаційних технологій, які дозволяють автоматизувати та

інтегрувати всі ключові етапи перевезення. Одним із центральних елементів цієї трансформації є створення єдиної централізованої бази даних для відображення місцезнаходження вантажу.

Завдяки використанню сучасних СКБД та технологій, таких як REST API, інтеграція з GPS-системами та картографічними сервісами, можна створити гнучке та масштабоване рішення, яке забезпечує повну прозорість логістичних процесів. Інформація, що зберігається в базі, є доступною в будь-який момент часу для авторизованих користувачів – диспетчерів, менеджерів, клієнтів – що значно покращує комунікацію між сторонами та дозволяє швидко приймати рішення у разі форс-мажорів [3].

Крім того, цифрова база даних дозволяє проводити глибокий аналіз ефективності логістичних операцій. Наприклад, можна виявити затримки на певних маршрутах, визначити найбільш надійних перевізників, оптимізувати схеми доставки та зменшити витрати. Згодом така система може стати основою для побудови більш складних рішень – систем прогнозування, автоматизованого планування маршрутів, інтеграції з ERP та CRM.

Таким чином, впровадження бази даних для відображення місцезнаходження вантажу є не лише актуальним рішенням для подолання кризи автоматизації в логістиці, а й стратегічним кроком до створення ефективної, надійної та сучасної транспортно-інформаційної системи. Це дозволяє компаніям підвищити конкурентоспроможність, знизити операційні витрати та забезпечити високий рівень обслуговування клієнтів.

1.3 CRM-система для логістики

Customer Relationship Management (CRM) – це система управління взаємовідносинами з клієнтами. Вона призначена для збирання, зберігання, аналізу та використання інформації про клієнтів з метою підвищення рівня обслуговування, збільшення продажів та покращення бізнес-процесів.

CRM-системи допомагають компаніям автоматизувати багато щоденних

завдань: відстеження комунікацій з клієнтами, управління угодами, створення звітів, нагадувань і навіть персоналізованого маркетингу.

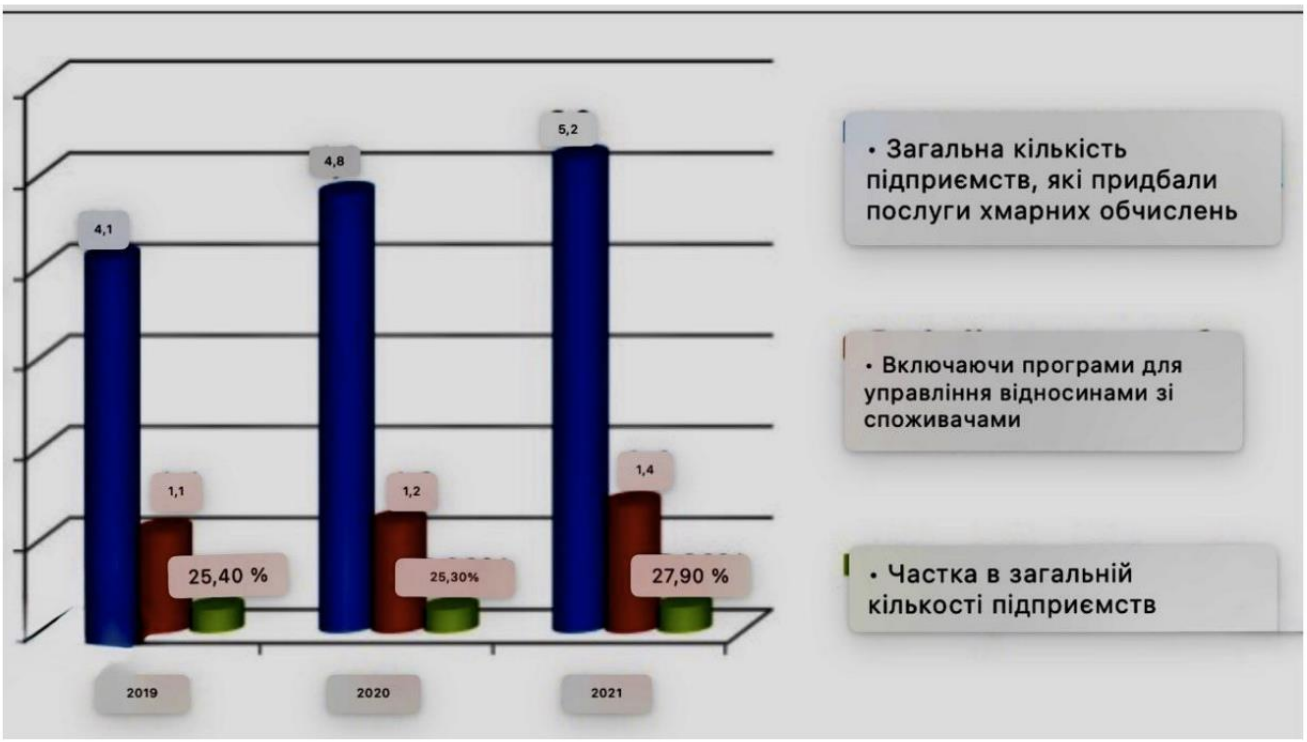

Рисунок 1.1 – Збільшення тенденції використання CRM-систем

Згідно зі статистичними даними, середньорічний темп зростання світового ринку CRM-систем становить 10,68%. У 2023 році його обсяг досяг 79,4 млрд доларів США (рис. 1.2), а до 2029 року прогнозується зростання до 131,9 млрд доларів. Це свідчить про стабільну тенденцію до впровадження CRM-технологій у все більшій кількості компаній по всьому світу.

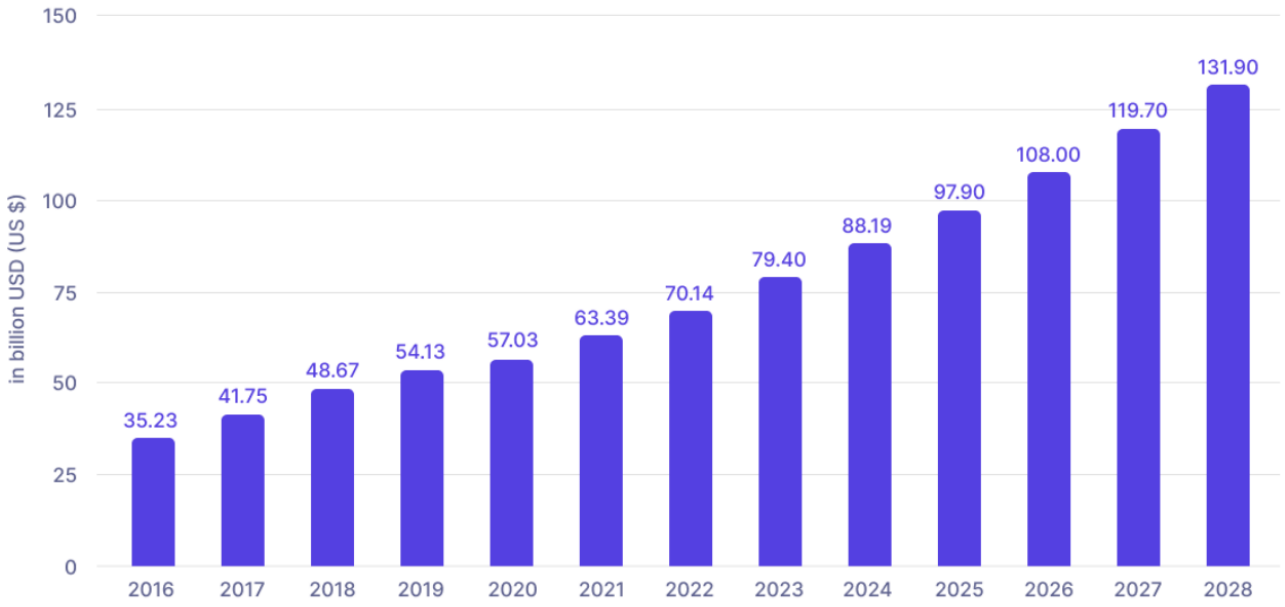

Рисунок 1.2 – Графік впровадження CRM-технологій

Дослідження методів оптимізації свідчить про високу ефективність застосування алгоритмів маршрутизації та математичного моделювання. Інтеграція інноваційних технологій, зокрема Інтернету речей (IoT) та аналізу великих даних (Big Data), забезпечує істотне підвищення продуктивності CRM-рішень у сфері логістики, сприяючи зниженню витрат та покращенню якості клієнтського сервісу.

Однією з ключових властивостей сучасних CRM-платформ є забезпечення зручної та уніфікованої взаємодії з клієнтами. Це охоплює збір, збереження та обробку інформації про клієнтів через різноманітні канали комунікації — електронну пошту, соціальні мережі, веб-інтерфейси та чат-боти. Не менш важливою функцією є управління замовленнями, яке охоплює весь їх життєвий цикл — від формування до доставки, з інтеграцією у відповідні ERP- і SCM-системи для досягнення ефективної координації бізнес-процесів [14].

CRM-системи також надають можливість здійснення моніторингу доставки в режимі реального часу, що дає змогу обом сторонам — компанії та клієнтам — оперативно відстежувати місцезнаходження і стан вантажу. Інтегровані аналітичні інструменти та засоби прогнозування дозволяють обробляти великі

обсяги даних для виявлення трендів, передбачення рівня попиту, виявлення проблем у ланцюгах постачання та пропонування ефективних рішень.

До найпоширеніших CRM-систем, які мають розширену функціональність для логістичної сфери, належать Salesforce CRM, Zoho CRM, SAP Customer Experience та Microsoft Dynamics 365. Вони пропонують модулі для автоматизації процесів, можливості інтеграції з іншими корпоративними системами, інструменти аналітики та прогнозування. Водночас підприємства стикаються з низкою викликів: висока вартість впровадження, обмежена адаптивність до специфічних потреб логістики, труднощі масштабування та потреба у додатковій підготовці персоналу.

Одним з актуальних прикладів спеціалізованої CRM-системи є Super Dispatch (рис. 1.3), яка поєднує ключові функції, необхідні для управління логістичними операціями та забезпечує зручність у щоденній роботі.

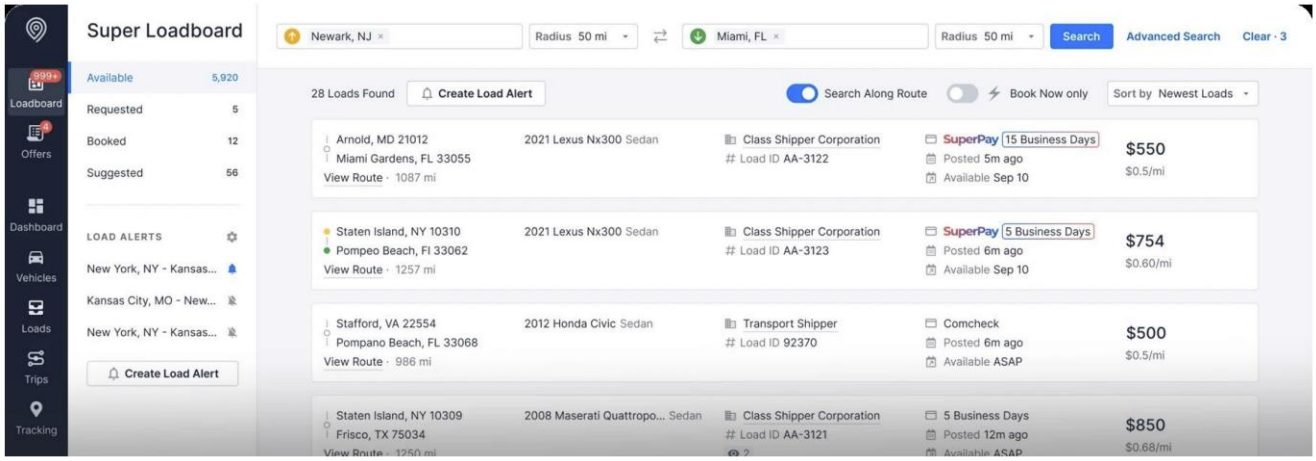

Рисунок 1.3 – Зовнішній вигляд CRM-системи Super Dispatch

У сфері логістики впровадження CRM-систем виступає одним із визначальних чинників підвищення ефективності взаємодії з клієнтами та оптимізації логістичних процесів. Такі системи забезпечують поліпшення цілої низки аспектів операційної діяльності після їх інтеграції в бізнес-середовище. Основні напрями покращення, які забезпечують CRM-рішення, відображено на рисунку 1.4.

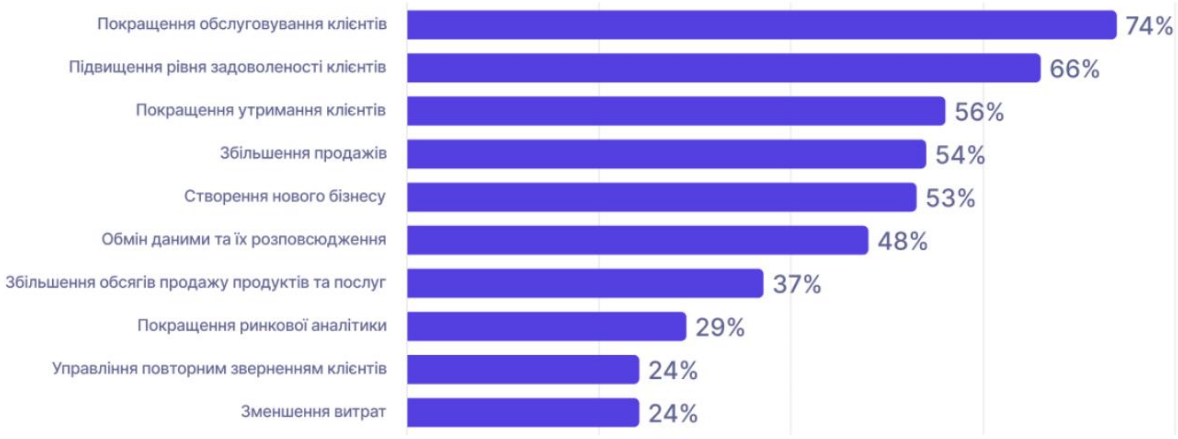

Рисунок 1.4 – Ключові переваги впровадження CRM-систем

У цьому підрозділі розглянуто функціональні можливості та результати впровадження трьох популярних CRM-рішень: *Salesforce*, *Microsoft Dynamics 365* та *Super Dispatch*.

Salesforce пропонує широкий набір інструментів для управління продажами, клієнтським сервісом і маркетингом. Гнучка архітектура цієї платформи дозволяє адаптувати її до специфічних потреб логістичних компаній, забезпечуючи легку інтеграцію з численними зовнішніми сервісами та додатками. Згідно з офіційною статистикою Salesforce, підприємства, які перейшли на цю систему з інших CRM-платформ, зафіксували:

- зростання ROI на **34%**,
- підвищення продуктивності на **30%** завдяки застосуванню інструментів штучного інтелекту,
- збільшення доходу від продажів на **32%**.

Microsoft Dynamics 365 є комплексною платформою, яка поєднує функціональність CRM та ERP. Це дає змогу логістичним компаніям централізовано керувати ланцюгами постачання, фінансами та клієнтськими процесами в межах єдиної екосистеми. Інтеграція з такими продуктами як *Office 365* та *Azure* забезпечує злагоджену взаємодію й оптимізацію робочих процесів. За відгуками компаній, які здійснили перехід з Salesforce на Dynamics 365,

спостерігається покращення у сфері управління даними та зниження витрат на впровадження.

Super Dispatch спеціалізується на автоматизації логістичних процесів у сфері автомобільних перевезень. Платформа включає функції управління замовленнями, відстеження вантажів у режимі реального часу, а також електронний документообіг. Система орієнтована на потреби перевізників і логістичних брокерів, спрощує операційну діяльність і зменшує обсяг ручної та паперової роботи. Користувачі відзначають зростання ефективності диспетчеризації та зменшення кількості помилок при обробці заявок.

Вибір CRM-рішення залежить від специфіки бізнесу:

- *Salesforce* — ідеальне рішення для компаній, що шукають гнучку, масштабовану платформу з потужною аналітикою;
- *Microsoft Dynamics 365* — оптимальний варіант для бізнесів, які прагнуть глибокої інтеграції CRM з існуючими процесами та продуктами Microsoft;
- *Super Dispatch* — спеціалізований інструмент для логістичних компаній, що здійснюють автомобільні перевезення.

У сфері логістики CRM-система може допомагати:

- координувати роботу з клієнтами (замовлення, скарги, зворотний зв’язок);
- вести історію перевезень і маршрути;
- автоматизувати виставлення рахунків та документообіг;
- інтегруватись із системами моніторингу вантажів і БД для відображення місцезнаходження [4].

CRM-система в логістичному бізнесі виконує набагато більше, ніж просто облік клієнтів. Вона стає інструментом для комплексного управління перевезеннями, дозволяючи координувати всі етапи логістичного процесу – від прийняття замовлення до його доставки. В умовах високої конкуренції на ринку транспорту та доставки, ефективна взаємодія з клієнтами, чітке планування маршрутів і швидке реагування на зміну обставин є критично важливими. І саме CRM надає необхідні інструменти для цього.

По-перше, CRM дозволяє вести централізований облік усіх замовлень на перевезення. Менеджери можуть легко переглядати статус кожного перевезення, бачити історію взаємодії з клієнтом, коментарі до замовлення, супровідні документи та терміни доставки. Це дає змогу уникнути плутанини, мінімізувати людські помилки та забезпечити прозорість процесу.

По-друге, система автоматично сповіщає клієнтів про етапи доставки: підтвердження отримання замовлення, час відправлення, поточне місцезнаходження вантажу та підтвердження доставки. Такі сповіщення підвищують довіру до компанії, зменшують кількість звернень у службу підтримки та покращують сервіс загалом.

Третім важливим аспектом є інтеграція CRM із базою даних місцезнаходження вантажів. Завдяки цьому менеджери в режимі реального часу бачать, де саме знаходиться вантаж, які можливі затримки або проблеми на маршруті. Це дозволяє швидко реагувати, перепланувати маршрути або інформувати клієнта про зміни.

CRM також значно полегшує аналітику та прогнозування. Керівництво компанії може аналізувати, які клієнти приносять найбільший дохід, які маршрути найчастіше використовуються, які проблеми виникають у процесі доставки. На основі цих даних формуються звіти, які допомагають приймати обґрунтовані управлінські рішення. Система допомагає формувати базу лояльних клієнтів. Наприклад, CRM може зберігати інформацію про особливі умови для кожного замовника, знижки, історію попередніх перевезень. Це дає можливість персоналізувати підхід і підтримувати довгострокову співпрацю [5].

Ще одна важлива перевага – CRM підтримує автоматизацію документообігу: рахунки, договори, акти виконаних робіт можуть формуватися автоматично на основі замовлень. Це суттєво економить час, знижує адміністративне навантаження та мінімізує ризики помилок у документах.CRM-система є не лише базою контактів, а й повноцінною платформою для управління перевезеннями. Її впровадження в логістичну діяльність дозволяє значно

підвищити ефективність, зменшити витрати, покращити обслуговування клієнтів і зробити бізнес більш гнучким та конкурентоспроможним.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА ТЕХНІЧНА РЕАЛІЗАЦІЯ

Проєктування інформаційної системи є ключовим етапом у створенні ефективного програмного забезпечення, що відповідає вимогам користувачів та забезпечує стабільну роботу в реальних умовах. У даному розділі здійснюється формалізація структури майбутньої системи, визначаються її основні компоненти, сценарії використання, ролі користувачів, а також особливості побудови бази даних. Технічна реалізація базується на сучасних інструментах та підходах, що дозволяють забезпечити інтеграцію з GPS-трекерами, швидку обробку даних та зручний інтерфейс для взаємодії з системою.

2.1 Користувачі та їхні ролі

Інформаційна система моніторингу вантажів розробляється з урахуванням потреб різних категорій користувачів, які взаємодіють із нею на різних рівнях доступу. Кожна роль виконує специфічні функції, спрямовані на забезпечення стабільної роботи системи, своєчасного оновлення даних, координації логістичних процесів та зручного інформування клієнтів. Успішна реалізація проєкту можлива лише за умови чіткого розподілу повноважень між учасниками, що забезпечує як ефективність управління, так і безпеку обробки інформації. Найвищий рівень доступу має адміністратор, який здійснює контроль за всіма модулями системи. Диспетчер (логіст) є основним виконавцем операційної частини – формує маршрути, слідкує за місцезнаходженням вантажів, аналізує маршрути та звіти. Клієнт отримує доступ до інформації про свої замовлення через особистий кабінет, тоді як GPS-система виступає технічним джерелом координат. Такий розподіл дозволяє забезпечити чітку структуру взаємодії та зменшує ризик помилок під час обробки даних [6]. Детальна характеристика користувачів системи та їхніх функцій представлена нижче у таблиці 2.1.

Таблиця 2.1

Користувачі системи та їхні ролі

Роль користувача	Опис	Основні функції
Адміністратор	Користувач з повним доступом до системи, відповідальний за налаштування та підтримку працездатності.	Керування обліковими записами Повний доступ до всіх функцій Моніторинг системної активності Управління даними вантажів Налаштування прав доступу
Диспетчер (логіст)	Основний оператор, що працює з інформацією про вантажі, маршрути та координує перевезення.	Додавання та редагування вантажів Формування маршрутів Перегляд карт та координат Отримання сповіщень Генерація звітів
Клієнт (замовник)	Користувач, що має обмежений доступ для перегляду інформації про свої замовлення.	Перегляд статусу доставки Перегляд місцезнаходження вантажу Отримання повідомлень Особистий кабінет
GPS-система	Зовнішній технічний модуль, що надсилає координати в реальному часі до бази даних.	Надсилання GPS-координат Оновлення місцезнаходження вантажів Взаємодія через API

Система відображення місцезнаходження вантажу будується на основі інтеграції декількох сучасних технологій позиціонування та передачі даних, зокрема GPS, Galileo та GSM. Усі вони відіграють ключову роль у процесі фіксації координат, передавання їх до бази даних і подальшої візуалізації на мапі.

Global Positioning System (GPS) – це Американська супутникова система глобального позиціонування. Забезпечує точне визначення координат у будь-якій точці планети. Є основним джерелом геоданих для більшості трекерів.

Galileo – європейська глобальна навігаційна супутникова система (GNSS), розроблена ЄС. Galileo є незалежною альтернативою GPS і ГЛОНАСС, забезпечує високу точність – до 1 метра для цивільного використання, а також кращу стабільність у міських зонах або складних географічних умовах. Сучасні GPS-трекери часто підтримують одночасне використання GPS + Galileo, що забезпечує

надійне позиціонування [7].

Global System for Mobile Communications (GSM) – використовується як транспортний канал для передачі координат від трекера до сервера. Дані надсилаються за допомогою мобільного інтернету (GPRS/3G/4G) або SMS. Завдяки GSM можливе передавання інформації з будь-якої точки, де є мобільне покриття, навіть за відсутності Wi-Fi або супутникового інтернету.

Архітектура взаємодії:

– трекер приймає координати з супутників GPS і Galileo;

– GSM-модуль передає ці координати через мобільну мережу;

– на сервері координати обробляються і зберігаються в базі даних MySQL;

– через веб-інтерфейс або мобільний додаток користувач бачить рух вантажу на інтерактивній мапі (Google Maps, OpenStreetMap).

Одним із ключових елементів системи моніторингу вантажів є взаємодія між GPS-трекером, мобільною мережею та сервером, де відбувається обробка і зберігання координат. На транспортний засіб встановлюється GPS-трекер, який отримує сигнал від супутникових систем, таких як GPS або Galileo. Далі отримані координати передаються через шлюзи (Gateways) за допомогою мобільної мережі GSM (наприклад, через GPRS або 4G) на сервер, де зберігаються в централізованій базі даних.

На серверній стороні працює програмна логіка, яка обробляє запити, перевіряє права доступу користувачів, зберігає історію переміщень вантажу та дозволяє візуалізувати його маршрут. Користувач, авторизувавшись у системі, отримує доступ до мапи, де в режимі реального часу може спостерігати за поточним розташуванням транспортного засобу. Такий підхід забезпечує повну прозорість, точність та надійність логістичних операцій, дозволяючи оперативно реагувати на будь-які зміни на маршруті [8]. Схематичне зображення архітектури системи наведено на рисунку 2.1.

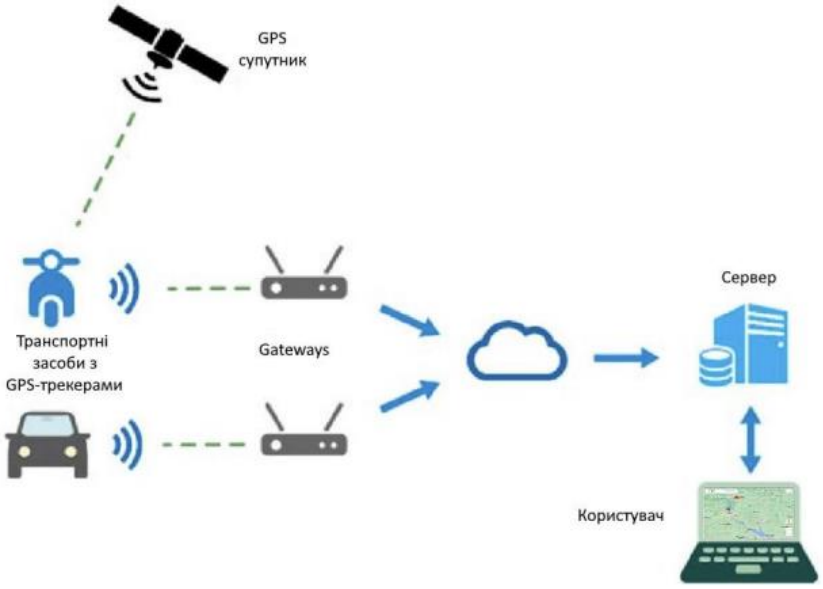

Рисунок 2.1 – Схема взаємодії транспортного засобу з GPS-трекером, шлюзом та сервером для відображення місцезнаходження вантажу

2.2 Функціональна модель системи (UML-діаграми, сценарії використання)

У межах роботи з інформаційною системою користувачі виконують низку основних дій, які забезпечують повноцінне управління процесом моніторингу вантажів. Однією з базових функцій є перегляд мапи, де в режимі реального часу відображається поточне місцезнаходження транспортного засобу з GPS-трекером. Користувач також має змогу додавати нові вантажі до системи, вказуючи ключові характеристики: назву, вагу, пункт відправлення та призначення, відповідальну особу тощо [9]. Окремий функціональний блок відповідає за моніторинг маршруту, що дозволяє відстежити рух вантажу, аналізувати відхилення від заданого шляху та отримувати повідомлення про затримки або зупинки. Крім того, система надає можливість перегляду історії переміщень, що особливо корисно для звітності, аналітики та виявлення логістичних «вузьких місць». Для адміністративних ролей доступні також дії з керування користувачами, налаштуванням прав доступу та аналізом журналів активності.

На рисунку 2.2 представлено основні сценарії взаємодії користувачів із системою.

Користувач (диспетчер або логіст) виконує ключові дії: переглядає карту з місцезнаходженням вантажів, аналізує маршрути, перевіряє координати, працює зі списком вантажів та отримує сповіщення про події (затримка, відхилення, прибуття).

Адміністратор має доступ до адміністративних функцій – керує обліковими записами, налаштовує доступ, контролює системну активність та має повний контроль над інформацією.

GPS-модуль (зовнішня система) автоматично надсилає координати у базу даних у режимі реального часу, забезпечуючи актуальність геопозиційних даних.

Кожна роль у системі реалізована з урахуванням принципів безпеки та ефективності, а логіка взаємодії відображена у вигляді діаграми варіантів використання (Use Case Diagram).

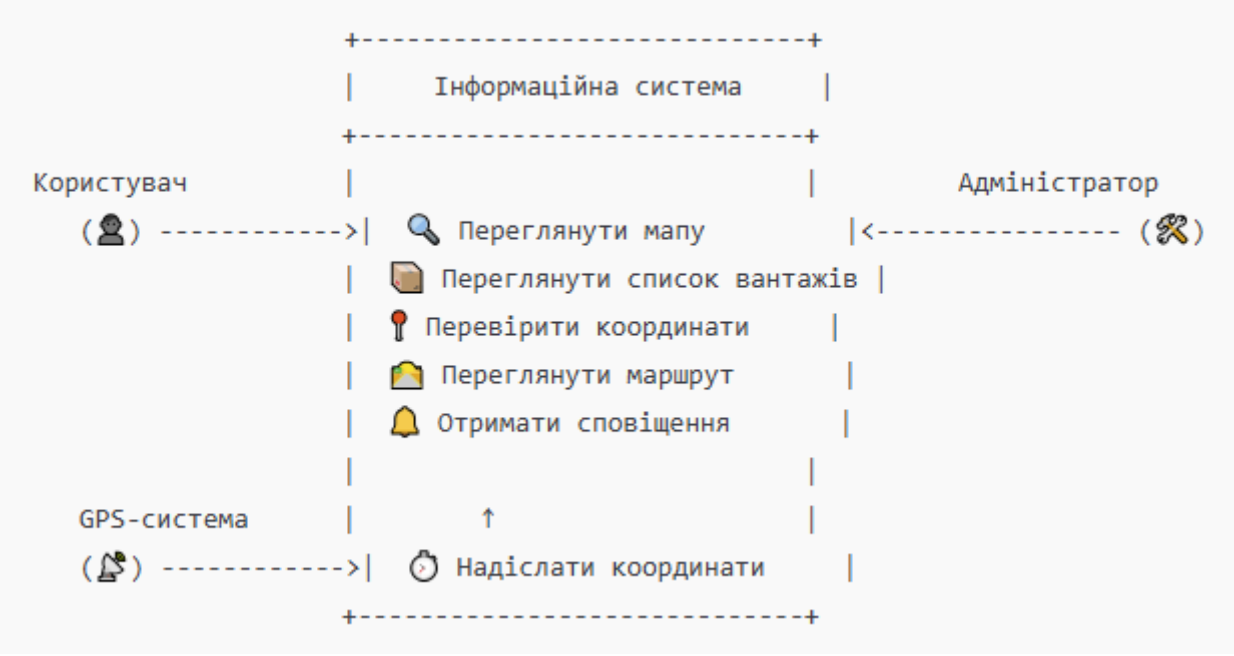

Рисунок 2.2 – UML-діаграма варіантів використання системи моніторингу вантажів

На рисунку 2.3 зображено інтерфейс форми авторизації, яка є початковим етапом взаємодії користувача з системою. Панель входу реалізована у вигляді

простої та інтуїтивно зрозумілої форми, що містить два поля: логін і пароль, а також кнопку «Вхід». Після введення облікових даних користувач проходить процедуру автентифікації

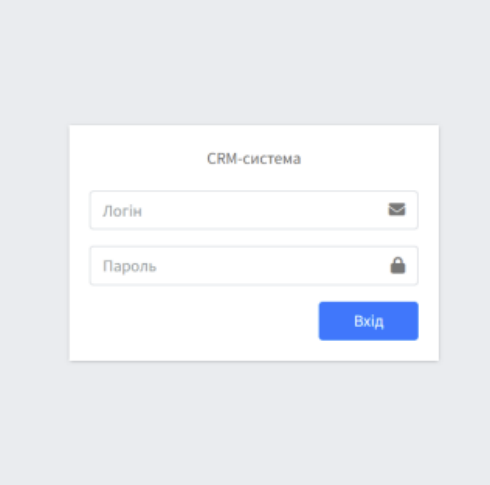

Рисунок 2.3 – Форма авторизації користувача

На рисунку 2.4 зображено інтерфейс авторизації для замовника (клієнта), який взаємодіє з CRM-системою для оформлення та відстеження своїх вантажних перевезень. Інтерфейс побудований у вигляді модального вікна з двома основними полями: Email і Пароль, що дозволяє користувачу швидко і безпечно увійти до особистого кабінету. Форма також містить посилання для реєстрації нових користувачів, що дозволяє клієнтам самостійно створювати облікові записи.

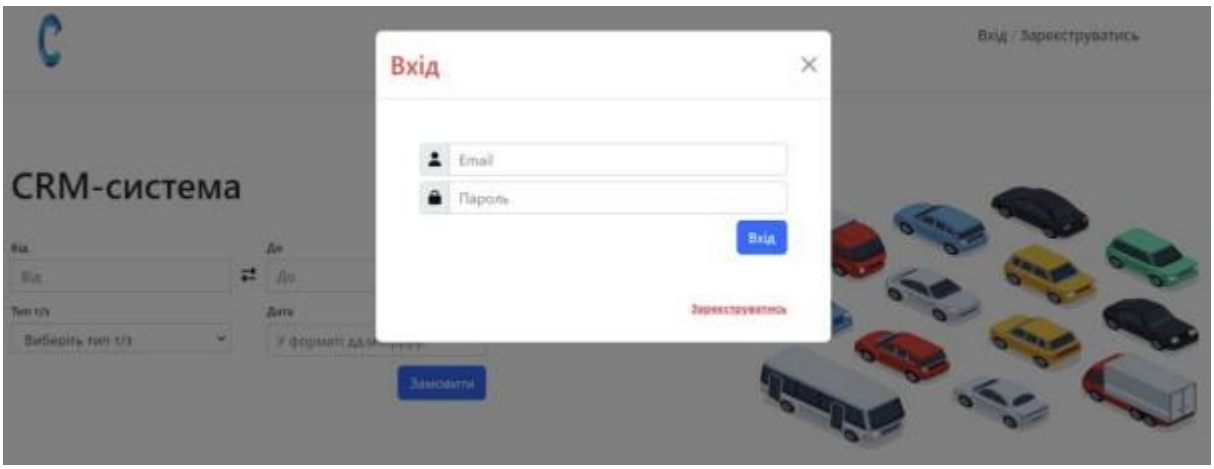

Рисунок 2.4 – Панель для входу замовника

На рисунку 2.5 представлено форму реєстрації замовника, яка реалізована у вигляді модального вікна поверх головного інтерфейсу CRM-системи. Форма містить обов’язкові поля для введення імені, номера телефону, email-адреси, пароля та адреси користувача. Кожне поле супроводжується відповідною іконкою, що підвищує зручність використання та інтуїтивність взаємодії. Кнопка «Зареєструватись» активує запит на створення нового облікового запису, після чого дані передаються на сервер для збереження в базі даних.

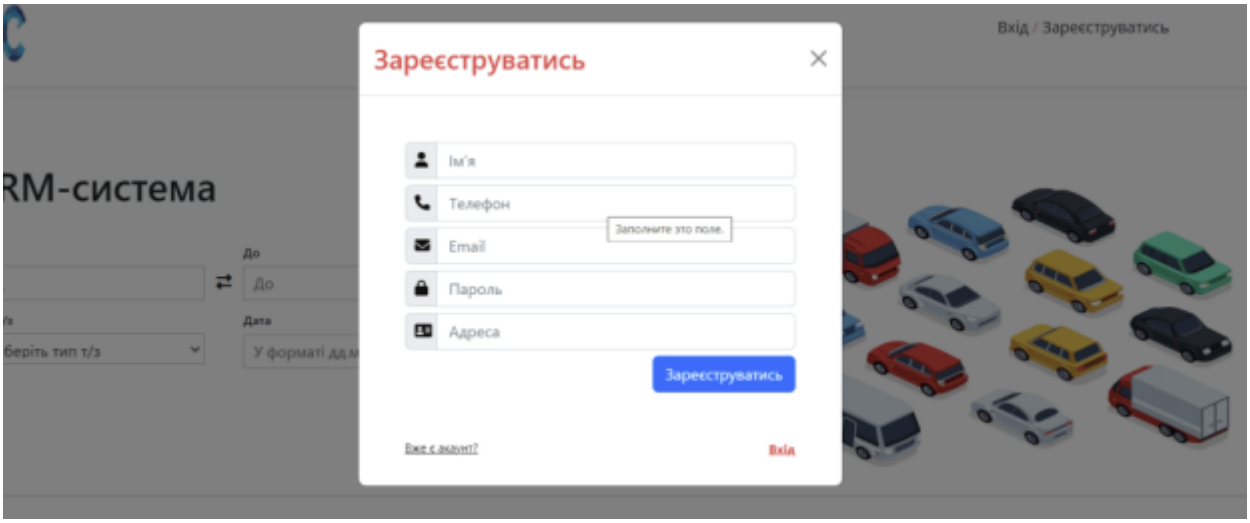

Рисунок 2.5 – Форма реєстрації замовника в CRM-системі

На рисунку 2.6 зображено інтерфейс адміністративної панелі для створення нового користувача в системі. Дана форма доступна лише для користувачів з роллю адміністратора і дозволяє вводити основні реєстраційні дані, такі як ім’я, email, роль, пароль, а також додаткову інформацію про статус користувача. Нижче розміщений блок налаштування прав доступу, де адміністратор може вибірково надати або обмежити доступ до певних функцій системи (створення, редагування, перегляд, видалення). Ролі розділені за категоріями (транспорт, вантажі, клієнти, замовлення, повідомлення тощо), що забезпечує гнучке та точне адміністрування доступу.

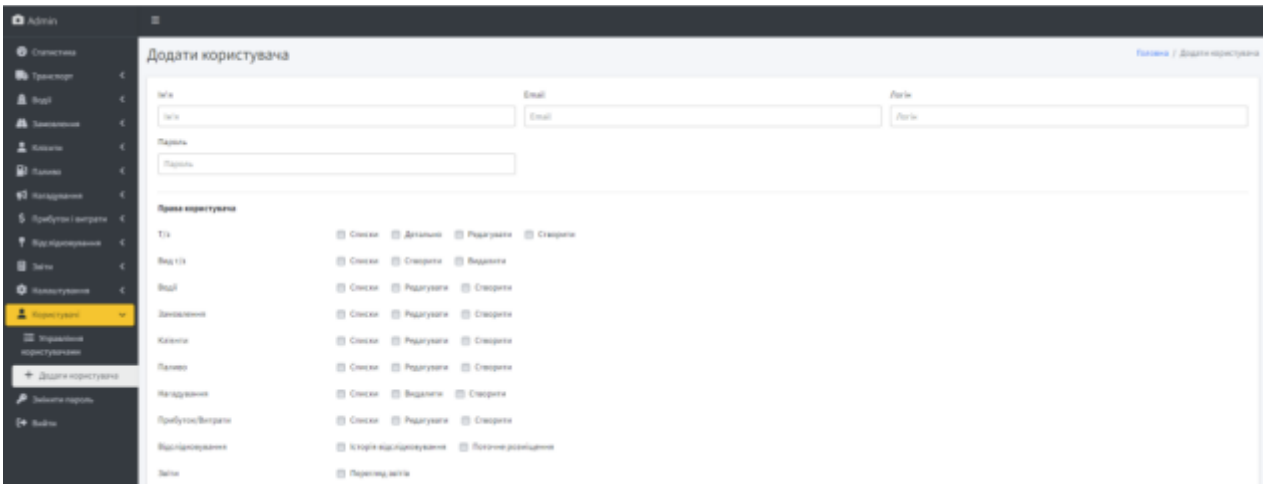

Рисунок 2.6 – Форма для створення користувача в адміністративній панелі

Окрім операційної частини, система також містить функціонал для візуального відображення аналітичних даних та ключових показників. Інтерфейс адміністративної панелі містить модуль статистики, який дозволяє у зручному форматі переглядати загальний стан системи, кількість активних користувачів, вантажів, замовлень та відправлень. Окремі блоки відображають поточні місцезнаходження вантажів, статус транспортних засобів, а також графіки, що демонструють динаміку втрат і прибутків. Така інформаційна панель дозволяє адміністраторам та диспетчерам приймати оперативні рішення, відстежувати ефективність логістичних процесів та виявляти відхилення або проблеми у роботі. Загальний вигляд цього модуля наведено на рисунку 2.7.

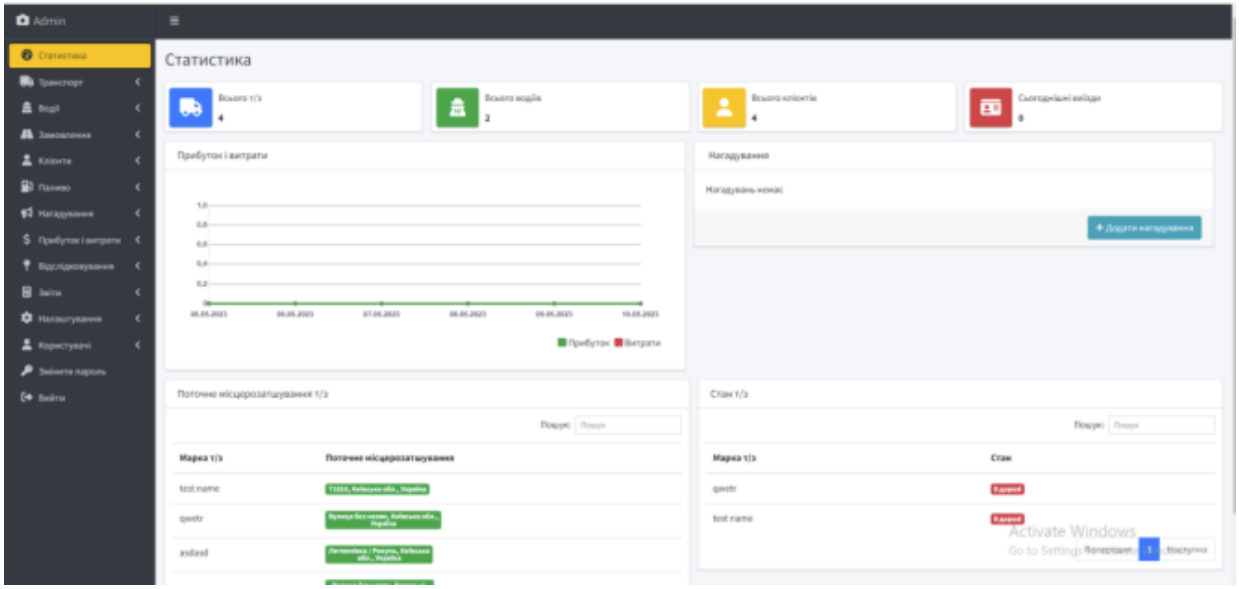

Рисунок 2.7 – Інтерфейс статистичної панелі адміністратора

Система також забезпечує зручний функціонал для ведення обліку транспортних засобів, які беруть участь у перевезеннях. В адміністративній панелі реалізовано окрему форму для додавання нового ТЗ, де користувач із відповідними правами може ввести детальну інформацію: державний номер, VIN-код, марку, модель, витрату пального, об’єм паливного бака, а також тип і вид транспортного засобу [10]. Окреме поле відповідає за вибір кольору, що дозволяє відображати ТЗ на карті з візуальним маркером. Передбачено також введення терміну дії реєстрації, що дозволяє вчасно контролювати адміністративні строки. Така форма спрощує управління транспортним парком компанії та підвищує точність обліку. Інтерфейс цього модуля наведено на рисунку 2.8.

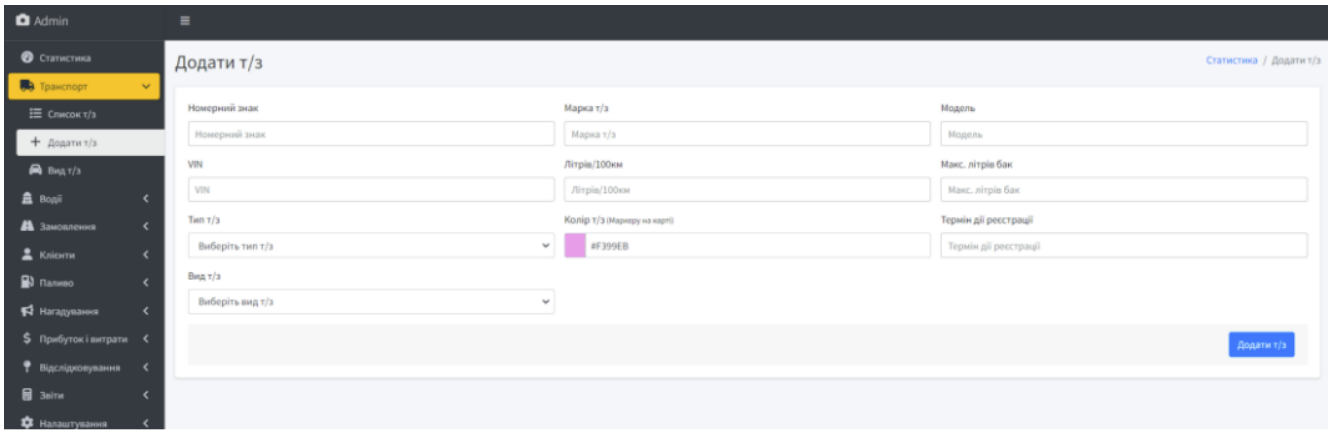

Рисунок 2.8 – Форма додавання транспортного засобу в систему

Для ефективного управління автопарком система надає модуль «Список транспорту», який дозволяє переглядати, редагувати та контролювати всі наявні транспортні засоби компанії. Таблиця відображає ключові характеристики кожної одиниці транспорту, зокрема: номер у списку, марку, модель, реєстраційний номер, VIN-код, тип/вид транспорту, а також статус активності. Для кожного рядка доступні кнопки швидкого редагування, перегляду або видалення запису. Функція пошуку у правому верхньому куті дає змогу швидко знайти потрібний транспорт за будь-яким із параметрів. Подібна структура дозволяє підтримувати актуальність даних, забезпечує швидкий доступ до інформації та зручне адміністрування. Вигляд модуля представлено на рисунку 2.9.

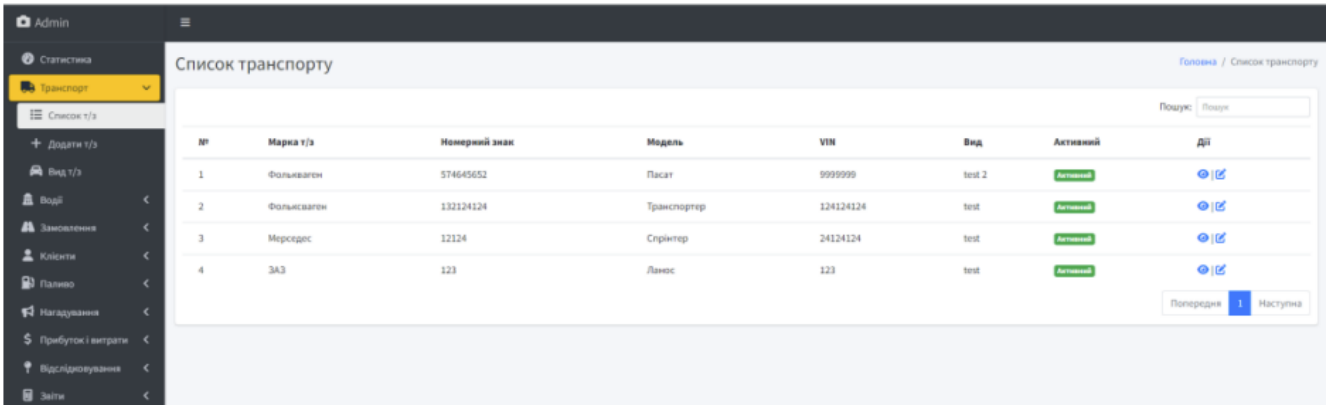

Рисунок 2.9 – Список транспортних засобів у системі

Для кращого розуміння логіки взаємодії користувача з інформаційною системою доцільно подати послідовність основних дій у вигляді діаграми діяльності. Вона ілюструє типовий сценарій роботи з системою моніторингу вантажів: від моменту входу до аналізу даних про доставку. Даний візуальний елемент дозволяє побачити, які етапи проходить користувач, які модулі системи залучені до процесу та як реалізовано зв’язок між інтерфейсом і базою координат.

На рисунку 2.10 наведено приклад діаграми діяльності користувача без стрілок напрямку, що акцентує увагу на самих діях, а не їхній послідовності.

Рисунок 2.10 – Діаграма діяльності користувача без позначення напрямку дій

Адміністратор – відповідає за загальне управління системою, налаштування доступу та контроль за її роботою. Адміністратор виконує стратегічні та технічні дії, пов’язані з керуванням обліковими записами, додаванням транспортних засобів, аналізом статистики та формуванням звітності. Саме ця роль забезпечує цілісність і безпеку всієї інформаційної системи.

На рисунку 2.11 наведено послідовність основних дій адміністратора у вигляді діаграми діяльності без позначення напрямку, що дозволяє акцентувати увагу на ключових функціях, виконуваних на різних рівнях доступу.

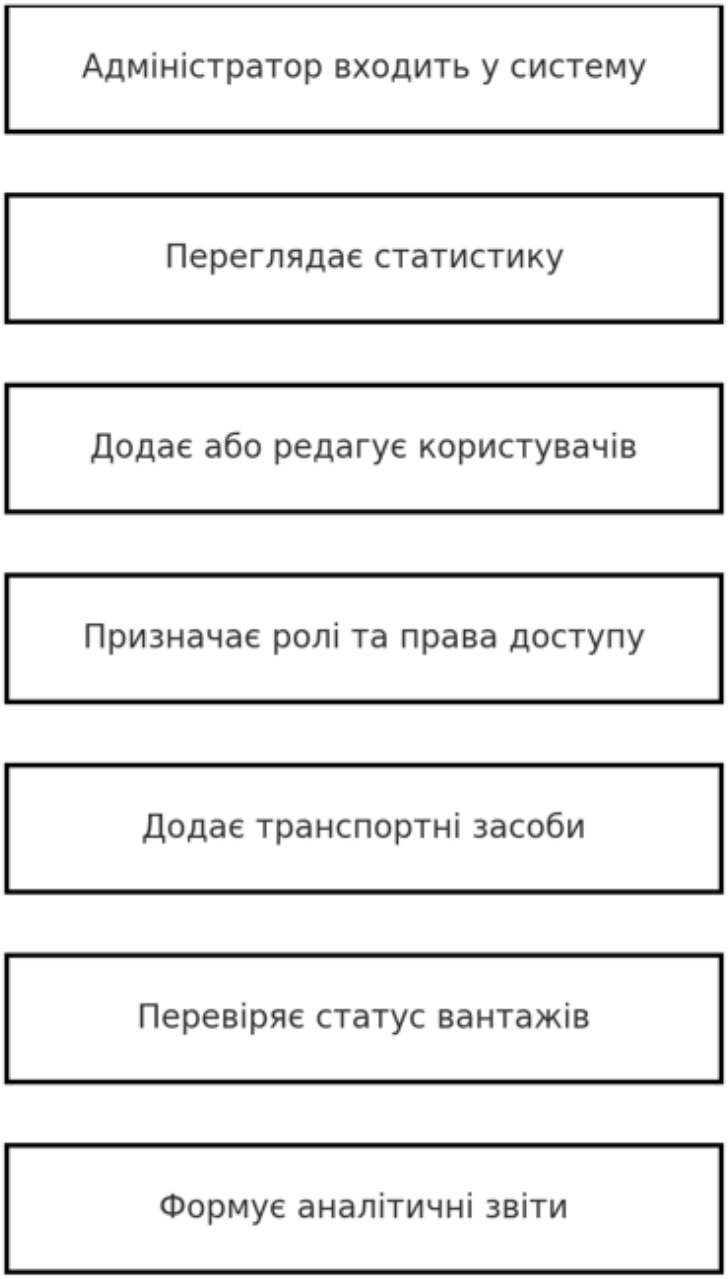

Рисунок 2.11 – Діаграма діяльності адміністратора в системі

2.3 Архітектура та структура бази даних (MySQL, зв’язки між сутностями)

Однією з ключових функцій системи є візуалізація маршруту вантажу на карті. Завдяки інтеграції з картографічними сервісами, такими як Google Maps або OpenStreetMap, користувач отримує можливість переглядати переміщення транспортного засобу в режимі реального часу [11]. У верхній частині інтерфейсу

реалізовано фільтри за датою, реєстраційним номером ТЗ або ідентифікатором вантажу, що дозволяє швидко знайти потрібну інформацію. Маршрут на мапі будується на основі координат, що надходять від GPS-модуля, і відображається у вигляді зв'язаної лінії між точками. Такий підхід дозволяє не лише бачити поточне місцезнаходження вантажу, а й аналізувати пройдений шлях, виявляти затримки, зупинки або відхилення від заданого маршруту. Інтерфейс модуля наведено на рисунку 2.12.

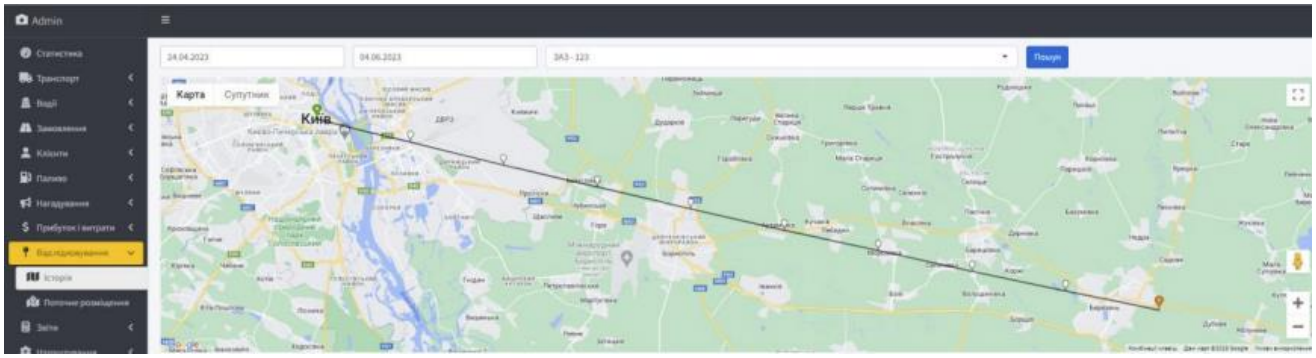

Рисунок 2.12 – Відображення маршруту вантажу на карті в інтерфейсі системи

Відображення поточних позицій усіх активних транспортних засобів у системі. За допомогою інтеграції з картографічним сервісом користувач отримує візуальний огляд місцезнаходження машин на великій мапі. Кожен транспортний засіб позначається відповідним іконками, які можуть відрізнятись за кольором чи типом залежно від транспортного засобу або стану доставки [12]. Це дозволяє диспетчеру швидко оцінити географічне розташування всіх вантажів та приймати управлінські рішення в реальному часі. Такий функціонал особливо корисний при координації великого автопарку, моніторингу міжнародних перевезень або виявленні затримок. Інтерфейс цієї функції наведено на рисунку 2.13.

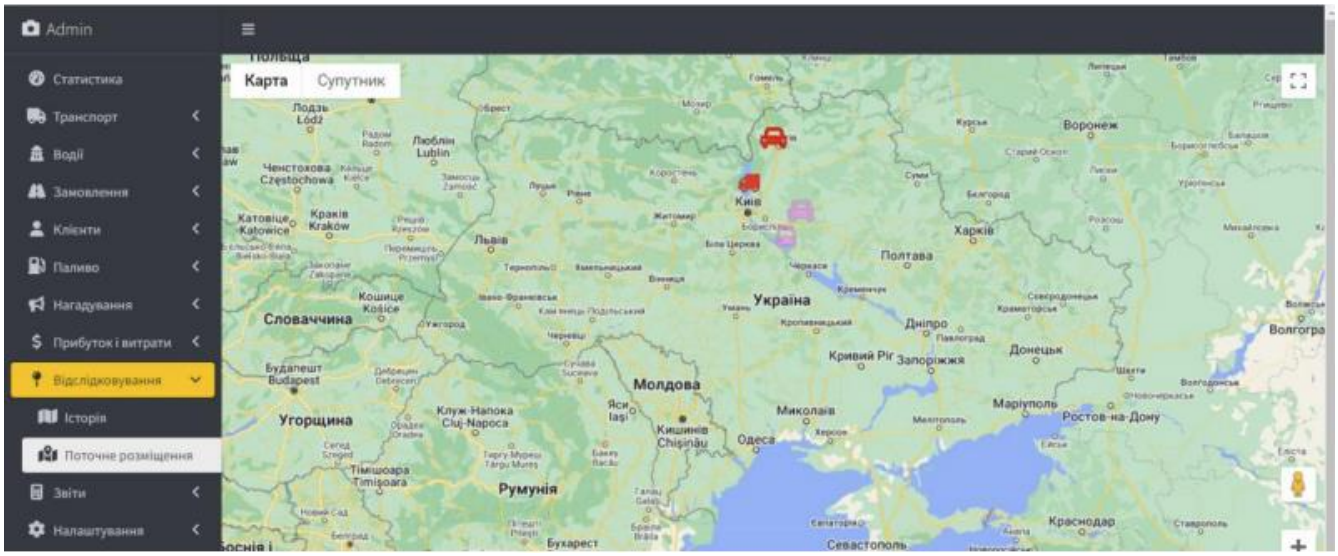

Рисунок 2.13 – Поточне розміщення транспортних засобів на карті

Система також підтримує побудову розширених маршрутів міжнародних перевезень з відображенням ключових зупинок, часу в дорозі та супутньої інформації. Такий функціонал дозволяє не лише спостерігати за поточним переміщенням, а й аналізувати маршрутні дані в цілому: загальний час відправлення, кількість зупинок, тривалість поїздки та відстань [13]. Додаткові кнопки у верхній частині інтерфейсу дозволяють керувати маршрутом: відмічати початкову та кінцеву точки, вказувати зупинки, прогнозувати витрати пального та навіть переглядати маршрут через поштові сервіси.

Це особливо важливо у разі логістики між країнами, де потрібен контроль часу, безпеки, перетину кордонів і чітке планування ресурсу. Приклад такого перегляду наведено на рисунку 2.14.

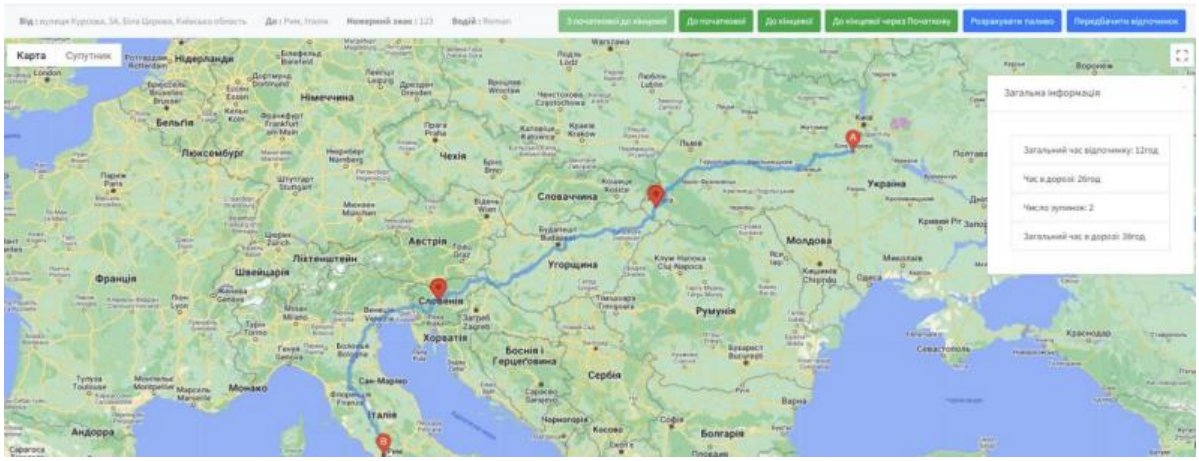

Рисунок 2.14 – Побудова маршруту міжнародного перевезення з аналітикою руху

У рамках системи реалізовано корисну функцію — розрахунок витрат палива на основі заданої кількості літрів. Відповідне модальне вікно з'являється при натисканні на кнопку в інтерфейсі маршруту, де користувач може ввести поточну кількість пального в баку. Після натискання кнопки «Вирахувати» система виконує обчислення на основі заздалегідь заданого середнього споживання пального для кожного транспортного засобу [14]. Це дозволяє швидко оцінити, чи вистачить пального для проходження маршруту, та у разі потреби запланувати дозаправку. Такий функціонал особливо корисний для планування міжнародних рейсів або далеких перевезень, де немає змоги постійно оновлювати інформацію вручну. Інтерфейс розрахунку наведено на рисунку 2.15.

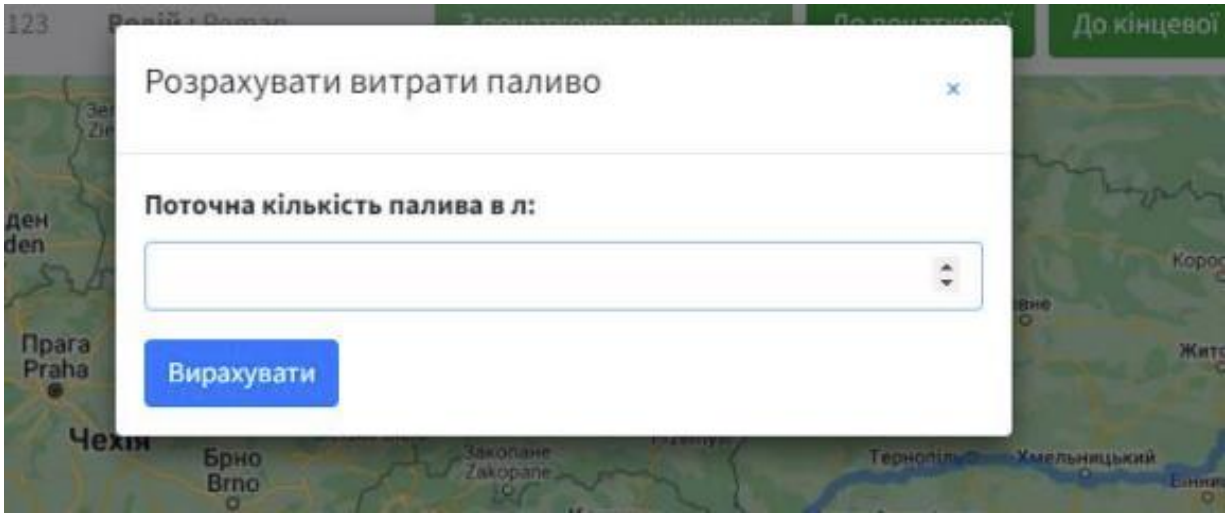

Рисунок 2.15 – Інтерфейс розрахунку витрат пального в системі

Ще однією важливою функцією системи є розрахунок часу відпочинку водія, що відповідає вимогам безпеки та міжнародним транспортним регламентам. У спеціальному модальному вікні користувач має можливість вказати тривалість відпочинку та загальний час у дорозі. Після введення значень система автоматично розраховує, коли водій має зробити паузу у відповідності до норм (наприклад, не більше 9 годин безперервного керування). Це дозволяє не лише дотримуватися правил дорожнього руху, а й забезпечити безпечне планування перевезень, зниження навантаження на водія та уникнення штрафів під час перевірок. Інтерфейс цього модуля показано на рисунку 2.16.

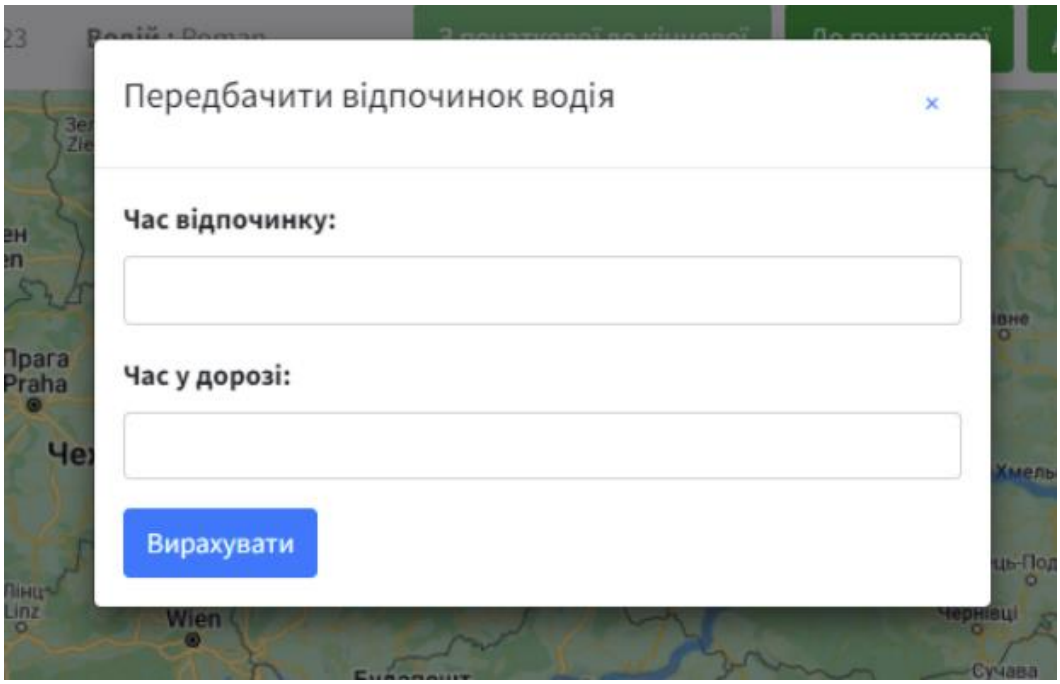

Рисунок 2.16 – Інтерфейс розрахунку часу відпочинку водія в системі

У системі передбачено інтерактивну зміну маршруту перевезення безпосередньо на мапі. Користувач може в режимі реального часу змінювати траєкторію руху, перетягуючи маршрут мишкою, що дає змогу швидко адаптувати логістику до нових умов – заторів, ДТП, змін у графіку тощо. Це особливо корисно для диспетчерів, які координують великі перевезення або потребують терміново оновити маршрут. Додаткові кнопки дозволяють повернути маршрут до початкового або кінцевого варіанту. Такий функціонал

значно підвищує гнучкість системи та знижує ризики при транспортуванні. Приклад інтерфейсу для зміни маршруту наведено на рисунку 2.17.

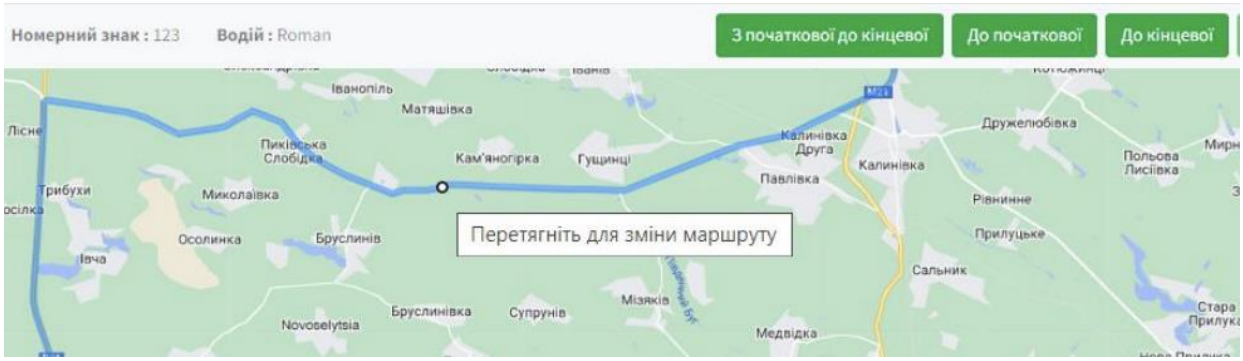

Рисунок 2.17 – Інтерактивне редагування маршруту на мапі в режимі онлайн

Для реалізації системи зберігання та обробки даних у проєкті було обрано MySQL – одну з найпоширеніших реляційних систем управління базами даних (СУБД) з відкритим вихідним кодом. Такий вибір обумовлений низкою переваг, що ідеально відповідають вимогам даної інформаційної системи:

– стабільність та надійність, MySQL зарекомендувала себе як перевірена часом платформа, що демонструє стабільну роботу навіть при значному навантаженні;

– реляційна модель, дані про користувачів, вантажі, координати, маршрути та замовлення мають логічні зв’язки, реляційна структура чудово підходить для відображення таких взаємозалежностей;

– масштабованість – MySQL легко адаптується до зростання обсягів даних, що дозволяє системі розвиватися без необхідності переходу на складніші СУБД;

– висока сумісність із PHP, серверна частина системи реалізується з використанням мови PHP, яка має вбудовані засоби для роботи з MySQL, що спрощує інтеграцію та знижує час розробки;

– підтримка транзакцій та цілісності даних – це важливо для забезпечення коректного зберігання координат, історії руху вантажу, логів тощо;

– гнучке управління доступом, система дозволяє налаштовувати права доступу для різних ролей (адміністратор, диспетчер, клієнт).

У даному проєкті база даних розробляється з використанням реляційної моделі, що дозволяє чітко структурувати інформацію, встановити зв’язки між об’єктами та забезпечити швидкий доступ до даних. Нижче наведено опис основних сутностей, реалізованих у базі даних [15]:

– Users – таблиця користувачів. Містить поля: логін, email, роль користувача (адміністратор, диспетчер, клієнт), хеш пароля, токени автентифікації.

– Cargos – вантажі. Містить назву вантажу, вагу, тип, статус (в дорозі, доставлено, затримано), а також зв’язок із користувачем, який створив запис.

– Routes – маршрути, що включають пункт відправлення (A), пункт призначення (B), маршрутні точки, довжину шляху та очікуваний час у дорозі.

– Coordinates – координати GPS. Зберігаються широта, довгота, час фіксації, а також ідентифікатор вантажу, до якого прив’язані ці координати.

– Notifications – система сповіщень для користувачів про зміни у статусі вантажу, оновлення місцезнаходження або адміністративні події.

База даних побудована за принципом чітких логічних зв’язків:

– Один користувач може мати декілька вантажів (User 1 → *Cargos N*)

– Один вантаж може мати багато координат у процесі руху (Cargo 1 → *Coordinates N*)

– Один маршрут може бути призначений для багатьох вантажів (Route 1 → *Cargos N*)

Ці зв’язки реалізовані через зовнішні ключі (foreign keys), що забезпечує цілісність та узгодженість інформації в таблицях.

2.4 Середовище розробки та основні технології (Visual Studio Code, PHP, CodeIgniter)

Visual Studio Code (VS Code) — це сучасне середовище для написання й редагування коду, яке підтримує велику кількість мов програмування. Його переваги — легкість, зручність у використанні та широкі можливості налаштування. VS Code дозволяє розробникам швидко створювати проєкти, організовувати структуру файлів, редагувати й форматувати код, а також автоматично виявляти та виправляти помилки завдяки вбудованим інструментам і підсвічуванню синтаксису [16].

Цей редактор підтримує різні мови, серед яких і **PHP**. Через платформу **Visual Studio Marketplace** можна встановлювати додаткові розширення для розширення функціоналу. Для PHP VS Code забезпечує підсвічування синтаксису, автозавершення, відстеження дужок та використання фрагментів коду, що значно пришвидшує написання.

Серед додаткових функцій слід виділити підтримку **налагодження (debugging)**. Наприклад, за допомогою розширення **Xdebug**, яке інтегрується з PHP, розробник отримує можливість [17]:

1. **Стежити за виконанням коду** — переглядати повну історію викликів функцій, передані параметри та шлях до помилки.
2. **Виводити дані у зручному форматі** — завдяки підсвічуванню та структурованому представленню `var_dump`, що також охоплює суперглобальні змінні.
3. **Використовувати брейкпойнти** — запускати код построково з можливістю зупинки виконання для перегляду поточних значень змінних безпосередньо в IDE або через браузер.

Таким чином, **Visual Studio Code** у поєднанні з **Xdebug** надає зручні та потужні інструменти для ефективної розробки і налагодження PHP-додатків.

PHP (Hypertext Preprocessor) — це мова програмування, яку часто використовують для створення сайтів і різних веб-застосунків. Вона зручна тим, що спеціально розроблена для роботи з веб-сторінками, тому чудово підходить, коли потрібно зробити сайт не просто красивим, а «живим» — таким, що взаємодіє з користувачем [17-18].

Однією з причин популярності PHP є те, що її підтримує дуже багато людей — величезна спільнота розробників. Завдяки цьому в інтернеті можна знайти чимало готових прикладів, інструкцій, порад, а також спеціальні «набори інструментів» — фреймворки, які суттєво полегшують роботу програміста.

Ще одна перевага — простота у використанні. У PHP досить простий синтаксис (тобто правила, за якими пишеться код), тому з неї зручно починати навіть тим, хто тільки знайомиться з програмуванням. Вона легко поєднується з HTML, а отже, можна без складнощів створювати веб-сторінки, що реагують на дії користувача.

Крім того, PHP має багато можливостей: вона «вміє» працювати з базами даних, зберігати й читати файли, обробляти інформацію з форм — одним словом, робити усе, що потрібно для створення повноцінного сайту або онлайн-сервісу.

CodeIgniter — це зручний і потужний фреймворк на мові PHP, який часто використовують для створення різних веб-додатків. У нашому випадку він став основою для CRM-системи, яку ми розробляли. У цьому розділі розглянемо, як саме ми використовували CodeIgniter, які його особливості та як він взаємодіє з іншими частинами системи [18].

Цей фреймворк обрали не випадково. По-перше, він простий у використанні. Його інтерфейс інтуїтивно зрозумілий, а сам CodeIgniter не нав’язує жорстких правил — це дозволяє швидко розпочати розробку та не витрачати зайвий час на налаштування. По-друге, CodeIgniter добре підходить для створення масштабованих додатків. Його архітектура легко дозволяє розширювати систему, додавати нові модулі чи функції без суттєвих змін у вже готовій частині коду.

Ще однією важливою перевагою є безпека. CodeIgniter вже містить інструменти, які захищають веб-додаток від поширених загроз — наприклад, від атак через підроблені запити (CSRF) або небезпечні дані, введені користувачем.
У нашій CRM-системі CodeIgniter відповідав за так звану «бекенд»-частину — обробку запитів, збереження й обробку даних, керування логікою роботи. Основні моменти, які варто згадати [18]:

- За допомогою **маршрутизації** ми пов’язували URL-адреси з конкретними частинами коду, що відповідають за обробку запитів.
- **Моделі** забезпечували роботу з базою даних: отримання, додавання, зміна чи видалення інформації. Все це — без потреби писати «важкі» SQL-запити, адже CodeIgniter має зручний інтерфейс для роботи з даними.
- **Шаблони** відповідали за вигляд нашої CRM: вони показували дані користувачу, допомагали оформити інтерфейс і дали змогу відокремити візуальну частину від логіки обробки даних.

Завдяки цим можливостям CodeIgniter став надійною основою для нашого веб-додатку й забезпечив швидкий розвиток системи без складнощів у технічному супроводі.

MySQL — це система керування базами даних, яка працює з реляційною моделлю. Вона створена для обробки великих обсягів даних і є альтернативою платним СКБД. MySQL — це проєкт з відкритим кодом, що з часом значно розширився і став одним із найпопулярніших інструментів у розробці веб-додатків.

Найчастіше її використовують для створення динамічних сайтів. Вона підтримується багатьма мовами програмування, працює швидко, стабільно і зручна в налаштуванні. MySQL — це багатопотоковий сервер, який легко інтегрується з PHP, Java, Python та іншими мовами [19].

2.5 Реалізація програмної логіки

У цьому розділі наведено технічний опис структури й архітектури CRM-системи, яка була реалізована на основі фреймворку CodeIgniter і бази даних MySQL. Описано основні складові системи, їхні функції, а також те, як вони взаємодіють між собою для забезпечення стабільної та ефективної роботи всього застосунку [20].

Система побудована за принципом MVC (Model–View–Controller), де:

- **Моделі** відповідають за роботу з базою даних: отримання, збереження, оновлення та видалення інформації.
- **Контролери** отримують запити від користувача, передають їх у відповідну модель, а потім повертають оброблені дані до представлення.
- **Види (шаблони)** формують зовнішній інтерфейс користувача, відображаючи інформацію у зручному вигляді.

Архітектура системи дозволяє легко додавати нові модулі, наприклад, облік клієнтів, моніторинг транспортних засобів або аналітичні звіти. Дані зберігаються у базі MySQL, а доступ до них здійснюється через моделі з перевіркою введення і захистом від SQL-ін’єкцій.

Усі компоненти системи працюють разом як єдиний механізм: користувач вводить запит, контролер обробляє його, модель звертається до бази даних, результат повертається через шаблон користувачу.

Завдяки такій структурі CRM-система є зрозумілою, стабільною та готовою до подальшого розширення.

Структура бази даних

База даних є однією з найважливіших частин будь-якої інформаційної системи, зокрема веб-додатків і CRM-систем. Саме вона відповідає за зберігання всієї інформації, яку використовує система, — від даних користувачів до звітів і журналів дій [21].

У даному проєкті для роботи з даними використовується MySQL — одна з найпопулярніших реляційних систем управління базами даних. Вона забезпечує надійне, швидке й ефективне збереження інформації та добре підходить для інтеграції з PHP-фреймворками, зокрема з CodeIgniter.

Як зазначалося раніше в розділі, концепція бази даних побудована на взаємозв’язках між таблицями. Загалом структура містить 14 таблиць, кожна з яких відповідає за окремий тип інформації — користувачі, ролі, замовлення, повідомлення, транспортні засоби тощо. Усі ці таблиці логічно зв’язані між собою, що дозволяє системі ефективно обробляти запити та формувати звіти.

На рисунку 2.18 представлено схему взаємозв’язків між таблицями, яка відображає логіку організації даних і забезпечує наочне розуміння роботи бази.

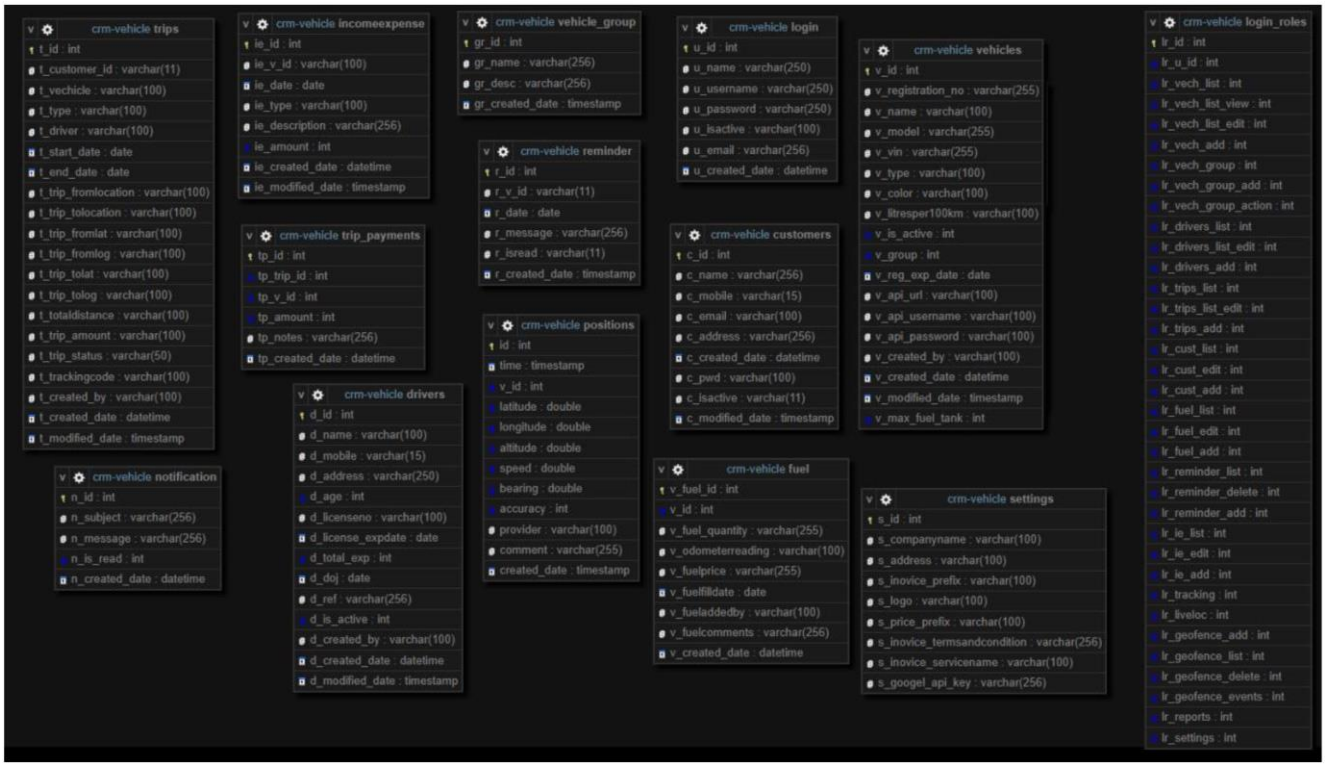

Рисунок 2.18 – Структура таблиць та їх зв’язків у базі даних CRM-системи

Діаграма прецедентів — це зручний інструмент, який допомагає зрозуміти, як саме користувачі взаємодіють із системою. Вона показує, хто і які функції виконує, дає можливість уявити, як виглядають основні сценарії роботи з CRM-системою [21-23].

У цьому розділі представлена така діаграма для нашої системи (рис. 2.19), що ілюструє, як різні ролі — наприклад, клієнт, водій, оператор або адміністратор — працюють із різними частинами функціоналу.

З діаграми видно, що частина можливостей є спільною для кількох ролей. Наприклад, замовник, оператор і адміністратор можуть переглядати інформацію про замовлення. Водночас водій має можливість планувати маршрут і переглядати його разом із замовником — це специфічна дія для цієї ролі.

Загалом така візуалізація допомагає краще структурувати логіку системи й розуміти, як вона може розвиватися надалі, додаючи нові функції чи ролі [22].

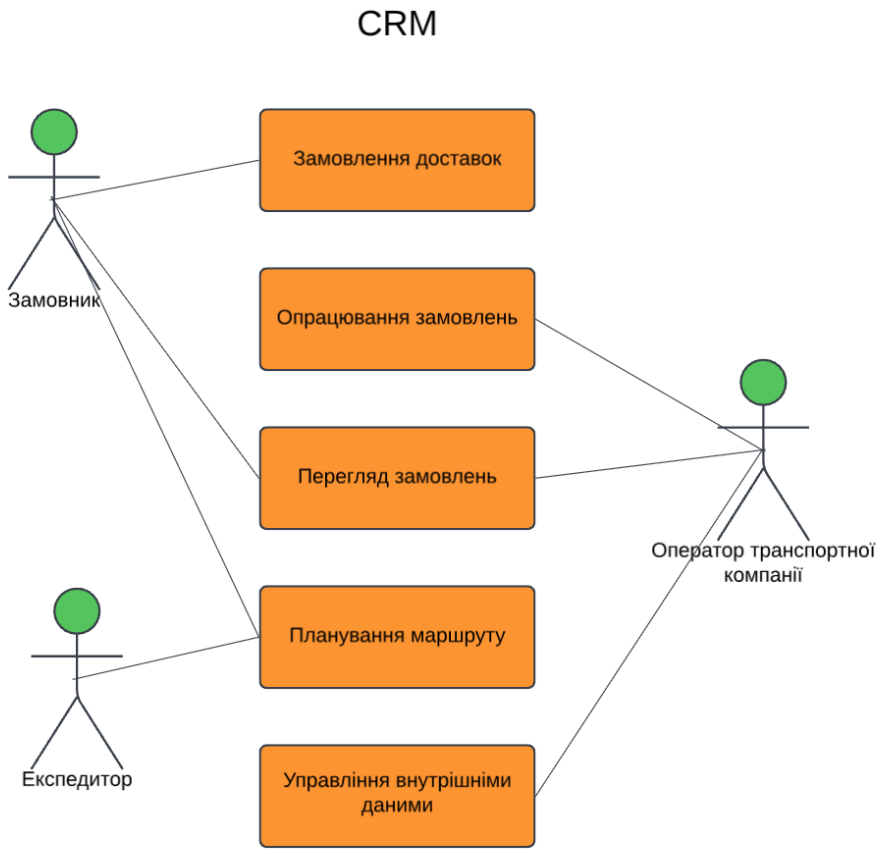

Рисунок 2.19 – Діаграма діяльності

Як показано на рис. 2.20, коли користувач надсилає запит до системи, він спочатку потрапляє до файлу index.php — з цього починається обробка в CodeIgniter.

Наступним етапом працює маршрутизатор (Router), який вирішує, куди передати запит. Якщо сторінка, яку запитує користувач, вже є в кеші, запит не обробляється повторно — система просто повертає збережену відповідь (крок 3).

Якщо сторінки в кеші немає, запит переходить до перевірки на безпеку (крок 4). На цьому етапі система аналізує вхідні дані, щоб переконатися, що вони не містять загроз.

Після цього запит надходить до контролера, який визначає, які моделі, бібліотеки, хелпери або скрипти потрібно підключити. Далі вся оброблена інформація передається у шаблон (View), який формує сторінку для відображення.

Ця сторінка, окрім того, що повертається користувачеві, також зберігається в кеші — щоб наступного разу завантажуватись значно швидше без повторного опрацювання [23].

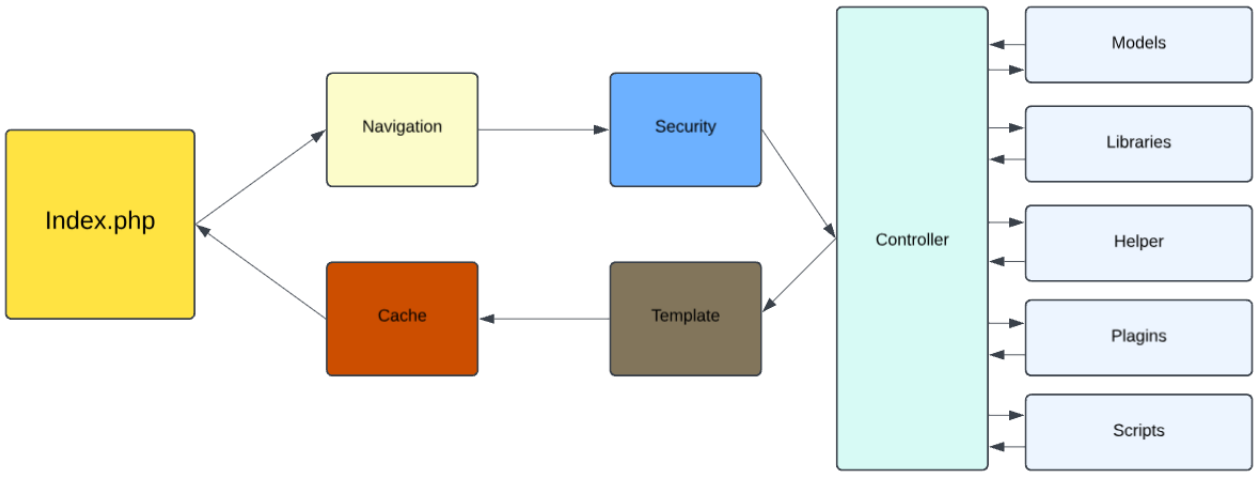

Рисунок 2.20 – Побудова внутрішньої структури програми

Внутрішня структура програмного рішення

У цьому розділі описано основні частини системи, їхнє призначення та взаємозв'язки. Від правильно побудованої структури програмного забезпечення залежить, наскільки система буде зручною в обслуговуванні, гнучкою для оновлень і простою в розширенні.

Застосунок побудований на фреймворку CodeIgniter, який має чітко визначену структуру. На рис. 2.21 зображено основні директорії проєкту, а нижче подано короткий опис кожної з них:

- ***application*** — головна папка, де зберігається весь користувацький код. У ній знаходяться такі підрозділи:
  - *config* — налаштування бази даних, маршрутів, автозавантаження тощо.
  - *controllers* — обробка запитів, взаємодія з моделями та шаблонами.
  - *models* — доступ до бази даних, обробка бізнес-логіки.
  - *views* — шаблони веб-сторінок, що виводяться користувачу.

- *libraries* — додаткові бібліотеки, створені або підключені вручну.
- *helpers* — допоміжні функції для повторного використання в коді.
- *language* — мовні файли для підтримки багатомовності.
- *third_party* — сторонні бібліотеки або пакети.

- ***system*** — ядро фреймворку CodeIgniter. Тут зберігаються технічні файли, які не рекомендується змінювати вручну, оскільки вони відповідають за основну роботу фреймворку.
- ***public*** — папка для публічних ресурсів: стилів CSS, JavaScript, зображень тощо. Усе, що завантажується у браузері напряму, розміщується саме тут.

Завдяки такій структурі система чітко розділена на логічні блоки, що робить її легкою для розробки, тестування та підтримки [23].

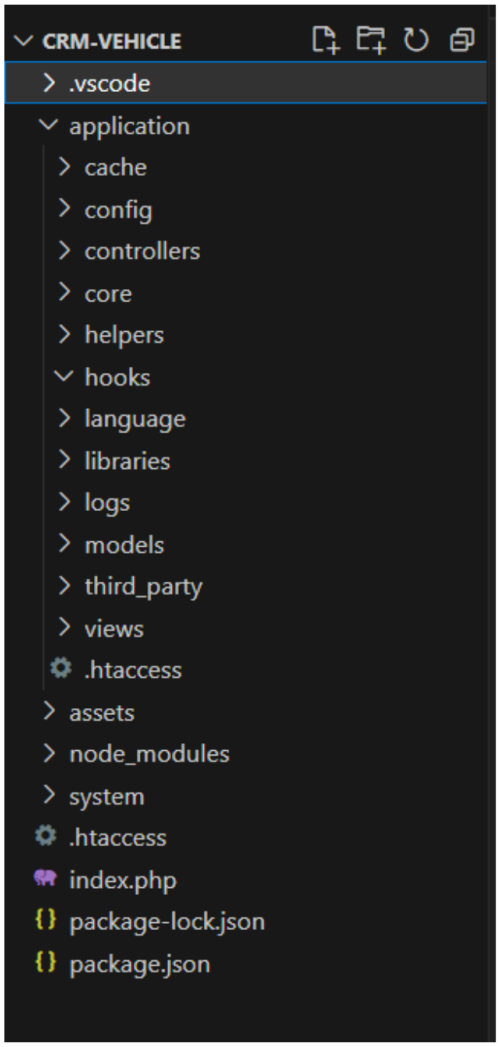

Рисунок 2.21 – Основні каталоги та файли програмного продукту

На рис. 2.22 зображено блок-схему, яка показує, як працює GPS-контролер у системі. Все починається з ініціалізації — контролер запускається, оголошує змінні та готує необхідні ресурси для роботи.

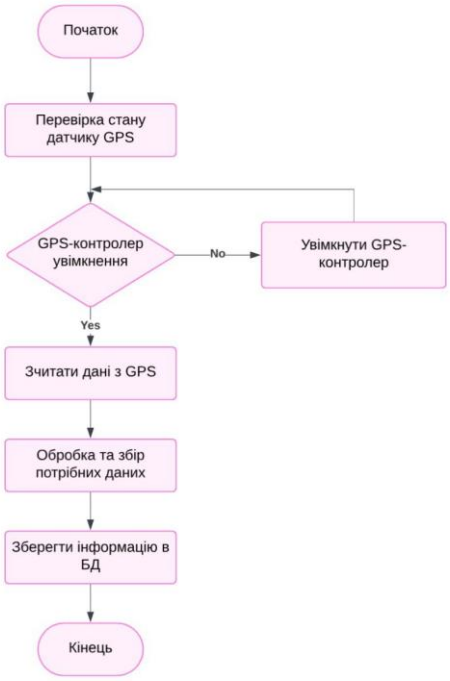

Рисунок 2.22 – Блок-схема роботи GPS-контролер

Далі встановлюється з'єднання з GPS-пристроєм, з якого контролер починає отримувати дані. За допомогою окремої підпрограми зчитуються GPS-пакети, які потім розшифровуються і перетворюються у зрозумілий формат. Інформація, як-от координати, швидкість чи напрямок руху, зберігається в базі даних.

Після отримання координат, система перевіряє, чи дані правильні. Якщо є помилки або неточності, такі записи фільтруються або виправляються, якщо це можливо. Потім виконується обробка — визначаються такі параметри, як широта, довгота, швидкість тощо.

На основі цих GPS-даних система приймає рішення. Наприклад, чи перебуває об'єкт у визначеній зоні, чи відбувається якась подія (наприклад, перевищення швидкості). Якщо так — запускається відповідна дія або фіксація.

Усі дії, включно з обробкою або перевірками, завершуються збереженням у базі даних. Після цього дані можуть бути використані для виведення в інтерфейсі

користувача. Алгоритм завершується на кроці «Кінець», що означає завершення циклу роботи GPS-контролера [24].

2.6 Google Maps API та інтеграція з картографічними сервісами

Інтеграція Google Maps API у CRM-систему.

Google Maps API — це потужний набір інструментів, розроблений компанією Google, який дозволяє додавати функціональність карт і геолокації до веб-додатків. Його можливості широко застосовуються в сучасних CRM-системах для зручного відображення та обробки просторових даних.

Основні можливості API, що використовуються:

- **Інтерактивне відображення карт:** За допомогою Google Maps API користувачі CRM можуть переглядати інтерактивні карти з розташуванням клієнтів, об’єктів компанії або інших важливих точок. Це значно покращує візуалізацію та аналітику даних.
- **Геокодування та зворотне геокодування:** API надає функціонал для перетворення поштових адрес у координати (широта й довгота) та навпаки. Це дає змогу точно визначати місцезнаходження клієнтів, шукати найближчі об'єкти, або автоматично заповнювати адресну інформацію.
- **Побудова маршрутів і навігація:** Завдяки функціям маршрутизації можна розраховувати оптимальні шляхи між точками на карті, враховуючи відстань і час у дорозі. Такий підхід особливо корисний для планування логістики: доставки, виїздів до клієнтів або координації роботи між складами.

Застосування у розробленому додатку:

У рамках створення системи для відстеження руху транспорту, Google Maps API використовується для візуалізації розташування об’єктів. На сторінках, таких як «***Історія***» і «***Поточне місцезнаходження***», карти інтегровані для зображення маршруту або поточної позиції транспортного засобу, що значно підвищує зручність користувача [25].

3 ВПРОВАДЖЕННЯ ТА ДОСТУП КОРИСТУВАЧІВ

3.1 Деплой програми та системні вимоги

Розроблений веб-додаток не потребує спеціального програмного забезпечення — для його використання достатньо будь-якого пристрою з доступом до Інтернету. Завдяки адаптивному дизайну, інтерфейс коректно відображається як на мобільних телефонах, так і на настільних комп'ютерах та ноутбуках.

Щоб надати публічний доступ до застосунку, його необхідно розгорнути (деплоїти) на сервері. Поки додаток знаходиться лише на локальній машині або в репозиторії, він не є доступним для користувачів через мережу. Повноцінна робота системи починається лише після правильного налаштування та запуску на хостинговому сервері.

Розгортання веб-додатку на сервері (деплой).

Deploy (деплой) — це процес публікації веб-сайту або веб-додатку на сервері з метою надання доступу користувачам через Інтернет. Саме після розгортання ресурс стає доступним для взаємодії через браузер.

Основні етапи деплою:

- Завантаження програмного коду на сервер;
- Встановлення необхідних залежностей;
- Збірка (компіляція) фронтенд-частини за потреби;
- Проведення міграцій бази даних (наприклад, запуск SQL-скриптів);
- Оновлення версії застосунку.

Покрокова інструкція для розгортання PHP-проєкту з MySQL:

1. **Підготовка веб-сервера.**

Сервер має мати встановлений та налаштований веб-сервер (Apache або Nginx), який забезпечує обробку PHP-скриптів.

2. **Встановлення PHP.**

Необхідно встановити відповідну версію PHP та переконатися, що активовані всі потрібні розширення (наприклад, mysqli, pdo, mbstring).

3. **Налаштування бази даних.**

Потрібно встановити СУБД MySQL або сумісну (MariaDB), створити нову базу даних і надати доступ до неї користувачу.

4. **Передача файлів на сервер.**

Слід скопіювати структуру проєкту до директорії, з якої веб-сервер обслуговує сайти (наприклад, /var/www/html або іншу відповідно до конфігурації).

5. **Налаштування конфігураційних файлів.**

У файлах налаштувань проєкту необхідно вказати параметри підключення до бази даних: хост, логін, пароль та ім’я бази.

6. **Імпорт структури бази даних.**

Необхідно імпортувати дамп SQL до новоствореної бази за допомогою phpMyAdmin або через командний рядок.

7. Запуск і перевірка сайту.

Після завершення усіх налаштувань потрібно відкрити адресу сайту в браузері та переконатися, що система працює коректно [26].

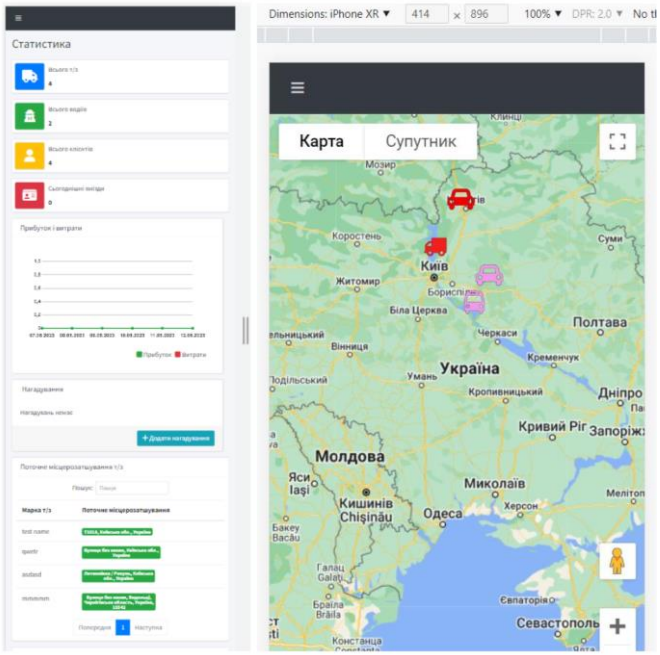

Рисунок 3.1 – Відображення адаптивного інтерфейсу додатку на мобільному пристрої

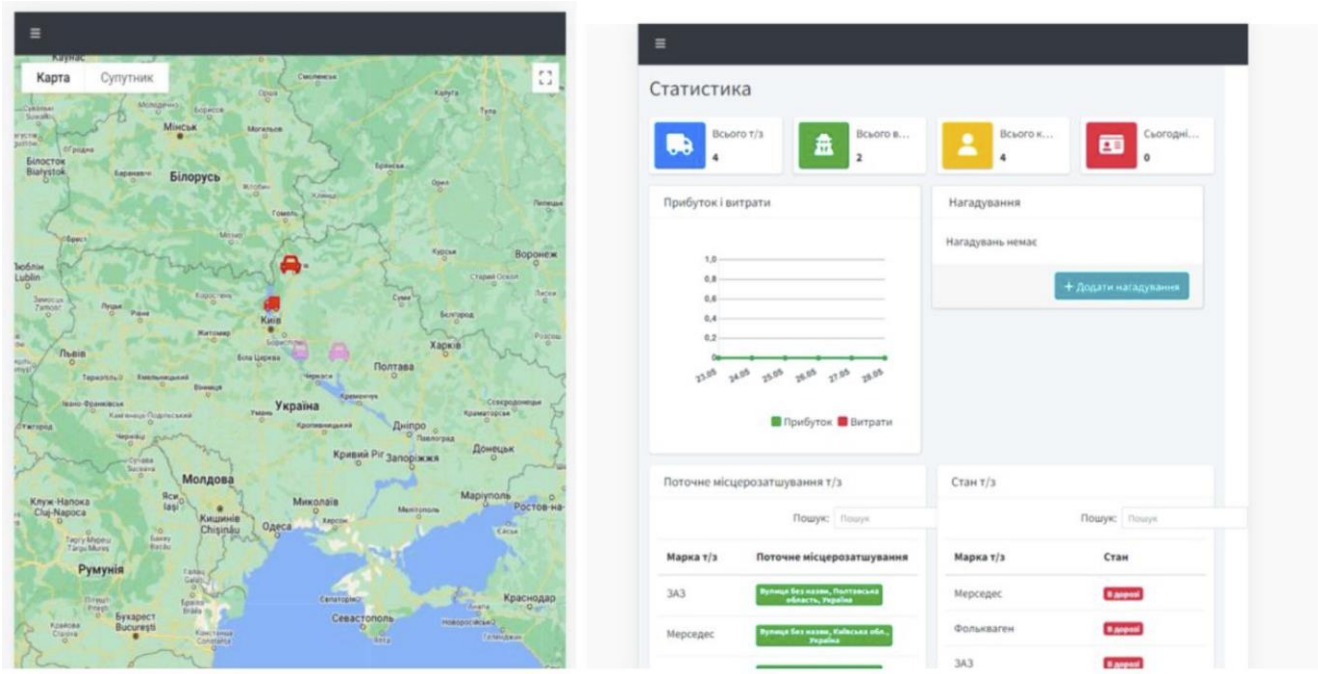

Рисунок 3.2 – Відображення адаптивного дизайну додатку на планшетному пристрої

Проведений аналіз адаптивності користувацького інтерфейсу веб-додатку демонструє його ефективну роботу на різних типах пристроїв. На рисунку 4.1 показано коректне відображення елементів інтерфейсу на мобільному телефоні, збереження читабельності, доступності функціоналу та зручної навігації. Рисунок

3.2 ілюструє, що додаток аналогічно добре масштабується під роздільну здатність планшетів — усі ключові блоки та елементи розташовані логічно й гармонійно. Це підтверджує відповідність додатку сучасним вимогам адаптивного дизайну, що забезпечує стабільний і комфортний досвід користувача незалежно від пристрою доступу [26].

3.2 Алгоритм роботи та основні функції (AJAX, REST API)

Детальний опис функцій для роботи з географічними даними

Опис програмного коду, що реалізує обробку та розрахунки на карті, сприяє кращому розумінню внутрішніх процесів системи, її логіки та принципів роботи. Такий підхід дозволяє не лише полегшити подальше масштабування та модернізацію проєкту, а й забезпечує більш ефективне налагодження під час розробки чи впровадження нових функцій.

Зазначені функції відіграють ключову роль у роботі системи, яка використовує географічні координати, відображення об’єктів на карті та інші можливості, пов’язані з геолокацією. Аналіз їх роботи дозволяє глибше зрозуміти, як саме відбувається взаємодія з картографічними API, які алгоритми залучено до обробки координат, маршрутизації та візуалізації об’єктів на мапі.

Усі приклади коду реалізовані з використанням бібліотеки ***Google Maps***, яка забезпечує інтерактивне відображення просторових даних, а також реалізацію функціоналу для **відстеження транспорту в реальному часі [27]**.

Функція livetracking()

Функція livetracking() відповідає за відображення в реальному часі переміщення транспортних засобів на карті. Вона забезпечує динамічне оновлення даних через асинхронні запити (AJAX), отримуючи з сервера координати об’єктів та іншу супутню інформацію.

Основні етапи виконання функції:

1. ***Перевірка ідентифікатора транспортного засобу.***

Якщо функції передано параметр `id`, формується відповідний маршрут до API для отримання актуального положення заданого транспортного засобу.

2. ***Надсилання AJAX-запиту для отримання маршруту.***

   На основі введеного коду відстеження `t_trackingcode` здійснюється запит до сервера з метою отримання даних про поїздку, включно з координатами.

3. ***Визначення точок маршруту.***

   Із отриманих координат визначаються початкова та кінцева точки маршруту.

4. ***Обробка маршруту за допомогою функції*** *`fromAToB`.*

   Для побудови та відображення маршруту на карті викликається функція `fromAToB` (опис якої наведено в підпункті 6.3.2), яка здійснює розрахунок шляху між заданими точками.

5. ***Отримання даних про поточне місцезнаходження транспорту.***

   Надсилається другий AJAX-запит для отримання актуального положення всіх транспортних засобів.

6. ***Генерація маркерів на карті.***

   На основі координат створюються маркери, які вказують на розташування транспортних засобів.

7. ***Додавання інформаційних віконець до маркерів.***

   Кожен маркер доповнюється спливаючим вікном з даними про модель ТЗ, його швидкість, час останнього оновлення тощо.

8. ***Збереження маркерів у масиві*** *`markersArray`.*

   Усі створені маркери додаються до масиву `markersArray` для подальшого керування (очищення, оновлення, фільтрація тощо).

9. ***Обробка винятків та повідомлень.***

   У разі помилок запитів або відсутності даних передбачено відповідні повідомлення для користувача або альтернативне відображення інформації [27].

Функція fromAToB()

Призначена для обчислення маршруту між двома точками на карті з можливістю додавання проміжних зупинок. Вона приймає три параметри: fromLocation (початкова точка), toLocation (кінцева точка) та необов’язковий параметр waypoints (список проміжних точок).

Основні кроки роботи функції:

1) Виконується очищення карти від усіх попередніх маркерів.

2) Очищується HTML-блок .general .card-body, у якому виводиться загальна інформація про маршрут.

3) Зберігаються координати початкової та кінцевої точок, а також проміжні зупинки маршруту.

4) Створюється об’єкт directionsService з бібліотеки Google Maps для формування маршруту.

5) Якщо вже існує об’єкт directionsDisplay, його скидають у значення null, щоб уникнути конфлікту з новим маршрутом.

6) Ініціалізується новий об’єкт directionsDisplay, який відповідає за відображення маршруту на мапі.

7) Додається обробник події directions_changed, який реагує на зміну маршруту і може використовуватись для додаткових дій, наприклад, збереження або аналізу маршруту.

8) Формується об’єкт request, який включає всі необхідні параметри: координати початку, кінця і проміжні точки.

9) Надсилається запит до сервісу Google Maps з метою отримання маршруту.

10) У разі успішної відповіді (статус OK), маршрут відображається на мапі за допомогою directionsDisplay.

Ця функція забезпечує наочну побудову маршруту та дозволяє користувачу взаємодіяти з інформацією про поїздку. Вона є складовою частиною логіки відображення транспортних засобів у системі моніторингу на основі картографічних сервісів Google [28].

Функція checkFuelAndSetMarkers()

Реалізує логіку визначення доступної відстані на основі поточного рівня палива та витрати, а також візуалізує на карті потенційні точки заправки. Її використання дозволяє користувачеві проаналізувати, наскільки вистачить палива в межах заданого маршруту, і де саме можуть виникнути потреби у заправці.

Основні етапи виконання функції:

1) Зчитуються значення з полів вводу, а саме: поточний обсяг пального (currentFuel), витрата пального на 100 км (litresPer100Km) та максимальний обсяг паливного бака (maxFuelTank).

2) Обчислюється максимальна відстань, яку транспортний засіб може подолати з повним баком (maxDistanceOnFullTank).

3) Виконується розрахунок поточної доступної відстані на основі значення currentFuel та параметра витрати.

4) Очищуються попередньо додані маркери з карти, щоб не дублювати інформацію при повторному запуску функції.

5) Ініціалізуються змінні для накопичення витрати палива (sumFuel) та зберігання нових маркерів (fuelMarkers).

6) Отримується структура маршруту (directionsResult), яка включає набір кроків маршруту з координатами.

7) Для кожного кроку маршруту виконується аналіз витрати палива та, при виявленні потенційної точки, де рівень пального може бути критичним, додається відповідний маркер.

8) На карту додаються всі розраховані маркери із зазначенням місць можливих зупинок для заправки.

9) Після затримки в 1 секунду виводиться інформація на карту щодо прогнозованої витрати палива.

Завдяки цій функції користувач отримує можливість оцінити ресурс паливного запасу на конкретному маршруті, а також візуально спостерігати за ключовими точками, де необхідна або бажана заправка.

Такий підхід підвищує зручність планування поїздок, особливо для комерційного транспорту або тривалих поїздок.

```
function checkFuelAndSetMarkers() {
    const currentFuel = parseFloat(document.getElementById('currentFuel').value);
    const litresPer100Km = parseFloat(document.getElementById('litresPer100Km').value);
    const maxFuelTank = parseFloat(document.getElementById('maxFuelTank').value);

    const maxDistanceOnFullTank = (maxFuelTank / litresPer100Km) * 100;
    let availableDistance = (currentFuel / litresPer100Km) * 100;

    clearPreviousMarkers();

    let sumFuel = 0;
    let fuelMarkers = [];

    const directionsResult = getDirectionsResult(); // припустимо, що функція вже визначена

    directionsResult.routes[0].legs[0].steps.forEach(step => {
        const distance = step.distance.value / 1000;
        sumFuel += (distance * litresPer100Km) / 100;

        if ((availableDistance - distance) <= 20) { // якщо скоро буде мало палива
            const position = step.end_location;
            const marker = new google.maps.Marker({
                position: position,
                map: map,
                icon: 'fuel_icon.png',
                title: 'Можлива точка для заправки'
            });
            fuelMarkers.push(marker);
        }

        availableDistance -= distance;
    });

    setTimeout(() => {
        displayFuelInfo(sumFuel); // виведення розрахованих даних
```

```
    }, 1000);
}
```

Цей код демонструє загальний принцип: він читає дані, обчислює залишкову відстань і додає маркери у точки, де ймовірне падіння рівня пального. За потреби можу адаптувати під конкретну структуру твого додатку [27-29].

Функція predictRest()

Призначена для прогнозування зупинок на відпочинок під час подорожі, враховуючи задані параметри, і візуалізує ці зупинки на інтерактивній карті.

Її реалізація охоплює кілька ключових етапів:

1) Зчитування даних, введених користувачем у відповідні поля, що визначають тривалість активного водіння (driveTime) та перерв (restTime).

2) Видалення попередніх маркерів з карти з метою оновлення актуального стану маршруту.

3) Ініціалізація змінних, які обчислюють накопичену тривалість подорожі та відпочинку (sumDrivingTime і sumRestTime).

4) Отримання даних про маршрут із системи навігації (directionsResult), на основі яких будуть розміщуватись маркери.

5) Поступова обробка кроків маршруту для обчислення моментів, коли слід робити зупинки, згідно із встановленим часом водіння.

6) Створення маркерів на мапі для точок, де заплановано відпочинок, і збереження їх у масиві restStopMarkers.

7) Пошук об'єктів інфраструктури поблизу точок відпочинку — таких як кафе, готелі або ресторани — та додавання відповідних маркерів до окремого масиву restPlacesMarkers.

8) Обробка знайдених місць з можливістю їх візуалізації на карті або виконання додаткових операцій з цими об'єктами.

9) Затримка у виконанні на 1 секунду для забезпечення коректного відображення маркерів та подальше виведення зведеної інформації про сумарний час водіння та відпочинку безпосередньо на карту.

Таким чином, функція predictRest() забезпечує інтуїтивно зрозумілий механізм для планування зупинок під час подорожі, допомагаючи водієві краще розподілити час та знаходити комфортні місця для перепочинку вздовж маршруту [29].

3.3 Доступ та управління користувачами (JWT, OAuth)

У сучасних веб-додатках, зокрема тих, які реалізують інтерактивні функції на основі картографічних сервісів і забезпечують персоналізований досвід (як-от прогнозування точок відпочинку на маршруті), критично важливим є контроль доступу до функціоналу та управління користувачами. У цьому контексті застосовуються механізми авторизації та автентифікації, зокрема ***JWT (JSON Web Token)*** та ***OAuth 2.0***.

JWT — маркерна автентифікація

JWT використовується для забезпечення безпечного обміну даними між клієнтом і сервером. Після входу користувача система генерує токен, що містить закодовану інформацію про користувача та його права доступу. Цей токен зберігається на стороні клієнта (зазвичай у локальному сховищі або куках) і автоматично додається до кожного запиту. У контексті функції predictRest(), яка використовує зовнішні API та потребує персоналізованих налаштувань маршруту, JWT забезпечує:

- Безпечний доступ до маршрутизатора та картографічних даних.
- Індивідуалізацію інформації про час водіння та відпочинку для кожного користувача.
- Збереження історії поїздок або налаштувань без необхідності повторного входу.

OAuth 2.0 — делегування доступу

Для інтеграції із зовнішніми сервісами, такими як Google Maps, Google Places API або сервісами для бронювання готелів чи кафе, доречно використовувати OAuth 2.0. Цей протокол дає змогу надавати додатку обмежений

доступ до ресурсів сторонніх сервісів без передачі особистих облікових даних користувача. Наприклад:

- Якщо користувач бажає використовувати особистий Google-акаунт для збереження маршрутів або рекомендацій місць для відпочинку.
- Якщо система надає можливість авторизації через популярні платформи (Google, Facebook), щоб полегшити вхід.
- Якщо користувач погоджується надати доступ до геолокації або історії місцезнаходжень для покращення точності рекомендацій.

Комбіноване застосування

У дипломному проєкті реалізація системи доступу на основі JWT та OAuth дозволяє:

– Гнучко контролювати рівні доступу (наприклад, лише авторизовані користувачі можуть додавати власні точки відпочинку або зберігати маршрути).

– Забезпечити інтеграцію з картографічними API без втрати безпеки.

– Покращити користувацький досвід завдяки безперервній авторизації.

Таким чином, використання технологій JWT та OAuth у системі, яка прогнозує зупинки для відпочинку під час поїздок, не лише забезпечує високий рівень безпеки, а й відкриває можливості для персоналізованого та зручного використання розширеного функціоналу [30].

ВИСНОВКИ

Здійснено комплексне проєктування інформаційної системи моніторингу вантажів, яка відповідає сучасним вимогам логістичної галузі. Було детально визначено основні ролі користувачів, їхні функції та права доступу, що забезпечує ефективну взаємодію з системою на різних рівнях. Описано архітектуру системи, де чітко простежується зв’язок між GPS-трекерами, мобільними мережами та серверною частиною.

Розглянуто інтеграцію ключових технологій позиціонування — GPS, Galileo, GSM — які забезпечують точне відстеження місцезнаходження вантажів у реальному часі. Було побудовано функціональні UML-діаграми та діаграми діяльності, які ілюструють основні сценарії використання системи різними ролями: користувачем, адміністратором і технічним модулем.

Описано архітектуру бази даних на основі MySQL, де реалізовано реляційну модель з урахуванням нормалізації, забезпечення зв’язків між сутностями (Users, Cargos, Routes, Coordinates, Notifications) та гнучке управління доступом. Також представлено інтерфейсні рішення, що спрощують реєстрацію, авторизацію, перегляд маршрутів, аналітику, розрахунок витрат пального та часу відпочинку водія.

Розроблена структура системи є гнучкою, масштабованою та придатною до впровадження в реальні логістичні процеси, що створює основу для подальшої реалізації функціоналу на етапі практичного впровадження.

Здійснено комплексне проєктування інформаційної системи моніторингу вантажів, яка відповідає сучасним вимогам логістичної галузі. Було детально визначено основні ролі користувачів, їхні функції та права доступу, що забезпечує ефективну взаємодію з системою на різних рівнях. Описано архітектуру системи, де чітко простежується зв’язок між GPS-трекерами, мобільними мережами та серверною частиною.

Розглянуто інтеграцію ключових технологій позиціонування — GPS, Galileo, GSM — які забезпечують точне відстеження місцезнаходження вантажів

у реальному часі. Було побудовано функціональні UML-діаграми та діаграми діяльності, які ілюструють основні сценарії використання системи різними ролями: користувачем, адміністратором і технічним модулем.

Описано архітектуру бази даних на основі MySQL, де реалізовано реляційну модель з урахуванням нормалізації, забезпечення зв’язків між сутностями (Users, Cargos, Routes, Coordinates, Notifications) та гнучке управління доступом. Також представлено інтерфейсні рішення, що спрощують реєстрацію, авторизацію, перегляд маршрутів, аналітику, розрахунок витрат пального та часу відпочинку водія.

Розроблена структура системи є гнучкою, масштабованою та придатною до впровадження в реальні логістичні процеси, що створює основу для подальшої реалізації функціоналу на етапі практичного впровадження.

ПЕРЕЛІК ПОСИЛАНЬ

1. Табунщик Г. В. Проектування, моделювання та аналіз інформаційних систем: Навчальний посібник / Г.В. Табунщик, Р.К. Кудерметов, А. В. Притула. – Запоріжжя : ЗНТУ, 2011. – 292 с.

2. Семакин И. Г. Базы данных: Учебник для вузов / И.Г. Семакин, Л.Л. Шестаков. – М.: БИНОМ. Лаборатория знаний, 2012. – 416 с.

3. Корольов А. В. Системи баз даних: Навчальний посібник / А.В. Корольов. – К.: Каравела, 2009. – 320 с.

4. Крамаренко А. Г. Проектування баз даних: Навчальний посібник / А.Г. Крамаренко. – К.: Центр учбової літератури, 2015. – 248 с.

5. Date C. J. An Introduction to Database Systems / C.J. Date. – 8th ed. – Boston: Pearson Education, 2003. – 1024 p.

6. Elmasri R., Navathe S. B. Fundamentals of Database Systems. – 7th ed. – Boston: Pearson, 2016. – 1272 p.

7. Шейко В. М. Геоінформаційні системи в логістиці: Навчальний посібник / В.М. Шейко. – Харків: УкрДАЗТ, 2013. – 220 с.

8. Ковальов Г. Є. Геоінформаційні системи: Навчальний посібник / Г.Є. Ковальов, Л.О. Олійник. – К.: КНЕУ, 2010. – 280 с.

9. Бондаренко С. І. Основи логістики: Навчальний посібник / С.І. Бондаренко. – Харків: ХНАДУ, 2011. – 312 с.

10. Harrington J. L. Relational Database Design and Implementation / J.L. Harrington. – 4th ed. – Morgan Kaufmann, 2016. – 670 p.

11. Морозова Л. Г. Транспортна логістика: Навчальний посібник / Л.Г. Морозова. – К.: ЦУЛ, 2014. – 352 с.

12. Gill A. Geographic Information Systems and Transportation: Principles and Applications / A. Gill. – Springer, 2019. – 398 p.

13. PostGIS in Action / L. Obe, R. Hsu. – 2nd ed. – Manning Publications, 2015. – 612 p.

14. Шаповал І. М. Системи управління базами даних: Навчальний посібник / І.М. Шаповал. – К.: КНЕУ, 2018. – 348 с.

15. Bernhardsen T. Geographic Information Systems: An Introduction / T. Bernhardsen. – 3rd ed. – Wiley, 2002. – 428 p.

16. Visual Studio Documentation URL: https://docs.microsoft.com/ru-ru/visualstudio/install/install-visual-studio

17. Common Language Runtime (CLR) overview URL: https://docs.microsoft.com/en-us/dotnet/standard/clr

18. Структура веб-сайтів, різновиди веб-сайтів URL: http://urokinformatiki.in.ua/urok-26-struktura-veb-sajtivriznovidi-veb-sajtiv/

19. Жученко А. І., Ярощук Л. Д. Основи проектування баз даних : навчальний посібник для студентів спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» / НТУУ «КПІ». – Київ : НТУУ «КПІ», 2015. – 158 с.

20. S. K. Singh. Database Systems: Concepts, Design and Applications / S. K. Singh – London : Dorling Kinderslay. – 2011

21. Steven Holzner. PHP: The Complete Reference. P. 15.

22. Табунщик Г. В. Проектування, моделювання та аналіз інформаційних систем: Навчальний посібник / Г.В. Табунщик, Р.К. Кудерметов, А. В. Притула. – Запоріжжя : ЗНТУ, 2011. – 292 с.

23. CSS Tutorial URL: https://www.w3schools.com/css/.

24. Daniel Correa, Paola Vallejo. Practical Laravel: Develop clean MVC web applications. Independently published, 2022. 177 pp.

25. Dane, Cameron HTML5, JavaScript, and jQuery 24 "Hour Trainer / Dane Cameron. - 150 с.

26. Thomas H. Cormen. Introduction to algorithms. Second Edition / Thomas H. – Cambridge : The MIT Press. – 2005, pp. 521-525

27. Документація до фреймворку CodeIgniter. URL: https://codeigniter.com/userguide3/general/welcome.html.

28. Jimmy Molina Ríos, Nieves Pedreira-Souto. Approach of Agile Methodologies in the Development of Web-Based Software URL: https://www.researchgate.net/publication/336522596_Approach_of_Agile_Methodologies_in_the_Development_of_Web-Based_Software.

29. OpenAI Platform Documentation URL: https://platform.openai.com/docs/.

30. Zeleniy, O., Rudenko, D., Lyubchenko, V., & Lyashenko, V. (2022). Image Processing as an Analysis Tool in Medical Research. IJAAR.

31. Cherednichenko, O., Vovk, M., Yanholenko, O., & Yakovleva, O. (2020). Towards the Technology of Employers' Requirements Collection Development. Integrated Computer Technologies in Mechanical Engineering: Synergetic Engineering, 1113, 228-239. Springer Nature.

32. VI Міжнародна науково-технічна конференція «Сучасний стан та перспективи розвитку IoT», 15 травня 2025 року, ДУІКТ – «Модель бд для відображення місцерозташування вантажу»

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)

Державний університет інформаційно-комунікаційних технологій

Кафедра інформаційних систем та технологій

Кваліфікаційна робота на тему:

«МОДЕЛЬ БД ДЛЯ ВІДОБРАЖЕННЯ МІСЦЕРОЗТАШУВАННЯ ВАНТАЖУ »

на здобуття освітнього ступеня бакалавра
зі спеціальності 126 Інформаційні системи та технології
освітньо-професійної програми Інформаційні системи та технології

Виконав: НІКІФОРОВ Єгор ІСД-42
Науковий керівник роботи: САГАЙДАК Віктор

КИЇВ - 2025

Актуальність дослідження. У сучасному світі логістика є важливою складовою економічної діяльності, і ефективне управління перевезеннями відіграє ключову роль у зниженні витрат, покращенні сервісу та мінімізації ризиків. Одним із головних аспектів сучасної логістики є контроль місцезнаходження вантажу в реальному часі.

Мета роботи – розробка моделі бази даних для ефективного відображення та управління інформацією про переміщення вантажів.

Об'єкт дослідження – інформаційна система моніторингу вантажів.

Предмет дослідження – методи та моделі управління базами даних для зберігання та обробки даних про місцезнаходження вантажів.

Наукові завдання:

- Дослідити проблеми управління перевезеннями та відстеження вантажу у логістичних компаніях.
- Проаналізувати сучасні інформаційні системи, що використовуються для моніторингу вантажоперевезень.
- Розробити концептуальну, логічну та фізичну моделі бази даних для системи моніторингу.
- Визначити методи зберігання та обробки інформації про переміщення вантажів.
- Інтегрувати базу даних із системами GPS-моніторингу та картографічними сервісами.
- Виконати тестування працездатності системи та оцінити її ефективність у логістичних процесах.
- Запропонувати рекомендації щодо подальшого вдосконалення системи для підвищення її продуктивності та надійності.

2

Логістика та її проблеми

3

Однією з головних проблем є відсутність своєчасного та достовірного інформування про місцезнаходження вантажів, що ускладнює контроль за перемішенням та створює ризики затримок. Затримки у доставці, своєю чергою, часто спричиняються не лише неефективним плануванням маршрутів, а й браком взаємодії між логістичними ланками та недостатнім використанням сучасних IT-рішень. Додаткову загрозу становлять втрати та пошкодження вантажів під час перевезення, що може бути наслідком людського фактору, технічних збоїв або поганої координації.

Проблема	Опис
Відсутність актуальної інформації	Інформація про вантаж оновлюється із запізненням або вручну, що ускладнює контроль.
Затримки у доставці	Виникають через неефективне планування маршрутів або технічні проблеми.
Втрати та пошкодження вантажів	Через відсутність моніторингу вантажі можуть бути втрачені або пошкоджені.
Низький рівень автоматизації	Використання застарілих методів управління знижує ефективність.
Непрозорість логістичних процесів	Слабка інтеграція систем ускладнює аналітику та прийняття рішень.

Таблиця 1 Основні проблеми логістичної галузі

Користувачі та їхні ролі

4

Інформаційна система моніторингу вантажів розробляється з урахуванням потреб різних категорій користувачів, які взаємодіють із нею на різних рівнях доступу. Кожна роль виконує специфічні функції, спрямовані на забезпечення стабільної роботи системи, своєчасного оновлення даних, координації логістичних процесів та зручного інформування клієнтів. Успішна реалізація проєкту можлива лише за умови чіткого розподілу повноважень між учасниками, що забезпечує як ефективність управління, так і безпеку обробки інформації. Найвищий рівень доступу має адміністратор, який здійснює контроль за всіма модулями системи. Диспетчер (логіст) є основним виконавцем операційної частини – формує маршрути, слідкує за місцезнаходженням вантажів, аналізує маршрути та звіти.

Роль користувача	Опис	Основні функції
Адміністратор	Користувач з повним доступом до системи, відповідальний за налаштування та підтримку працездатності.	Керування обліковими записами Повний доступ до всіх функцій Моніторинг системної активності Управління даними вантажів Налаштування прав доступу
Диспетчер (логіст)	Основний оператор, що працює з інформацією про вантажі, маршрути та координує перевезення.	Додавання та редагування вантажів Формування маршрутів Перегляд карт та координат Отримання сповіщень Генерація звітів
Клієнт (замовник)	Користувач, що має обмежений доступ для перегляду інформації про свої замовлення.	Перегляд статусу доставки Перегляд місцезнаходження вантажу Отримання повідомлень Особистий кабінет
GPS-система	Зовнішній технічний модуль, що надсилає координати в реальному часі до бази даних.	Надсилання GPS-координат Оновлення місцезнаходження вантажів Взаємодія через API

Таблиця 2 Користувачі системи та їхні ролі

Функціональна модель системи (UML-діаграми, сценарії використання) 5

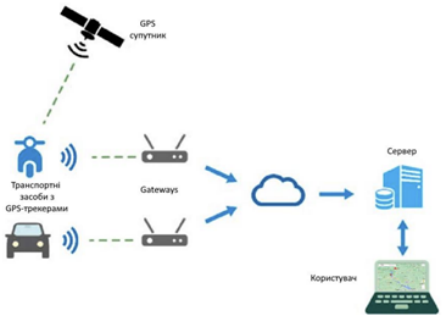

Рисунок 1 – Схема взаємодії транспортного засобу з GPS-трекером, шлюзом та сервером для відображення місцезнаходження вантажу

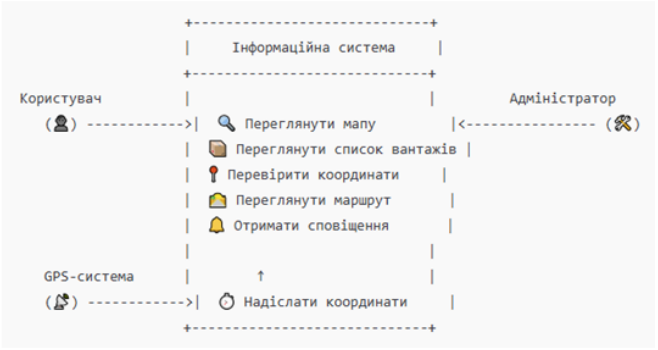

Рисунок 2 – UML-діаграма варіантів використання системи моніторингу вантажів

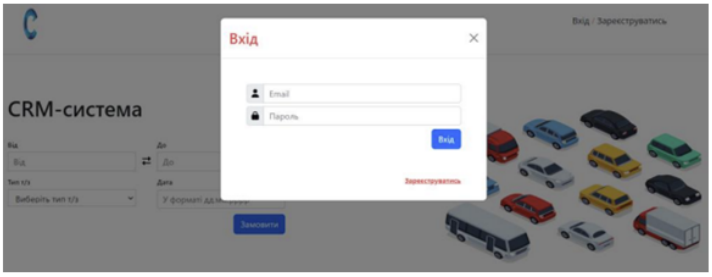

Рисунок 4 – Панель для входу замовника

Рисунок 3 – Форма авторизації користувача

Функціональна модель системи (UML-діаграми, сценарії використання) 6

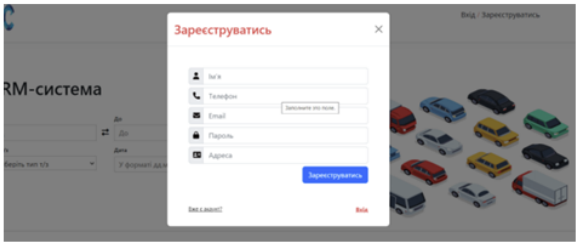

Рисунок 5 – Форма реєстрації замовника в CRM-системі

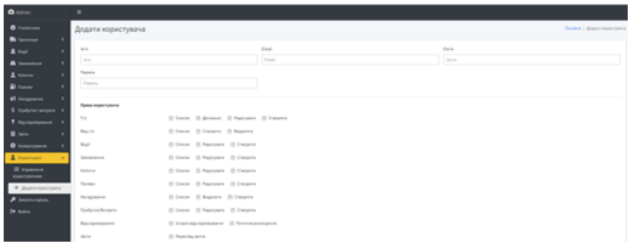

Рисунок 6 – Форма для створення користувача в адміністративній панелі

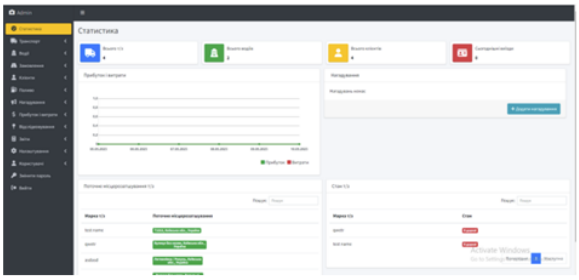

Рисунок 7 – Інтерфейс статистичної панелі адміністратора

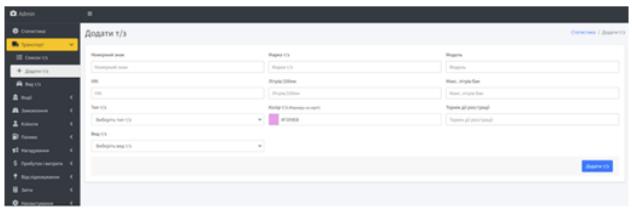

Рисунок 8 – Форма додавання транспортного засобу в систему

Функціональна модель системи (UML-діаграми, сценарії використання) 7

Рисунок 10 – Діаграма діяльності користувача без позначення напрямку дій

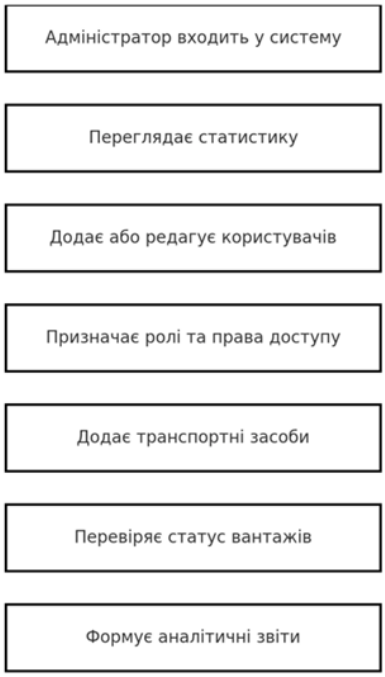

Рисунок 9 – Діаграма діяльності адміністратора в системі

Архітектура та структура бази даних (MySQL, зв'язки між сутностями) 8

Рисунок 11 – Відображення маршруту вантажу на карті в інтерфейсі системи

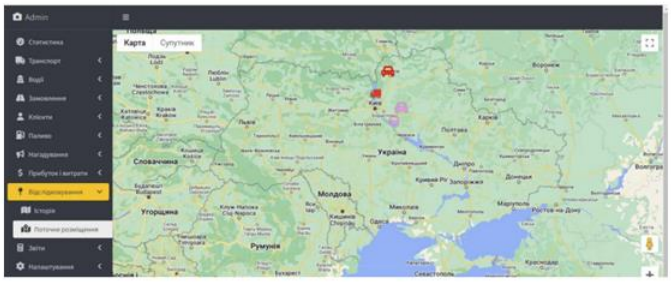

Рисунок 12 – Поточне розміщення транспортних засобів на карті

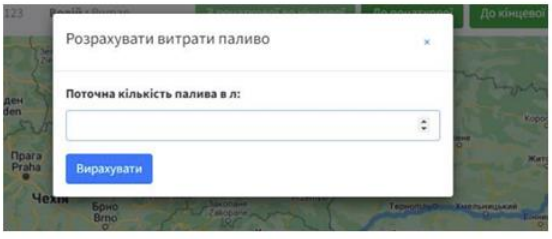

Рисунок 14 – Інтерфейс розрахунку витрат пального в системі

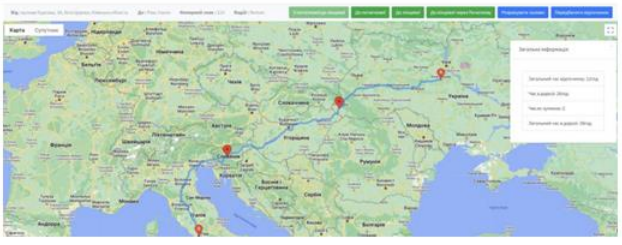

Рисунок 13 – Побудова маршруту міжнародного перевезення з аналітикою руху

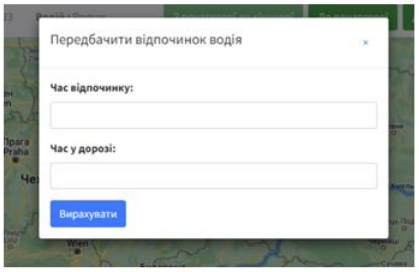

Рисунок 15 – Інтерфейс розрахунку часу відпочинку водія в системі

Реалізація програмної логіки

9

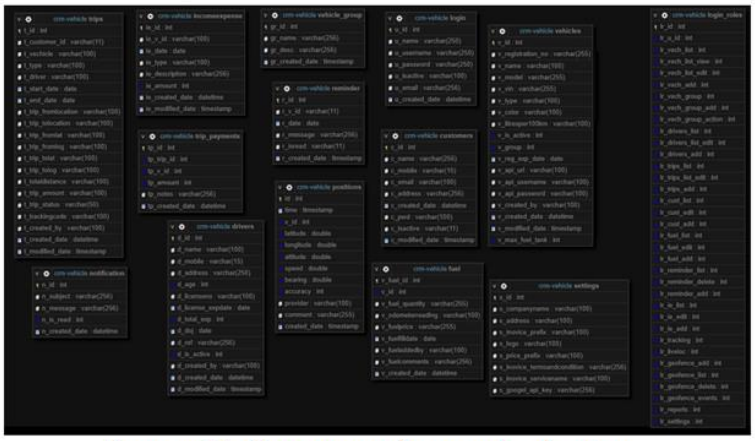

Рисунок 16 – Структура таблиць та їх зв'язків у базі даних CRM-системи

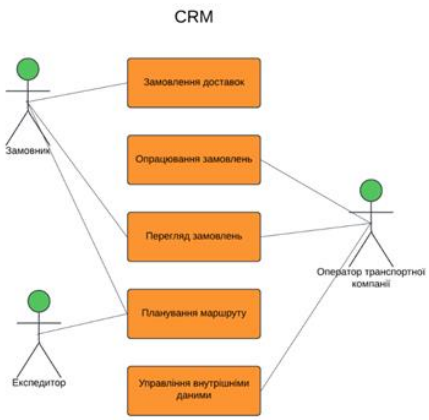

Рисунок 17 – Діаграма діяльності

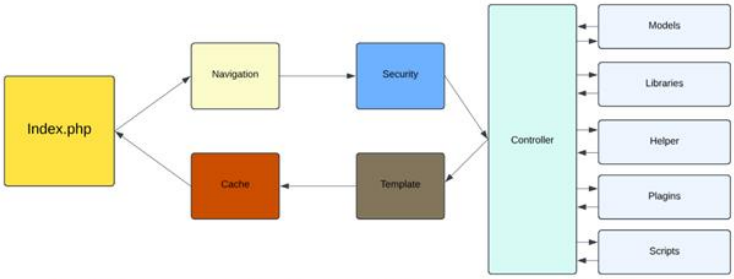

Рисунок 18 – Побудова внутрішньої структури програми

Деплой програми та системні вимоги (10)

Розгортання веб-додатку на сервері (деплой).

Deploy (деплой) — це процес публікації веб-сайту або веб-додатку на сервері з метою надання доступу користувачам через Інтернет. Саме після розгортання ресурс стає доступним для взаємодії через браузер.

Основні етапи деплою:

- Завантаження програмного коду на сервер;
- Встановлення необхідних залежностей;
- Збірка (компіляція) фронтенд-частини за потреби;
- Проведення міграцій бази даних (наприклад, запуск SQL-скриптів);
- Оновлення версії застосунку.

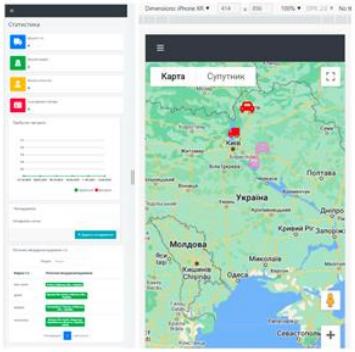

Рисунок 19 – Відображення адаптивного інтерфейсу додатку на мобільному пристрої

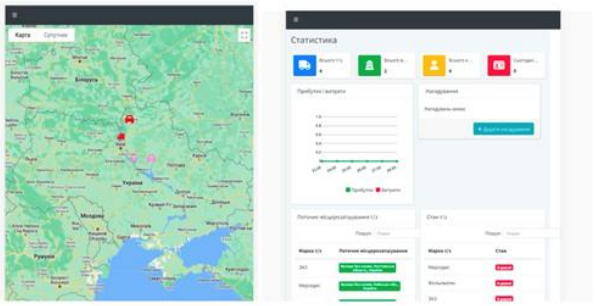

Рисунок 20 – Відображення адаптивного дизайну додатку на планшетному пристрої

ВИСНОВКИ (11)

- Здійснено комплексне проєктування інформаційної системи моніторингу вантажів, яка відповідає сучасним вимогам логістичної галузі. Визначено ролі користувачів, їхні функції та права доступу для ефективної взаємодії з системою.
- Описано архітектуру системи з чітким зв'язком між GPS-трекерами, мобільними мережами та серверною частиною. Забезпечено точне відстеження вантажів у реальному часі через інтеграцію GPS, Galileo та GSM.
- Побудовано UML-діаграми та діаграми діяльності, які демонструють ключові сценарії взаємодії користувачів із системою. Розглянуто сценарії для користувачів, адміністраторів і технічних модулів.
- Архітектура бази даних на MySQL реалізована з урахуванням нормалізації та зв'язків між сутностями. Забезпечено гнучке управління доступом до даних.
- Представлено інтерфейсні рішення для реєстрації, авторизації, перегляду маршрутів та аналітики. Реалізовано функції розрахунку витрат пального і часу відпочинку водія.
- Структура системи є гнучкою та масштабованою, що дозволяє впровадити її в реальні логістичні процеси. Це створює основу для подальшого розширення функціоналу на етапі практичного застосування.

Апробація

12

VI Міжнародну науково-технічну конференцію «Сучасний стан та перспективи розвитку IoT»

Підкреслено важливість об’єднання різних підсистем у єдину інтегровану інформаційну систему для моніторингу вантажів. Розглянуто типові протоколи зв’язку, що забезпечують обмін даними між GPS-модулями, базою даних і картографічними сервісами.