Міністерство освіти і науки України
Луцький національний технічний університет
Факультет комп'ютерних та інформаційних технологій
Кафедра інженерії програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА

ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

РОЗРОБКА ВЕБДОДАТКУ ДЛЯ ЛОГІСТИЧНОЇ КОМПАНІЇ З ІНТЕРАКТИВНОЮ КАРТОЮ ТА ЗВОРОТНІМ ЗВ'ЯЗКОМ З ВИКОРИСТАННЯМ VUE.JS, NODE.JS ТА MYSQL

DEVELOPMENT OF A WEB APPLICATION FOR A LOGISTICS COMPANY WITH AN INTERACTIVE MAP AND FEEDBACK SYSTEM USING VUE.JS, NODE.JS AND MYSQL

спеціальність 121 «Інженерія програмного забезпечення»
освітня програма «Інженерія програмного забезпечення»

Виконав: здобувач вищої освіти
групи ІПЗ-41
Озірський Роман Миколайович

Керівник:
Ліщина Наталія Миколаївна

Кваліфікаційну роботу
допущено до захисту
«10» 06 2025 р.
Гарант освітньої програми:
к.т.н., доцент
Ліщина Наталія Миколаївна
_____

Луцьк – 2025 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *інженерії програмного забезпечення*
Ступінь вищої освіти *бакалавр*
Галузь знань: *12 «Інформаційні технології»*
Спеціальність: *121 «Інженерія програмного забезпечення»*
Освітня програма: «*Інженерія програмного забезпечення»*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____
«20» 12 2024р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Озірському Роману Миколайовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи *«Розробка вебдодатку для логістичної компанії з інтерактивною картою та зворотнім зв'язком з використанням Vue.js, node.js та MySQL»*
Керівник роботи: *к.т.н., доцент Ліщина Н. М.*
затверджені наказом закладу вищої освіти від *«19» грудня 2024 р.* № *474/01-02*
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи бакалавра *«10» червня 2025 р.*
3. Вихідні дані до роботи: *Visual Studio Code, Vue.js, SCSS, HTML5, JavaScript (ES6+), Node.js, Express.js, MySQL, Git, Netlify. методичні вказівки до виконання кваліфікаційної роботи бакалавра*
4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити): *проаналізувати предметну область, сформувати вимоги до системи, обрати стек технологій, розробити логістичний вебдодаток, розробити базу даних, протестувати та оптимізувати його для пошукової системи, розробити документацію.*
5. Перелік графічного матеріалу: *10 рисунків, 1 додаток*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	*Ліщина Н. М.*		
Специфікація вимог до розробленої системи	*Ліщина Н. М.*		
Розробка об'єкта проектування	*Ліщина Н. М.*		
Нормоконтроль	*Вознюк А. В.*		
Гарант ОП	*Ліщина Н. М.*		
Показник запозичень тексту	1,39 %		
Академічна доброчесність	*Ліщина Н. М.*		

7. Дата видачі завдання *«20» грудня 2024 р.*

№	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1	Огляд літературних джерел по темі кваліфікаційної роботи бакалавра	до 04.02.2025 р.	Виконано
2	Аналіз проблеми розробки та впровадження об'єкту проектування	до 01.03.2025 р.	Виконано
3	Обгрунтування вибору шляхів, технологій і засобів вирішення поставленого завдання	до 15.03.2025 р.	Виконано
4	Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних	до 29.03.2025 р.	Виконано
5	Практична реалізація об'єкта проектування та розробка бази даних	до 26.04.2025 р.	Виконано
6	Тестування та налагодження об'єкта проектування	до 03.05.2025 р.	Виконано
7	Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру	до 10.06.2025 р.	Виконано

Здобувач вищої освіти _____ Роман ОЗІРСЬКИЙ

Керівник кваліфікаційної роботи _____ Наталія ЛІЩИНА

АНОТАЦІЯ

Озірський Р. М. Розробка вебдодатку для логістичної компанії з інтерактивною картою та зворотним зв’язком з використанням Vue.js, node.js та MySQL. Рукопис. Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення» спеціальності «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2025.

Кваліфікаційна робота бакалавра складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатку.

У першому розділі здійснено аналіз предметної області логістики та сучасних ІТ-рішень та сформульовані завдання на кваліфікаційну роботу.

У другому розділі визначено функціональні й нефункціональні вимоги та чітко наведено детальну специфікацію вимог до розроблюваної системи та описано процес її проєктування з використанням UML-нотації, який включає діаграми прецедентів, класів, активності та послідовності. Також було проаналізовано наявні інструменти на ринку та обрано стек технологій для розробки.

У третьому розділі описано практичну реалізацію вебдодатку, розробку клієнтської та серверної частин із побудовою бази даних, організацією обробки заявок і інтеграцією з інтерактивною картою та Telegram-ботом для оперативних сповіщень. Протестовано та налагоджено проєкт, оптимізовано для пошукових систем та створено технічну документацію.

У висновках зроблено підсумки виконаної роботи, оцінено досягнуті кількісні та якісні показники, визначено практичну значущість рішення та окреслено перспективи подальшого розвитку системи.

Ключові слова: вебдодаток, логістична компанія, автоматизація обробки заявок, інтерактивна карта, зворотний зв’язок, Telegram-сповіщення.

ABSTRACT

Ozirskiy R. Development of a Web Application for a Logistics Company with an Interactive Map and Feedback System Using Vue.js, Node.js, and MySQL. Manuscript. Bachelor's Qualification Work for the Educational Program "Software Engineering", Specialty "Software Engineering". Lutsk National Technical University. Lutsk, 2025.

The bachelor's qualification work consists of an introduction, three chapters, conclusions, a list of references, and an appendix.

In the first chapter, the subject domain of logistics and modern IT solutions is analyzed, and the objectives of the qualification work are formulated.

The second chapter defines the functional and non-functional requirements, provides a detailed specification for the system under development, and describes the design process using UML notation, including use-case, class, activity, and sequence diagrams. It also reviews existing market tools and justifies the chosen technology stack.

The third chapter presents the practical implementation of the web application: development of the client-side and server-side components with database construction, organization of request processing, and integration with an interactive map and a Telegram bot for real-time notifications. The project is tested and debugged, optimized for search engines, and accompanied by technical documentation.

In the conclusions, the work performed is summarized, the achieved quantitative and qualitative results are assessed, the practical significance of the solution is determined, and prospects for further system development are outlined.

Keywords: web application, logistics company, automated request processing, interactive map, feedback, Telegram notifications.

ЗМІСТ

ВСТУП

Актуальність теми. Сьогодні логістичні компанії працюють у дуже динамічних умовах, де важливо реагувати швидко й точно. Від того, наскільки оперативно обробляється інформація, часто залежить успіх компанії та її конкурентоспроможність. У багатьох випадках частина процесів досі виконується вручну, а дані зберігаються в різних програмах або навіть на папері. Це створює плутанину, затримки, дублювання роботи і призводить до втрат важливої інформації під час передачі між працівниками. Саме тому розробка зручного вебзастосунка, який об’єднує всі етапи роботи з клієнтом і всередині компанії, є не просто актуальною – вона реально вирішує щоденні проблеми логістичного бізнесу. Такий інструмент дозволяє диспетчерам, водіям і менеджерам швидко обмінюватися даними, бачити заявки в одному місці та реагувати на них без зайвої тяганини. Клієнтам це теж зручно – їм не потрібно чекати відповідей годинами. А інтерактивна карта в системі дає можливість планувати маршрути в реальному часі, бачити затримки чи зміни погодних умов, і одразу ж підлаштовуватися під ситуацію. Усе це суттєво спрощує щоденну роботу компанії та зменшує ризик помилок.

Окрім зручності для клієнтів і співробітників, вебдодаток також допомагає створити повноцінну базу замовлень, яка стане основою для аналітики та прогнозування. Менеджери зможуть у будь-який момент переглянути, скільки заявок опрацьовано, скільки маршрутів виконано за тиждень чи місяць, скільки часу займає обробка кожної заявки, а також побачити випадки відмов чи проблемні ситуації. Така інформація дозволяє приймати зважені управлінські рішення, а спираючись на реальні цифри. Це відкриває можливість не просто реагувати на ситуації, а постійно вдосконалювати внутрішні процеси й робити їх ефективнішими. У межах цієї роботи увага приділяється не лише технічній стороні – як зберігаються дані, як працює зворотний зв’язок, – а й тому, як за допомогою такого інструменту можна покращити всю логіку роботи логістичної компанії. Також буде

обґрунтовано, чому важливо, щоб система працювала швидко, мала простий і зрозумілий інтерфейс, надійно зберігала інформацію та могла легко масштабуватись у разі збільшення кількості клієнтів або обсягів перевезень.

Метою даної кваліфікаційної роботи бакалавра є розробити вебдодаток для логістичної компанії з вбудованої інтерактивною картою для побудови маршруту.

Об’єктом роботи є бізнес-процеси приймання, обробки та виконання замовлень у сучасних логістичних компаніях, що функціонують у високодинамічному середовищі.

Предметом даної роботи є реалізація механізмів життєвого циклу логістичних даних у вебдодатку: їхнього збору, валідації, збереження та динамічної візуалізації маршрутів і параметрів замовлень.

На період роботи над проєктом були поставлені наступні завдання:

– проаналізувати предметну область;

– сформувати вимоги для вебдодатку;

– обрати стек технологій;

– розробити вебдодаток;

– провести тестування;

– спроєктувати та розробити базу даних;

– оптимізувати вебдодаток для пошукових систем;

– підготувати технічну та користувацьку документацію.

З метою апробації матеріалів роботи підготовлено тези, які були подані до збірнику тез «Міжнародної науково-практичної конференції ІТОНВ-2025» (м. Луцьк, 23-24 травня 2025 року) (додаток А).

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

У сучасних умовах цифровізації бізнесу логістичні компанії активно впроваджують інформаційні технології для оптимізації процесів перевезення, відстеження вантажів та комунікації з клієнтами. Особливої актуальності набувають вебзастосунки, які поєднують функціональність інтерактивної карти для візуалізації маршрутів з інтегрованими інструментами зворотного зв’язку. Такі системи дозволяють не лише підвищити ефективність управління перевезеннями, а й забезпечити прозорість для замовників.

Більшість традиційних логістичних систем зосереджуються переважно на автоматизації внутрішнього обліку (наприклад, управління складами, облік транспорту, маршрутизація), однак не враховують потреби клієнтів у зворотному зв’язку, а також не мають інструментів для інтерактивної візуалізації. Це призводить до зниження задоволеності користувачів, втрати довіри до сервісу у випадках виникнення проблем із доставкою, а також до додаткового навантаження на операторів служби підтримки. Крім того, сучасні умови передбачають потребу в системах, які є адаптивними до різних типів користувачів і здатні працювати як на десктопах, так і на мобільних пристроях.

Порівняльний аналіз існуючих рішень демонструє [1], що такі популярні сервіси, як Route4Me, Onfleet або Tookan, мають потужний функціонал маршрутизації й управління доставкою. Вони дозволяють оптимізовувати логістичні процеси, відстежувати транспорт у реальному часі, автоматично сповіщати клієнтів про статус замовлення. Проте їх основний фокус – це підприємства великого масштабу або кур’єрські служби, де система використовується в комплексі з іншими корпоративними інструментами. У випадку малих і середніх компаній, які працюють на локальному або

регіональному рівні, такі рішення часто є надто дорогими або технічно складними для впровадження.

Ще однією проблемою є відсутність персоналізованого підходу у взаємодії з клієнтом. У багатьох існуючих системах відсутні можливості інтерактивного зворотного зв’язку або прямого впливу користувача на процес доставки. Наприклад, неможливість миттєвого внесення змін до адреси доставки, повідомлення про затримки або просте надсилання відгуку часто викликає розчарування. Це свідчить про необхідність створення інтерфейсу, який забезпечує не лише відображення інформації, але й зручну взаємодію із системою в реальному часі [2].

Окремо слід відзначити, що більшість сучасних логістичних додатків не надають відкритого API або можливостей для редагування, що унеможливлює інтеграцію з іншими внутрішніми системами компаній. У результаті компанії вимушені будувати власні модулі або обходитися без інтеграції, що негативно впливає на ефективність бізнесу та ускладнює масштабування сервісу в майбутньому.

Актуальність створення нової системи полягає в необхідності поєднання кількох ключових характеристик: інтерактивна карта, що дозволяє відображати маршрути в реальному часі; механізм зворотного зв’язку з користувачем, який дозволяє залишати коментарі, скарги або оцінки доставки; адаптивний інтерфейс, що однаково зручно функціонує як на мобільних, так і на стаціонарних пристроях; і можливість розширення або інтеграції з іншими модулями системи компанії. Все це формує підґрунтя для створення інноваційного вебдодатка, орієнтованого на потреби як компаній, так і кінцевих користувачів.

Таким чином, виникає потреба у створенні вебзастосунка, який поєднує в собі зручний адаптивний інтерфейс, інтерактивну карту, повноцінну форму зворотного зв’язку з клієнтом із вбудованою валідацією даних, а також систему зберігання інформації в базі даних.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

Основною метою кваліфікаційної роботи є розробка сучасного вебзастосунка для логістичної компанії, який дозволяє здійснювати інтерактивну візуалізацію маршрутів перевезень, забезпечує зручну форму зворотного зв’язку з клієнтами, а також дає змогу ефективно обробляти та зберігати заявки клієнтів. Для досягнення цієї мети в процесі виконання роботи необхідно реалізувати декілька важливих завдань:

– проаналізувати предметну область і виявити основні потреби логістичних компаній у сфері автоматизації процесів;

– сформувати вимоги для створення вебдодатку;

– обрати засоби та методи розробки для ключових етапів реалізації вебдодатка;

– реалізувати вебдодаток з інтерактивною картою, формою зворотного зв’язку та адмін панелю;

– протестувати вебзастосунок та проаналізувати його функціональність, юзабіліті та відповідність поставленим вимогам;

– спроєктувати базу даних для зберігання інформації про запити користувачів і логістичні дані;

– оптимізувати вебдодаток для пошукових систем;

– підготувати технічну та користувацьку документацію з інструкціями з розгортання й експлуатації системи.

Досягнення поставлених завдань дозволить створити повноцінну систему, здатну покращити якість обслуговування клієнтів, автоматизувати обробку заявок і візуалізувати маршрути перевезень у реальному часі, що в комплексі забезпечить ефективну підтримку бізнес-процесів логістичної компанії. У процесі розробки також передбачено використання принципів модульності та розширюваності, що дозволить у майбутньому легко адаптувати систему під нові вимоги або розширити її функціонал. Зокрема, важливим аспектом є реалізація можливості масштабування застосунку на інші країни чи

регіони, інтеграції з популярними логістичними платформами, такими як Lardi-Trans або Della, а також підключення додаткових інструментів сповіщення, або наприклад Telegram-ботів для обробки заявок у режимі реального часу.

Особлива увага приділяється безпеці збереження та передачі даних користувачів, для чого планується реалізувати систему аутентифікації, захисту API-запитів та шифрування персональної інформації. Крім того, важливим завданням є створення гнучкої адміністративної панелі для керування заявками клієнтів та перегляд статистики.

Усі ці завдання в сукупності дозволять створити конкурентоспроможний інструмент, орієнтований на потреби малого та середнього логістичного бізнесу, який буде простим в керуванні, надійним у роботі та зручним для клієнтів та працівників компанії з різним рівнем технічної підготовки. Проєкт має забезпечити зниження витрат на обробку запитів, мінімізувати людський фактор та підвищити швидкість реагування компанії на запити клієнтів. Підсумкова система повинна мати інтуїтивно зрозумілий інтерфейс, швидкий час відгуку, адаптивність до різних типів пристроїв і можливість подальшого розширення без необхідності кардинальних змін у структурі коду чи базі даних. Таким чином, реалізація вказаних завдань дозволить не лише вирішити поточні функціональні потреби компанії, але й створити гнучку платформу, здатну підтримувати її розвиток у майбутньому. Подальший розвиток даного проєкту може передбачати інтеграцію з різними модулями для моніторингу ефективності логістичних процесів, систем автоматичної формування звітності, а також підтримку багатомовного інтерфейсу для виходу на міжнародний ринок. Також система має залишатися відкритою до оновлень, що сприятиме її тривалій актуальності та стабільній роботі в реальних умовах експлуатації.

Висновки до розділу 1

Проведений аналіз підтверджує потребу у створенні нової системи для логістичних компаній, яка поєднує візуалізацію, гнучкий зворотний зв’язок і

зручність користування. Сучасні виклики логістики потребують рішень, що враховують не лише технічні, але й комунікаційні аспекти взаємодії з клієнтом, адаптивність до різних пристроїв і можливість розширення в майбутньому.

Також сформульовано завдання кваліфікаційної роботи, що охоплюють весь життєвий цикл майбутньої системи: від проєктування інтерфейсу та моделювання інформаційних потоків до інтеграції каналів зворотного зв’язку.

РОЗДІЛ 2
СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проєктування програмного забезпечення

На основі аналізу предметної області та сформульованих завдань кваліфікаційної роботи було визначено функціональні та нефункціональні вимоги до програмного забезпечення.

Для розробки системи обрано поетапну модель розробки, яка дозволяє поступово впроваджувати нові модулі, тестувати їх і адаптувати їх під вимоги користувача. Такий підхід забезпечує гнучкість та знижує ризики при впровадженні.

Вимоги до системи були зібрані шляхом вивчення практичних потреб логістичних компаній малого та середнього бізнесу, а також аналізу користувацьких сценаріїв, що описують типову взаємодію з вебзастосунком.

Методи збору вимог включали порівняльний аналіз аналогічних платформ, інтерв’ювання потенційних користувачів, а також формування узагальнених сценаріїв використання системи у вигляді прецедентів.

Вимоги було структуровано у вигляді специфікації функціоналу, яка містить такі ключові можливості системи:

– введення нових логістичних даних у базу даних;

– редагування, архівація та видалення існуючої інформації;

– пошук даних за запитом користувача;

– обробка та збереження заявок;

– інтеграція з інтерактивною картою для візуалізації маршрутів;

– зворотний зв’язок із клієнтом через динамічну форму;

– захист та перевірка достовірності введених даних.

Система повинна підтримувати доступ через вебінтерфейс, а взаємодія між клієнтом і сервером здійснюватиметься за протоколом TCP/IP. З точки зору безпеки, передбачено реалізацію аутентифікації користувачів, контроль прав

доступу до адміністративних і користувацьких функцій, захист від SQL-ін’єкцій, а також механізми шифрування даних, які зберігаються та передаються. Також одним із основних чинників є контроль цілісності даних у базі, а також перевірка структури запитів та валідація введеної інформації.

UX/UI-дизайн системи був спроєктований з орієнтацією на простоту, інтуїтивну зрозумілість та адаптивність до різних розширень екранів. Інтерфейс користувача розділено на функціональні блоки: форма зворотного зв’язку, інтерактивна карта, панель адміністрування, система сповіщень та інші елементи взаємодії. Для проєктування інтерфейсу будуть використовуватися принципи компонування за шаблонами UI-дизайну та модульного підходу.

Інформаційні потоки в системі описують послідовність взаємодії користувача з функціональними модулями вебзастосунку, зокрема надсилання запиту з форми, обробка даних на сервері, збереження результатів у базу даних, відображення результатів на інтерфейсі. Обробка даних повинна бути асинхронною, з підтримкою валідації на клієнті та сервері. Це забезпечує зручність роботи, підвищену продуктивність і захист від помилок.

Нефункціональні вимоги покривають продуктивність системи, забезпечення безперервної та безпечної обробки даних, інтуїтивно зрозумілий і адаптивний інтерфейс на різних пристроях, а також загальну масштабованість, що дозволить нарощувати обсяги обробки без суттєвих змін архітектури.

Проєктування системи буде проводитись за принципом логічного поділу на шари: презентаційний, бізнес-логіки та зберігання даних. Для візуалізації внутрішніх процесів використано UML-нотацію – побудовано діаграму використання, що дозволяє чітко відобразити взаємодію між компонентами та окреслити їхні функції. Інтерфейс спроєктовано модульно, що гарантує легкість у підтримці та подальшому розширенні, адже кожен функціональний блок виділено як автономний компонент із чітко визначеними точками інтеграції. Таким чином, проведений аналіз і проєктування створюють надійну основу для реалізації вебдодатку та забезпечують чітку відповідність між бізнес-потребами та технічними рішеннями.

Побудовано діаграму взаємодій (рис. 2.1). Вона дозволяє чітко відобразити логіку взаємодії між компонентами та їхні функції в межах системи.

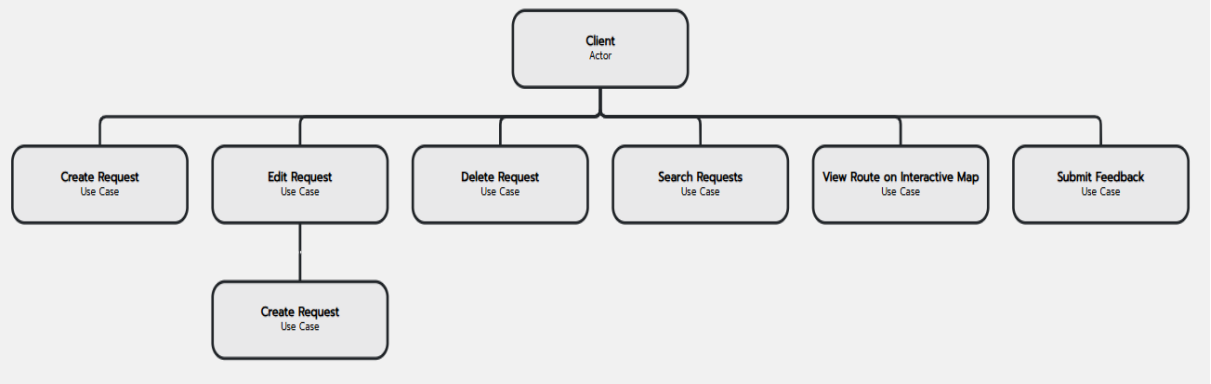

Рисунок 2.1 – Діаграма взаємодії клієнта

Сформульовані вимоги та результати проєктування забезпечать повне покриття функціоналу майбутнього вебзастосунку та створять надійну основу для його реалізації в наступних етапах. Крім функціональних і технічних аспектів, проєктування системи повинно враховувати базову масштабованість та простоту підтримки.

Таким чином, майбутня система буде відповідати потребам логістичної компанії в частині збирання заявок, інтерактивної презентації напрямків роботи та зручного каналу зв’язку з клієнтами.

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

Після формування функціональних і нефункціональних вимог до системи було здійснено глибокий аналіз існуючих інструментів і підходів, що можуть бути використані для розробки вебзастосунку логістичної компанії.

Кінцева мета полягала у створенні стабільної, масштабованої, безпечної системи, здатної обробляти заявки на перевезення, візуалізувати логістичні

маршрути, здійснювати інтерактивну взаємодію з клієнтами та забезпечувати зручне адміністрування з боку компанії.

Початково було проаналізовано найпоширеніші сучасні фреймворки та бібліотеки: React, Angular, Vue.js, Svelte. З них Vue.js було обрано лише для реалізації окремої частини − сторінки бази знань − з огляду на його простоту, можливість організувати компонентну структуру та легкість в інтеграції в існуючий статичний HTML-проєкт.

Структура фронтенду буде побудована за допомогою семантичної HTML-розмітки та компонентного підходу до стилізації із застосуванням SCSS і методології BEM. Це дозволить в майбутнього забезпечити масштабованість стилів, ізоляцію візуальних блоків, легку підтримку та розширення інтерфейсу в майбутньому.

JavaScript буде використано для реалізації інтерактивних елементів, таких як модальні вікна, форма зворотного зв’язку з динамічною валідацією, лічильники, акордеон у секції FAQ, плавна прокрутка, а також scroll spy, що визначать активні пункти навігації залежно від положення користувача на сторінці.

У процесі розробки клієнтської частини вебдодатку для логістичної компанії планується використовувати HTML, SCSS [4] та фреймворку Vue.js. Мова розмітки HTML буде застосовуватись для створення структурного каркаса вебсторінок, зокрема для реалізації таких елементів, як форма зворотного зв’язку, навігаційне меню, секції з інтерактивною картою та блоки відгуків. Для стилізації інтерфейсу передбачається використання SCSS – сучасного препроцесора CSS, який забезпечить зручність у підтримці коду завдяки використанню змінних, вкладеності та модульної структури. Основним інструментом для реалізації інтерактивної поведінки елементів інтерфейсу стане фреймворк Vue.js, що дозволить впровадити реактивність, двостороннє зв’язування даних у формах, взаємодію з сервером через API, а також динамічне керування контентом на сторінці. Такий технологічний підхід забезпечить в

майбутньому гнучкість проєкту та зручність використання для кінцевих користувачів і співробітників компанії.

Для реалізації серверної частини вебзастосунка планується використовувати платформу Node.js [5], яка забезпечить асинхронну обробку запитів та високу продуктивність. За допомогою Node.js буде організовано обробку даних, що надходять із клієнтської форми розрахунку після чого здійснюватиметься зберігання заявок у базі даних MySQL [6], а також налаштовуватиметься інтеграція з Telegram-ботом для оперативного сповіщення. Деплой клієнтської частини вебзастосунка передбачається здійснити за допомогою хостингової платформи Netlify [7], яка надає зручний інтерфейс для розгортання статичних сайтів, підтримує автоматичне оновлення після коміту в репозиторій GitHub [8], а також дозволяє реалізувати серверну логіку за допомогою Netlify Functions. Вибір саме Netlify зумовлений її простотою використання, високою швидкістю завантаження сторінок через глобальну CDN-мережу, підтримкою SCSS, Node.js-функцій, а також можливістю безкоштовного розгортання, що є важливим чинником у межах навчального та малого комерційного проєкту. Такий підхід дозволить забезпечити швидке, стабільне та доступне розміщення вебзастосунка в інтернеті.

Один із ключових функціональних елементів системи − інтерактивна карта − планується реалізуватись на основі бібліотеки Leaflet.js [9]. Було розглянуто альтернативні інструменти (Google Maps API, Mapbox), але Leaflet обрано з огляду на його відкритість, простоту інтеграції, високу продуктивність, відсутність ліцензійних обмежень, а також гнучкі можливості редагування.

Особлива увага була приділена забезпеченню безпеки. З боку клієнта буде реалізовано перевірку обов’язковості та формату полів у формі (індекси, вага, габарити), щоб запобігти надсиланню порожніх або некоректних запитів. З боку сервера − оформлення додаткової перевірки введення даних на довжину, тип, логічну коректність, а також унеможливлення виконання шкідливих SQL-

запитів. Обмін даними між клієнтом і сервером здійснюється за допомогою захищеного HTTP-з'єднання (HTTPS) із підтримкою CORS.

Побудована UML-діаграми: діаграма варіантів послідовності (рис. 2.2) демонструє основні сценарії (заповнення заявки, перегляд карти, відправка форми).

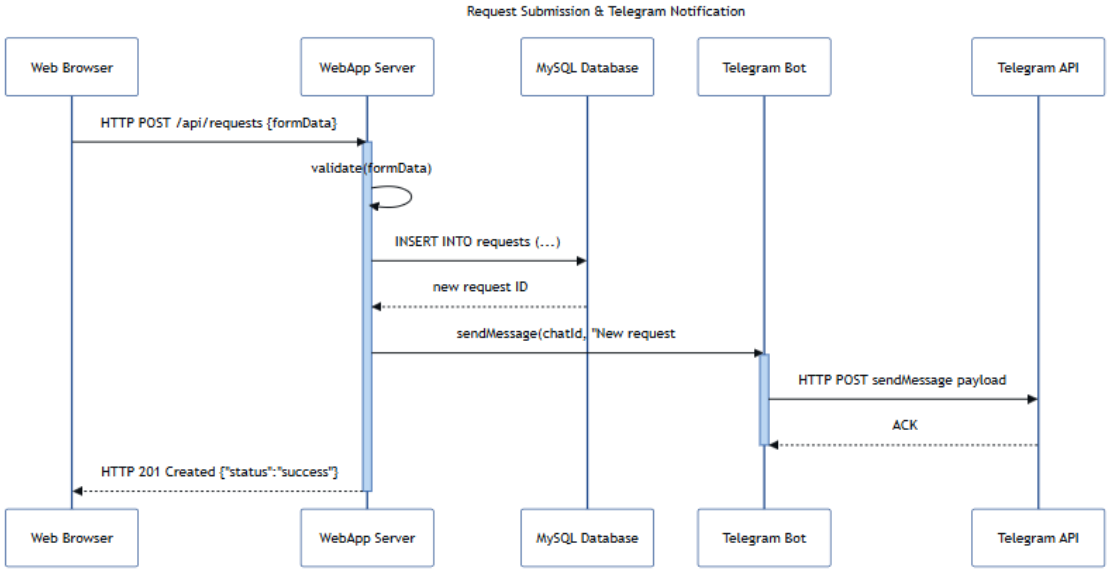

Рисунок 2.2 – Діаграма послідовності відправки та зберігання даних

Усі обрані інструменти − відкриті, стабільні, добре задокументовані й мають широку підтримку спільноти, що дозволяє розробляти продукт не лише відповідно до поточних вимог, а й з перспективою на розвиток.

Таким чином, поєднання HTML, JavaScript [10], SCSS , Vue.js, Leaflet, Node.js, Express [11] і MySQL забезпечує основу для розробки сучасного, швидкого, адаптивного й реалістично масштабованого вебдодатка, який відповідає поточним потребам логістичної компанії.

Висновки до розділу 2

Під час аналізу сфери логістики вдалося чітко окреслити ситуації, у яких майбутня система реально буде використовуватися, та сформувати конкретні вимоги до її роботи. Саме ці вимоги лягли в основу побудови загальної логіки і структури системи. Діаграми, які були створені на етапі проєктування, допомогли краще зрозуміти, як саме технічні рішення відповідають реальним потребам користувачів і процесам у компанії. Це суттєво полегшило подальше написання документації, складання сценаріїв для тестування та створення основи для розвитку системи в майбутньому. Окрім цього, було уважно розглянуто різні варіанти інструментів і обрано ті технології, які найкраще підходять для створення вебзастосунку в умовах логістичного бізнесу.

РОЗДІЛ 3
РОЗРОБКА ВЕБДОДАТКУ ДЛЯ АВТОМАТИЗАЦІЇ ОБРОБКИ ЗАЯВОК У ЛОГІСТИЧНІЙ КОМПАНІЇ З ІНТЕРАКТИВНОЮ КАРТОЮ ТА СИСТЕМОЮ ЗВОРОТНОГО ЗВ'ЯЗКУ

3.1 Практична реалізація об'єкта проєктування

Розробка вебзастосунку для логістичної компанії здійснювалась у декілька послідовних етапів, кожен з яких мав своє чітке призначення та технічне наповнення. Спочатку було визначено логічну структуру (рис. 3.1) майбутнього вебсайту, зокрема кількість сторінок, їх взаємозв'язки та типові сценарії взаємодії користувача з інтерфейсом.

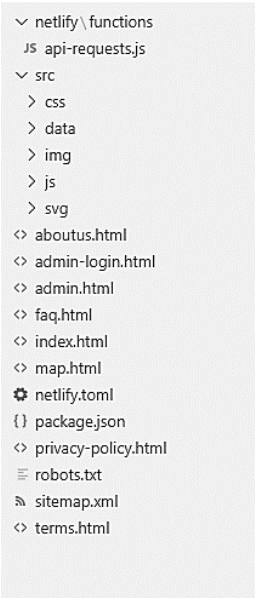

Рисунок 3.1 – Структура проєкту

Клієнтський інтерфейс реалізований із використанням семантичної HTML5-розмітки, модульного SCSS для стилів та ручного JavaScript-коду для реалізації динамічних елементів. Розмітка відповідає принципам адаптивного дизайну. SCSS-структура організована за принципами компонентності, із розділенням на змінні, міксини та окремі стилі для кожного логічного блоку. Методологія BEM дозволила уникнути конфліктів між класами та спростила

подальшу підтримку стилів. Сайт містить такі розділи як: головна сторінка, сторінка про нас, інтерактивна карта, форма заявки, база знань, часто поставлені питання (FAQ). Вся логіка переходів між секціями реалізована за допомогою якорів та плавної прокрутки.

У межах клієнтської частини вебзастосунку реалізовано інтерактивну форму «Отримати розрахунок», яка слугує основним інструментом для збору початкових даних від клієнта з метою розрахунку логістичної пропозиції. Форма має зручний адаптивний інтерфейс і поділена на логічні блоки, що дозволяє користувачеві послідовно вводити необхідну інформацію. У блоці «Маршрут доставки» передбачені поля для вказання пунктів відправлення та призначення. Розділ «Інформація про вантаж» включає: загальний об’єм вантажу (в кубічних метрах), габарити одиниці вантажу (довжина, ширина, висота), загальну вагу (в кг), кількість одиниць, тип вантажу та можливість вказати, чи є вантаж ADR (небезпечним для перевезення). Для уточнення деталей передбачене текстове поле «Коментар».

У секції «Дата подачі фури» користувач має можливість обрати бажану дату за допомогою зручного елемента вибору дати. У блоці «Контакти» реалізовані поля для введення імені, номера телефону та електронної пошти, що є необхідними для зворотного зв’язку з клієнтом. Також у формі передбачено можливість прикріпити файл, наприклад, специфікацію вантажу або фотографії. Завершується форма чек боксом з підтвердженням згоди на обробку персональних даних відповідно до політики конфіденційності, що забезпечує відповідність вимогам щодо захисту даних. Після заповнення всієї інформації користувач може натиснути кнопку «Отримати розрахунок», яка ініціює відправлення даних на сервер для подальшої обробки й формування комерційної пропозиції.

Додатково, після відправлення форми (ліст. 3.1) всі дані автоматично зберігаються в хмарну базу даних Supabase [12], що забезпечує швидкий доступ до заявок та їх централізоване зберігання. Зібрана інформація з бази надалі парситься у внутрішню адміністративну панель, де відповідальні працівники

мають змогу переглядати, фільтрувати та обробляти отримані запити. Завдяки цьому підходу ми можемо оперативно відповідати на нові заявки, підтримувати актуальний стан даних і забезпечити зручне керування логістичними процесами з боку компанії.

Лістинг 3.1 – HTML-структура адаптивної форми заявки для логістичної компанії

```
   <section class="shipping-form__section">
        <input type="text" name="pickup-location"
placeholder="Звідки" required>
    <input type="text" name="delivery-location"
placeholder="Куди" required>
  </section>
   <section class="shipping-form__section">
    <h2>Габарити вантажу</h2>
    <input type="number" name="length" placeholder="Довжина, м"
required>
    <input type="number" name="width" placeholder="Ширина, м"
required>
    <input type="number" name="height" placeholder="Висота, м"
required>
  </section>
    <input type="number" name="weight" placeholder="Вага, кг"
required>
    <input type="number" name="quantity" placeholder="Кількість
одиниць" required>
    <input type="text" name="cargo-type" placeholder="Тип
вантажу" required>
    <label><input type="checkbox" name="adr"> Це ADR
вантаж?</label>
  <textarea name="comment" placeholder="Коментар
(необов'язково)"></textarea>
  <input type="date" name="pickup-date" required>
  <section class="shipping-form__section">
    <input type="text" name="contact-name" placeholder="Ім’я"
required>
    <input type="tel" name="phone" placeholder="+380XXXXXXXXX"
required>
    <input type="email" name="email"
placeholder="email@example.com" required>
  </section>
  <label><input type="checkbox" name="consent" required>
Погоджуюсь на обробку даних</label>
  <button type="submit">Отримати розрахунок</button>
</form>
```

кінець лістингу 3.1

Ще однією з ключових функціональних особливостей розробленого вебзастосунка є інтерактивна карта, яка забезпечує візуалізацію маршруту доставки вантажу від точки А до точки Б. Вона реалізована з використанням бібліотеки Leaflet.js у поєднанні з відповідними API для картографічних даних. Користувач має можливість бачити не лише маршрут, а й актуальну дорожню ситуацію, зокрема, інформацію про погодні умови, трафік на основних автомагістралях, а також дані про дорожньо-транспортні пригоди (ДТП) та інші затримки на шляху. Така функціональність дозволяє покращити планування доставки, обирати оптимальний маршрут і завчасно враховувати зовнішні фактори, що можуть вплинути на строки виконання логістичних операцій. Інтерактивність карти підвищує прозорість процесу доставки як для компанії, так і для клієнта.

Карта, побудована за допомогою Leaflet (ліст. 3.2), була інтегрована на сторінку доставки. Вона відображає маркери основних міст Європи та України, що обслуговуються компанією. Всі координати зберігалися в окремому JavaScript-масиві. Маркери мали підказки з назвами міст, а саму карту можна було збільшувати, переміщати та масштабувати. Така візуалізація значно покращує сприйняття географії логістичних маршрутів.

Лістинг 3.2 – Ініціалізація карти

```
const cities = [
  { name: "Київ", coords: [50.45, 30.52] },
  { name: "Львів", coords: [49.84, 24.03] },
  { name: "Варшава", coords: [52.23, 21.01] },
  { name: "Берлін", coords: [52.52, 13.40] },
  { name: "Амстердам", coords: [52.37, 4.89] },
  { name: "Брюссель", coords: [50.85, 4.35] };
const map = L.map("map").setView([51.5, 14.5], 5);
L.tileLayer("https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png",
{
  attribution: "&copy; OpenStreetMap contributors"
}).addTo(map);
cities.forEach(({ name, coords }) => {
  L.marker(coords).addTo(map).bindPopup(`<b>${name}</b>`);
});
```

кінець лістингу 3.2

Окрім базового функціоналу, у застосунку реалізовано кілька покращень інтерфейсу для зацікавлення користувачів:

– анімовані лічильники, що демонструють кількість клієнтів, партнерів або виконаних рейсів; розділ з відгуками у вигляді каруселі;

– блок FAQ, в якому відповіді відкриваються за допомогою акордеону;

– плавне прокручування між секціями;

– scroll spy, що дозволяє відстежувати, яка секція наразі активна і відображає його.

Усі ці елементи створені без використання зовнішніх бібліотек, що дозволило глибше пропрацювати JavaScript-функціональність вручну, підвищивши тим самим якість коду та оптимізувавши завантаження сторінки.

Також було реалізовано функціональну сторінку бази знань (FAQ), яка дозволяє користувачеві швидко знайти відповіді на найпоширеніші запитання. Цей розділ створено з використанням фреймворку Vue.js (ліст. 3.3) та реалізовано у вигляді інтерактивного односторінкового компонента.

Лістинг 3.3 – Компонент сторінки FAQ з фільтрами, пошуком і акордеоном

```
<div class="Vcontainer" id="faqApp">
  <div class="faq-header">
    <h1 class="faq-title">База знань (FAQ)</h1>
  </div>
  <div class="faq-controls">
    <div class="faq-filters">
      <button
        v-for="cat in categories"
        :key="cat"
        :class="{ 'active-filter': selectedCategory === cat }"
        @click="selectCategory(cat)"
      >{{ cat }}</button>
    </div>
    <input type="text" v-model="searchText" class="faq-search"
placeholder="Пошук питання..." />
  </div>
  <div class="faq-counter" v-if="filteredFaqs.length">
    Знайдено: {{ filteredFaqs.length }} відповідей
  </div>
  <div class="faq-counter" v-else>
    Нічого не знайдено за вашим запитом.
  </div>
```

```
  <div v-for="(faq, index) in filteredFaqs" :key="index"
class="faq-item">
    <div class="question" :class="{ active: activeIndex === index
}" @click="toggle(index)">
      {{ formatQuestion(faq.question) }}
    </div>
    <div class="answer" :class="{ active: activeIndex === index
}">
      <p>{{ faq.answer }}</p>
      <div class="faq-views">Переглядів: {{ faq.views }}</div>
    </div>
  </div>
  <button class="scroll-top-btn" :class="{ visible: showScroll }"
    @click="scrollToTop" aria-label="Прокрутити
вгору">&#9650;</button>
</div>
```

кінець лістингу 3.3

Користувач має змогу:

– фільтрувати питання за категоріями;

– здійснювати повнотекстовий пошук;

– переглядати кількість знайдених відповідей;

– розгортати/згортати блоки відповідей у вигляді акордеону;

– переглядати лічильник переглядів кожної відповіді;

– швидко повертатись угору сторінки за допомогою кнопки прокрутки.

Важливим етапом у процесі розробки вебзастосунка стало впровадження адаптивного дизайну, що забезпечує коректне відображення інтерфейсу на пристроях із різними розмірами екранів. Завдяки використанню медіа-запитів, гнучкої сітки та можливостей SCSS вдалося реалізувати інтерфейс, який динамічно змінює свою структуру залежно від ширини екрана. Це дозволяє зручно працювати з вебзастосунком як на широкоформатних моніторах, так і на планшетах або мобільних телефонах (рис. 3.2). Основні елементи – такі як форма «Отримати розрахунок», інтерактивна карта, кнопки виклику дії, відгуки та вкладки – автоматично перебудовуються та змінюють розміри для зручності користувача. Також було враховано специфіку взаємодії з мобільними пристроями – наприклад, збільшено інтерактивні області кнопок, спрощено навігацію та реалізовано мобільне бургер-меню. Завдяки цьому вебзастосунок

залишається повноцінним інструментом для користувачів незалежно від того, яким пристроєм вони користуються, що особливо важливо у сфері логістики, де доступ до системи може здійснюватися як з офісу, так і в дорозі.

Рисунок 3.2 – Вигляд вебдодатку на мобільному пристрої

3.2 Тестування та налагодження інформаційно-комп'ютерної системи

Тестування вебзастосунка проводилося на всіх етапах життєвого циклу розробки з використанням різних типів тестування: модульного, інтеграційного, функціонального, регресійного та частково системного. Такий комплексний підхід забезпечив всебічну перевірку працездатності системи, її логічної цілісності, взаємодії між компонентами, а також стійкості до

навантажень і помилкових дій користувача. Тестування здійснювалося не лише на завершальному етапі, а й паралельно з розробкою, що дозволяло швидко виявляти та виправляти помилки, не накопичуючи технічний борг.

Модульне тестування виконувалося вручну, з фокусом на окремі функції JavaScript, що відповідали за ключову логіку: обчислення об’єму вантажу, перевірку введених значень, перемикання стану елементів інтерфейсу, роботу анімованих компонентів тощо. Важливою частиною була перевірка роботи обробників подій у формі, правильності підстановки класів активності та зміни в DOM. Для перевірки серверної частини, що працювала на базі Node.js та Express.js, проводилось тестування маршрутів REST API – насамперед, отримання та обробки запитів з форми доставки, взаємодії з MySQL, логування та обробки виняткових ситуацій. В якості інструментів застосовувались Postman для моделювання HTTP-запитів, а також Node.js-консоль для перевірки виводу.

Інтеграційне тестування охоплювало логічну взаємодію між фронтенд- та бекенд-частинами. Було протестовано передачу даних у форматі JSON з форми до сервера, коректність обробки на сервері, повернення відповіді користувачеві, збереження в базу даних та пересилання повідомлень до Telegram-бота. Особливу увагу приділено перевірці механізмів валідації як на клієнтській, так і на серверній стороні. Наприклад, перевірялося, що навіть при обхідній відправці некоректного JSON-запиту сервер зупиняє обробку й повідомляє про помилку.

Тестування сценаріїв з ADR-вантажами включало перевірку активації випадаючого списку з класами ADR після встановлення прапорця. Визначались граничні значення для кожного поля (наприклад, вага – 0, об’єм – 0,01 м³), і тестувалися сценарії, коли дані заповнено частково або з пропущеними ключовими полями. Реакція системи на помилки включала як підсвічування полів, так і виведення пояснювальних повідомлень для користувача.

Функціональне тестування здійснювалося відповідно до основних сценаріїв використання: заповнення та надсилання форми, перегляд FAQ,

навігація між секціями, масштабування карти, фільтрація запитань у базі знань. Для кожної сторінки (головна, про нас, карта, форма, FAQ, база знань) перевірялася як статична, так і динамічна поведінка. Було проведено оцінку адаптивності інтерфейсу на різних пристроях включаючи смартфони та великі монітори, у тому числі в горизонтальній орієнтації. Важливо було переконатися, що компоненти не виходять за межі контейнерів, не перекриваються та залишаються доступними.

Під час налагодження було виявлено та усунуто понад 10 помилок. Зокрема, некоректна обробка числових полів із нульовим значенням, неправильне позиціювання елементів на мобільних пристроях, дублювання елементів у слайдері відгуків, помилки у валідації email-адреси та номеру телефону, а також порушення логіки в реакції на ввімкнення поля ADR. Додатково було виправлено стилі, які не враховували крайові кейси при маленьких роздільних здатностях. Усі виправлення фіксувалися поетапно в системі контролю версій Git, що дозволяло легко відстежувати зміни.

У процесі налагодження використовувалися інструменти live-reload (наприклад, nodemon), автоперезапуск сервера, моніторинг логів і симуляція помилкових дій користувачів. Це дозволило оперативно виявляти помилки, які виникали лише в специфічних умовах, таких як повільне з’єднання або мала роздільна здатність. Особливу увагу приділено перевірці механізмів валідації як на клієнтській, так і на серверній стороні. Наприклад, перевірялося, що сервер коректно обробляє некоректні JSON-запити й повідомляє про помилку без продовження обробки. Було виявлено та усунуто понад 10 помилок, серед яких: некоректна обробка числових полів із нульовим значенням, дублювання елементів у слайдері відгуків, помилки у валідації email-адреси та номеру телефону, неправильне позиціювання елементів на мобільних пристроях, а також порушення логіки при активації поля ADR.

На завершальному етапі тестування (рис. 3.3) проводився аудит продуктивності з використанням Lighthouse [13] і Chrome DevTools [14], що дозволило виявити та усунути вузькі місця у завантаженні ресурсів і

візуалізації. Зокрема, було оптимізовано зображення, зменшено кількість запитів до сервера та впроваджено кешування статичних файлів. Також проведено перевірку на доступність відповідно до стандартів WCAG: оцінювалася контрастність кольорів, доступність з клавіатури, наявність ARIA-атрибутів і відповідність структури сторінок для екранних читалок. Усі виправлення поетапно фіксувалися в системі контролю версій Git, що дозволяло легко відстежувати зміни та відновлювати попередні стани.

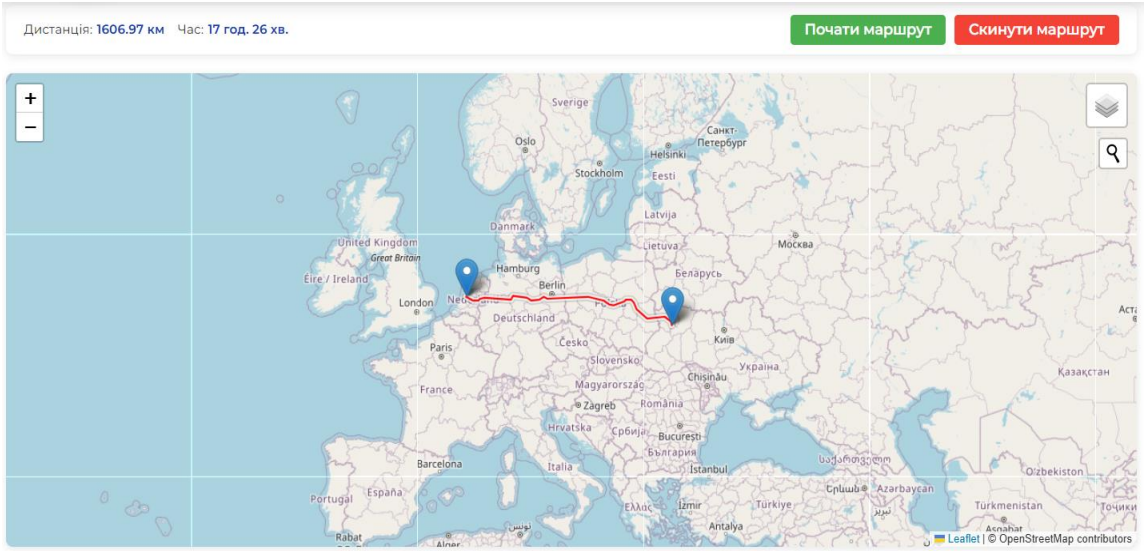

Рисунок 3.3 – Приклад тестування маршруту

У підсумку, проведене тестування забезпечило високу якість функціональних модулів, зменшення ймовірності збоїв, покращення UX і стабільності системи. Усі критичні помилки були вчасно виявлені та усунуті, що дозволило перейти до стадії розгортання застосунку з високим рівнем впевненості у його працездатності та відповідності поставленим вимогам.

3.3 Розробка бази даних

У процесі реалізації вебзастосунка для логістичної компанії особлива увага була акцентована створенню надійної системи збереження користувацьких даних. Саме база даних є ключовим компонентом

інформаційно-комп’ютерної системи, який забезпечує централізоване збереження усіх заявок, що надходять через форму на сайті, їх подальшу обробку, сортування та аналітичний доступ до інформації з боку працівників компанії. Враховуючи особливості предметної області, необхідність оперативного реагування на вхідні запити та простоту в адмініструванні, було прийнято рішення використовувати хмарну базу даних Supabase, яка базується на MySQL (рис. 3.4), але забезпечує доступ до неї через API з безпечним підключенням.

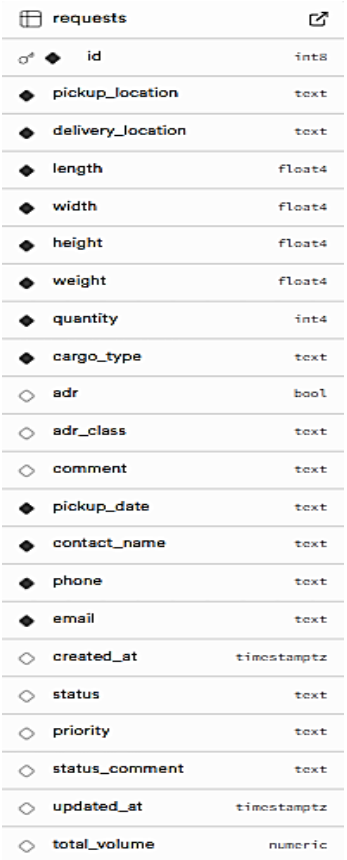

Рисунок 3.4 – Схема бази даних

База даних була спроєктована відповідно до логіки роботи основного функціоналу сайту. Коли користувач заповнює форму на головній сторінці, де вводить інформацію про маршрут доставки, габарити вантажу, його вагу, кількість одиниць, тип і клас ADR (якщо вантаж вимагає спеціальних умов перевезення), а також зазначає дату подачі транспорту та контактні дані, ці

відомості автоматично передаються на сервер. Важливо підкреслити, що всі ці дані не лише використовуються для надсилання повідомлення в Telegram-бот [15] компанії, але й паралельно зберігаються в базу даних та відображається в адмін-панелі (рис. 3.5). Це дозволяє не лише забезпечити миттєве сповіщення менеджера про нову заявку, а й зберігати всю інформацію у структурованому вигляді для подальшого аналізу, сортування, фільтрації та формування звітності.

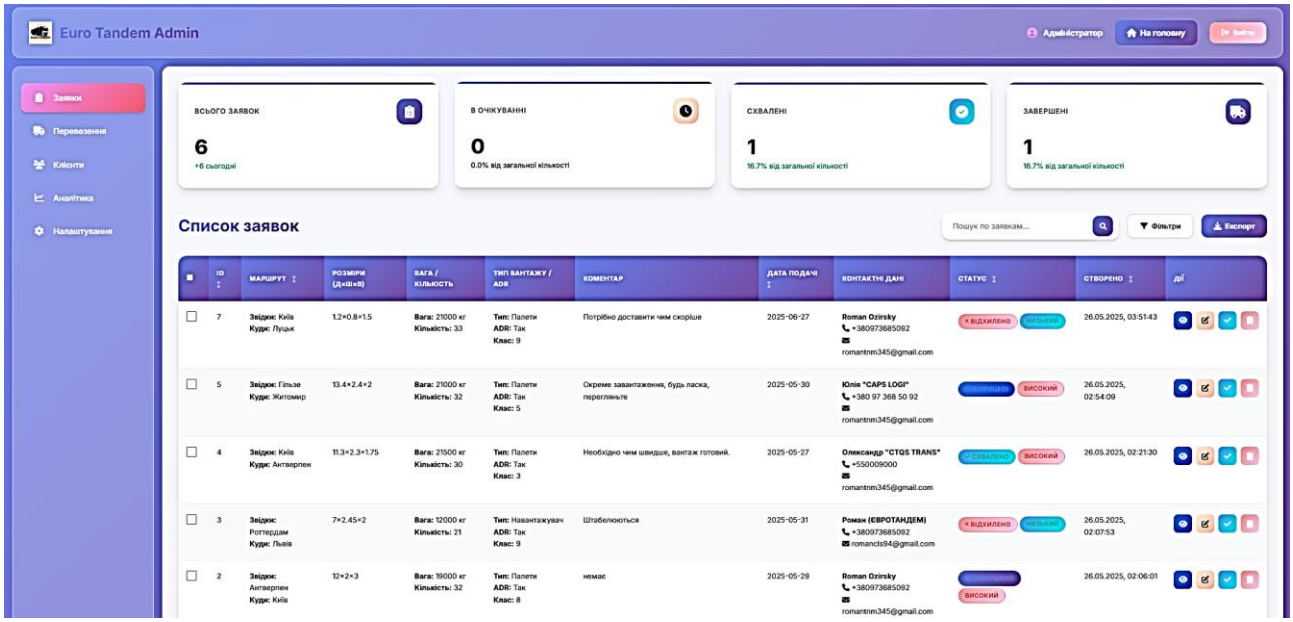

Рисунок 3.5 – Відображення адмін-панелі

Серверна частина застосунку, реалізована за допомогою Node.js, приймає POST-запити з фронтенду, виконує первинну перевірку на коректність вхідних даних, і після цього формує SQL-запит. Дані потрапляють у таблицю, структура якої враховує всі основні поля форми. Зокрема, передбачено поля для географічних назв (місце завантаження і розвантаження), числові поля для довжини, ширини, висоти, ваги, кількості одиниць, логічні поля для ADR-навантажень, текстові поля для типу вантажу та додаткових коментарів, а також поля для контактних даних користувача. Окремим полем фіксується дата створення заявки, що дозволяє проводити хронологічний аналіз або сортування за періодами. Усі дані, що зберігаються в базі, використовуються не лише для

пасивного накопичення, але й для активного управління запитами в межах закритої частини системи. Для цього реалізовано спеціальну адміністративну сторінку з маршрутом адмін, доступ до якої мають лише співробітники компанії. Доступ захищено системою авторизації, яка базується на токенах або внутрішній логіці перевірки прав користувача. На цій сторінці працівники логістичного відділу мають змогу переглядати усі заявки, які були надіслані через сайт, включаючи як нові, так і вже опрацьовані. Виведення даних реалізовано у вигляді таблиці або списку із зазначенням дати заявки, основної маршрутної інформації, параметрів вантажу та контактів клієнта. Це дозволяє швидко приймати рішення (ліст. 3.4), зв’язуватись із замовником та формувати внутрішню логістику.

Лістинг 3.4 – Скрипт динамічного завантаження з серверу

```
document.addEventListener('DOMContentLoaded', async () => {
  const errorDiv = document.getElementById('error');
  try {
    const response = await fetch('/api/requests');
    if (!response.ok) throw new Error(`Статус запиту:
${response.status}`);
    const data = await response.json();
    if (!Array.isArray(data) || data.length === 0) {
      errorDiv.textContent = 'Заявки не знайдені.';
      return;
    }
    const tbody = document.querySelector('#requests-table
tbody');
    data.forEach(req => {
      const tr = document.createElement('tr');
      tr.appendChild(Object.assign(document.createElement('td'),
{ textContent: req.id }));
      tr.appendChild(Object.assign(document.createElement('td'),
{ textContent: `${req.pickupLocation} / ${req.deliveryLocation}`
}));
      tr.appendChild(Object.assign(document.createElement('td'),
{ textContent: `${req.length}×${req.width}×${req.height}` }));
      tr.appendChild(Object.assign(document.createElement('td'),
{ textContent: `${req.weight} / ${req.quantity}` }));
      tr.appendChild(Object.assign(document.createElement('td'),
{ textContent: `${req.cargoType} / ${req.adr} / ${req.adrClass}`
}));
      tr.appendChild(Object.assign(document.createElement('td'),
{ textContent: req.comment || '' }));
```

```
      tr.appendChild(Object.assign(document.createElement('td'),
{ textContent: req.pickupDate }));
      tr.appendChild(Object.assign(document.createElement('td'),
{ textContent: `${req.contactName}, ${req.phone}, ${req.email}`
}));
      const rawTime = req.createdAt;
      const utcString = rawTime.replace(' ', 'T') + 'Z';
      const dateObj = new Date(utcString);
      const formattedTime = dateObj.toLocaleString('uk-UA', {
        year: 'numeric', month: '2-digit', day: '2-digit',
        hour: '2-digit', minute: '2-digit', second: '2-digit'
      });
   tr.appendChild(Object.assign(document.createElement('td'), {
textContent: formattedTime }));
      tbody.appendChild(tr);
    });
  } catch (err) {
    console.error(err);
    errorDiv.textContent = 'Помилка завантаження заявок: ' +
err.message;
  }
});
```

кінець лістингу 3.4

Конфігурація підключення до хмарної бази Supabase винесена в файл .env, де зберігаються URL-адреса доступу, токен автентифікації та параметри пулу з’єднань. На етапі ініціалізації сервер завантажує ці змінні середовища через пакет dotenv, після чого клієнтська бібліотека @libsql/client створює пул із заданою максимальною кількістю одночасних з’єднань, що забезпечує рівномірний розподіл навантаження та запобігає перевантаженню. Для підвищення надійності виконання SQL-запитів реалізовано механізм автоматичного повторного виконання з експоненційним збільшенням затримки між спробами: у разі помилки запит автоматично повторюється до трьох разів з поступовим наростанням інтервалу очікування. Всі події, починаючи від первинної спроби виконання запиту та завершуючись успішним записом або критичною відмовою, фіксуються централізованим логером Winston із зазначенням часу, параметрів запиту та стек-трейсу помилки. У разі системних збоїв формується сповіщення в Sentry, що забезпечує оперативний моніторинг і швидке реагування на інциденти. Така стратегія підключень і обробки

помилок гарантує стабільну роботу вебдодатку під високим навантаженням і створює прозору систему діагностики в разі непередбачених ситуацій.

Така інтеграція бази даних з адміністративною частиною сайту значно спрощує внутрішні бізнес-процеси компанії, дозволяє відмовитися від використання сторонніх CRM-систем або таблиць у ручному режимі. Усі дані доступні в режимі реального часу, а структура бази спроєктована таким чином, щоб у майбутньому легко розширювати її без порушення існуючої логіки. Наприклад, можуть бути додані нові поля для статусу заявки, коментарів менеджера, історії змін або результатів обробки.

Окремо слід зазначити, що вибір Supabase як хмарної бази даних виявився вдалим з погляду продуктивності та гнучкості. Сервіс має низьку затримку при доступі, підтримує одночасну обробку запитів і дозволяє безпечно працювати з даними з будь-якої точки світу. Це робить систему зручною для розгортання на будь-якому хостингу, а також відкриває можливість розширення функціоналу, наприклад, додавання push-сповіщень, синхронізації з іншими API або підключення зовнішніх аналітичних інструментів.

Під час створення бази були дотримані принципи нормалізації, що дозволяє уникнути дублювання даних, спростити оновлення інформації та мінімізувати ризики виникнення помилок при подальшому розширенні. Уся логіка запису, читання та валідації даних реалізована в окремому шарі серверної частини, що відповідає принципам MVC-архітектури й забезпечує хорошу масштабованість.

Таким чином, база даних не лише виконує роль пасивного сховища, а є активним ядром усієї інформаційної системи. Вона дозволяє формувати єдиний цифровий простір між клієнтами, що залишають заявки, та менеджерами компанії, які відповідають за логістику та зв’язок із замовниками (ліст. 3.5). Реалізована структура БД, її інтеграція із застосунком, захищеність, швидкодія та зручність використання створюють надійну основу для стабільної роботи всієї системи і дозволяють впевнено розвивати її в майбутньому.

Лістинг 3.5 – Серверна логіка обробки заявок та інтеграція з БД і Telegram

```
const { createClient } = require('@libsql/client');
const db = createClient({
  url: process.env.Supabase_DATABASE_URL,
  authToken: process.env.Supabase_AUTH_TOKEN,
});
exports.handler = async function(event) {
  if (event.httpMethod === 'POST') {
    const data = JSON.parse(event.body);
    await db.execute({
      sql: `INSERT INTO requests (...) VALUES (...)`,
      args: [...],
    });
    await fetch(`https://api.telegram.org/bot.../sendMessage`, {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ chat_id: ..., text: '...' }),
    });
    return { statusCode: 200,
      body: JSON.stringify({ success: true }),
    };
  }
  if (event.httpMethod === 'GET') {
    const result = await db.execute({ sql: `SELECT * FROM
requests`, args: [] });
```

кінець лістингу 3.5

3.4 SEO інформаційно-комп’ютерної системи

Однією з важливих складових у розробці сучасного вебзастосунка є реалізація ефективної пошукової оптимізації, що дозволяє підвищити видимість ресурсу в результатах пошуку та забезпечити стабільне залучення нових користувачів без додаткових витрат на рекламу. Для логістичної компанії це особливо актуально, оскільки більшість клієнтів шукають послуги з перевезення вантажів саме через пошукові системи. Саме тому в межах створення інформаційно-комп’ютерної системи було передбачено цілий комплекс заходів, спрямованих на покращення SEO-індексації сайту, збільшення його релевантності та зручності для пошукових роботів.

Оптимізація розпочалася з налаштування базової структури сторінок. Кожна сторінка містить унікальні заголовки, опис і ключові слова, що

відображають її зміст і відповідають пошуковим запитам потенційних клієнтів. Було враховано семантичну структуру HTML-документів, яка включає правильне використання заголовків, абзаців, списків, блоків змісту та інших елементів. Відповідно, вся розмітка сайту побудована за принципами логічної ієрархії, що полегшує роботу індексаторів та покращує продуктивність кожного файлу та зображення. Додатково для полегшення індексації пошуковими системами було створено файл з картою сайту, який містить перелік усіх сторінок ресурсу (рис. 3.6).

```xml
<?xml version="1.0" encoding="UTF-8"?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
  <url>
    <loc>https://eurotandem.com/</loc>
  </url>
  <url>
    <loc>https://eurotandem.com/aboutus.html</loc>
  </url>
  <url>
    <loc>https://eurotandem.com/map.html</loc>
  </url>
  <url>
    <loc>https://eurotandem.com/faq.html</loc>
  </url>
  <url>
    <loc>https://eurotandem.com/terms.html</loc>
  </url>
  <url>
    <loc>https://eurotandem.com/privacy-policy.html</loc>
  </url>
</urlset>       You, last week • Sitemap and robots
```

Рисунок 3.6 – Вміст файлу карти сайту

Це дозволяє роботам Google і Bing швидко знаходити нові матеріали. Також був сформований файл robots.txt [16] (рис. 3.7), у якому вказано, які розділи ресурсу мають бути відкриті для індексації, а які слід виключити з аналізу. Це особливо важливо для захисту службових або внутрішніх сторінок, зокрема сторінки адміністративного інтерфейсу. Певні покращення були впроваджені з погляду швидкості завантаження ресурсу.

```
1   User-agent: *
2   Disallow: /api/
3   Disallow: /src/
4   Sitemap: https://eurotandem.com/sitemap.xml
```

Рисунок 3.7 – Файл robots.txt

Для покращення сприйняття контенту пошуковими роботами кожної сторінки додано структуровані дані у форматі JSON-LD [17], що описують ключові сутності сайту: організацію (Organization), навігацію (BreadcrumbList), сторінку (WebPage) та розділ із запитаннями (FAQPage) (ліст. 3.6). Це дозволяє пошуковим системам створювати розширені сніппети й підвищує клікабельність посилань у видачі.

Лістинг 3.6 – Приклад підключення JSON-LD мікророзмітки для SEO

```
<script type="application/ld+json">
{
  "@context": "https://schema.org",
  "@graph": [
    {
      "@type": "Organization",
      "name": "Euro Tandem",
      "url": "https://eurotandem.netlify.app",
      "logo": "https://eurotandem.netlify.app/img/logo.png"
    },
    {
      "@type": "BreadcrumbList",
      "itemListElement": [
        { "@type": "ListItem", "position": 1, "name": "Головна",
"item": "https://eurotandem.netlify.app/" },
        { "@type": "ListItem", "position": 2, "name": "FAQ",
"item": "https://eurotandem.netlify.app/faq" }
      ]
    },
      {
      "@type": "FAQPage",
      "mainEntity": [
        {
          "@type": "Question",
          "name": "Як оформити заявку?",
          "acceptedAnswer": {
            "@type": "Answer",
            "text": "Заповніть форму на головній сторінці, вказавши
маршрут, параметри вантажу та контактні дані."
          }
        }
```

```
      ]
    }]
}
</script>
```

кінець лістингу 3.6

Оптимізація зображень, використання кешування статичних ресурсів, мінімізація стилів та сценаріїв дозволили зменшити час очікування користувачів та покращити технічну оцінку сайту за результатами аудиту в інструментах оцінки продуктивності. Це позитивно впливає на ранжування сайту, адже пошукові системи надають перевагу ресурсам із високою швидкістю завантаження. Було враховано й фактор мобільної адаптивності. Весь інтерфейс розроблявся із урахуванням потреб мобільних користувачів, що дозволяє забезпечити однаково комфортне використання як на десктопних, так і на мобільних пристроях. Це важливо не лише для зручності користувача, але і для досягнення високого показника Mobile-Friendly у Google.

Особлива увага приділялася оптимізації контенту (рис. 3.8). Наповнення сайту текстовими блоками, що включають ключові слова, пов’язані з вантажоперевезенням, логістикою, країнами Європи, послугами перевезення, забезпечує покращену релевантність сторінок. Такий підхід дозволяє досягти високих позицій у видачі за запитами типу «перевезення вантажів по Німеччині», «логістична компанія з ADR» або «доставка по Європі». В межах технічної реалізації також було враховано доступність ресурсу. Всі елементи інтерфейсу супроводжуються текстовими описами, мітками та атрибутами, що дозволяє використовувати сайт людьми з особливими потребами. Це, у свою чергу, враховується пошуковими системами при оцінюванні ресурсу та сприяє його вищому рейтингу. Крім цього, для аналітики було підключено інструменти Google Search Console та Google Analytics. Вони дають змогу відстежувати ключові запити, за якими користувачі потрапляють на сайт, ефективність сторінок, показник відмов, час перебування користувача на сайті та джерела трафіку. На основі цих даних можна здійснювати подальше коригування

контенту, структури та функціоналу сайту, щоб ще більше відповідати запитам цільової аудиторії.

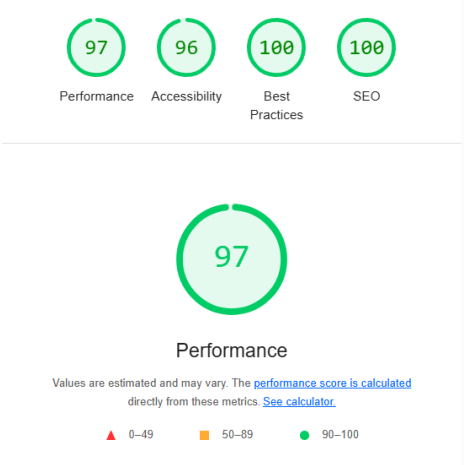

Рисунок 3.8 – Перевірка продуктивності та SEO за допомогою інструменту Lighthouse

Таким чином, реалізація SEO-оптимізації у межах інформаційно-комп’ютерної системи дозволила забезпечити відповідність ресурсу сучасним стандартам індексації, покращити його видимість у пошукових системах, а також підвищити довіру користувачів до бренду логістичної компанії. Всі вжиті заходи мають довгостроковий ефект і сприяють сталому зростанню кількості заявок без додаткових витрат на рекламне просування.

3.5 Розробка документації для програмного продукту

У межах реалізації інформаційно-комп’ютерної системи була створена супровідна документація, що охоплює технічні та користувацькі аспекти її інсталяції, налаштування і використання. Документація включає опис вимог до середовища, інструкції з розгортання, довідкові матеріали для кінцевих користувачів та рекомендації щодо підтримки і обслуговування системи.

Мінімальні системні вимоги до клієнтської частини вебдодатку передбачають використання сучасного браузера з підтримкою JavaScript

(наприклад, Google Chrome, Mozilla Firefox або Safari), двоядерного процесора, 2 ГБ оперативної пам’яті та стабільного підключення до Інтернету. На стороні сервера система функціонує на платформі Node.js (версія 18 або вище) з використанням хмарного сховища даних Supabase. Рекомендована операційна система – Ubuntu 20.04 або інша Unix-подібна система, мінімальний об’єм оперативної пам’яті – 512 МБ, дисковий простір – не менше 100 МБ.

Процедура встановлення програмного продукту включає клонування репозиторію з GitHub, налаштування змінних середовища для підключення до бази даних та Telegram API, встановлення залежностей через менеджер пакетів npm, запуск клієнтської та серверної частин, а також розгортання на обраному хостингу (наприклад, Netlify [15], Render або інші платформи з підтримкою Node.js функцій). У процесі налаштування також враховується створення структури бази даних у Supabase, реалізація запитів на збереження і отримання даних, а також забезпечення безпечної передачі інформації до Telegram-бота. Для кінцевих користувачів реалізовано довідкову інформацію у вигляді інтерактивного розділу «База знань» (FAQ), який містить категоризовані відповіді на найбільш поширені питання. Вбудований пошук і фільтри дозволяють швидко знайти відповідну інформацію, що підвищує зручність користування сайтом та зменшує потребу у зверненні до служби підтримки. Усі розділи бази знань оновлюються вручну через внутрішню систему керування вмістом.

Інтерфейс адміністратора, який є частиною системи, забезпечує доступ до збережених заявок користувачів. Сторінка доступна виключно авторизованим працівникам компанії. Вона дозволяє переглядати вхідні дані, фільтрувати заявки за ключовими полями, аналізувати час надходження і автоматично підраховувати об’єм вантажу. Для розробників та технічного персоналу всі зміни коду документуються у системі контролю версій Git. Це дозволяє відслідковувати хронологію змін, швидко виявляти причини помилок та відновлювати попередні версії програмного забезпечення. Окремий README-

файл містить базові інструкції для локального запуску, внесення змін до системи та її повторного деплою.

Таким чином, документація до програмного продукту охоплює як інструкції для користувачів, так і повний опис процедур встановлення, розгортання та супроводу системи. Також надає можливість її ефективного використання, масштабування в майбутньому та підтримки без участі основного розробника.

Висновки до розділу 3

У цьому розділі описано практичну реалізацію всіх компонентів системи. Розробку адаптивного інтерфейсу з інтерактивною картою та формою зворотнього зв’язку, налаштування серверної логіки й бази даних для обробки заявок та інтеграцію з Telegram-ботом для оперативних сповіщень. Також проведено оптимізацію для пошукових систем. Тестування показало стабільну роботу вебзастосунка, відповідність його функціоналу вимогам та готовність до впровадження. Також було розроблено технічну документації, щодо роботи з системою.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було досягнуто поставленої мети – розроблено вебдодаток для автоматизації обробки заявок у логістичній компанії з інтерактивною картою та системою зворотного зв’язку, що забезпечує централізоване зберігання даних, динамічну валідацію запитів і оперативне інформування менеджерів.

Під час проведеного аналізу предметної області було виявлено ключові проблеми в роботі логістичних компаній: ручну обробку заявок, розпорошення даних та слабку інтеграцію з клієнтами. Порівняння наявних рішень і бізнес-процесів допомогло чітко визначити функції, які потребують автоматизації, та сформувати основу для технічного завдання вебдодатку.

Сформовані вимог до системи базуються на реальних сценаріях взаємодії користувачів із сервісом. Визначено основні функціональні блоки, передбачено вимоги до продуктивності, безпеки, масштабованості та UI/UX, що стало базою для побудови архітектури та подальшої реалізації проєкту.

Було проведено аналіз та порівняння існуючих інструментів розробки на сучасному ринку. З огляду на цілі проєкту, було обрано стек HTML, SCSS, JavaScript, Vue.js, Leaflet, Node.js, Express і MySQL – як стабільну, масштабовану та добре підтримувану основу для побудови сучасного вебдодатку.

Розроблено логістичний вебдодаток, включаючи реалізацію адаптивного інтерфейсу з інтерактивною картою, форму заявки з валідацією, динамічних компонентів та логіки обробки на сервері. Застосунок функціонально охоплює усі основні бізнес-потреби компанії та є готовим до розширення і масштабування.

Проведене тестування вебдодатка охоплювало модульні, інтеграційні та функціональні рівні. Перевірено валідність даних, роботу API, відображення інтерфейсу на різних пристроях. Усі виявлені помилки було своєчасно усунуто,

що дозволило досягти стабільної роботи системи на завершальному етапі розробки.

Спроєктовано та розроблено базу даних, яка дозволила забезпечити структуроване зберігання всіх даних із форм. Реалізовано інтеграцію з MySQL, що дає змогу дієво зберігати, фільтрувати та аналізувати заявки в адміністративному інтерфейсі із забезпеченням захисту й доступу лише авторизованим працівникам.

Оптимізація системи для SEO включала поліпшення швидкодії, мінімізацію ресурсів, правильне структурування HTML-розмітки, додавання метаданих і забезпечення доступності. У результаті підвищено рейтинг сторінок у пошукових системах і поліпшено користувацький досвід.

Розроблена документація охопила технічні та користувацькі аспекти. Складено інструкції з розгортання, адміністрування, структури даних і API. Це гарантує простоту впровадження рішення у робоче середовище та полегшує його підтримку надалі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Logistics-optimization. URL: https://indewise.com/logistics-optimization/ (дата звернення: 28.02.2025).

2. Оцінка ефективності логістичних систем: методологія та практика. URL: https://cargofy.ua/uk/blog/ocinka-efektivnosti-logistichnih-sistem (дата звернення: 28.02.2025).

3. Vue.js. The Progressive JavaScript Framework. URL: https://vuejs.org/guide/introduction (date of access: 02.03.2025).

4. Sass. Syntactically Awesome Style Sheets. URL: https://sass-lang.com/ (date of access: 04.03.2025).

5. Node.js – Run JavaScript Everywhere. URL: https://nodejs.org/en (date of access: 05.03.2025).

6. MySQL. Developer Zone. URL: https://dev. mysql.com/ (date of access: 07.03.2025).

7. Netlify Deploy Previews – Share, Review, & Gather Feedback on Web Projects. URL: https://www.netlify.com/platform/core/deploy-previews (date of access: 09.03.2025).

8. Build software better, together. GitHub. URL: https://github.com/codespaces (date of access: 10.03.2025).

9. Leaflet – an open-source JavaScript library for interactive maps. URL: https://leafletjs.com/ (date of access: 12.03.2025).

10. JavaScript. MDN. Web Docs. URL: https://developer.mozilla.org/docs/Web/JavaScript (date of access: 14.03.2025).

11. Express – Node.js web application framework. URL: https://expressjs.com/ (date of access: 15.03.2025).

12. Supabase. The Open Source Firebase Alternative. URL: https://supabase.com/ (date of access: 25.03.2025).

13. Lighthouse. Chrome for Developers. URL: https://developer.chrome.com/docs/lighthouse (date of access: 27.04.2025).

14. PageSpeed Insights. URL: https://pagespeed.web.dev/ (date of access: 27.04.2025).

15. Telegram APIs. URL: https://core.telegram.org/ (date of access: 28.04.2025).

16. Create and Submit a robots.txt File. Google Search Central. Documentation. Google for Developers. URL: https://developers.google.com/search/docs/crawling-indexing/robots/create-robots-txt?hl=en (date of access: 29.04.2025).

17. JSON-LD – for Linked Data. URL: https://json-ld.org/ (date of access: 01.05.2025).

ДОДАТКИ

Додаток А - Апробація

Міністерство освіти і науки України
Волинська обласна рада
Луцька міська рада
Луцький національний технічний університет
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
Національний університет «Львівська політехніка»
Військовий інститут телекомунікацій та інформатизації імені Героїв Крут (м. Київ)
Тернопільський національний педагогічний університет імені В. Гнатюка
Рівненський державний гуманітарний університет
Навчально-науковий інститут «Українська інженерно-педагогічна академія» Харківського національного університету ім. В.Н. Каразіна
Люблінська політехніка (Польща)
Фрайберзька гірнича академія (Німеччина)
Гронінгенський університет (Нідерланди)
Кавказький університет (Грузія)
Політехнічний університет Браганси (Португалія)

ТЕЗИ ДОПОВІДЕЙ
X Міжнародної науково-практичної конференції з проблем вищої освіти і науки «Інформаційні технології в освіті, науці і виробництві (ІТОНВ-2025)

23-24 травня 2025 р.

Луцьк 2025

УДК 004.4

АВТОМАТИЗАЦІЯ ОБРОБКИ ЗАЯВОК У ЛОГІСТИЧНІЙ КОМПАНІЇ ЗА ДОПОМОГОЮ ВЕБЗАСТОСУНКУ З ІНТЕРАКТИВНОЮ КАРТОЮ ТА БАЗОЮ ДАНИХ

Ліщина Наталія Миколаївна
Луцький національний технічний університет, доцент кафедри інженерії програмного забезпечення, n.lishchyna@lutsk-ntu.com.ua
Озірський Роман Миколайович
Луцький національний технічний університет, студент групи ІПЗ-41, ozirsky.r1807@lntu.edu.ua

AUTOMATION OF REQUEST PROCESSING IN A LOGISTICS COMPANY USING A WEB APPLICATION WITH AN INTERACTIVE MAP AND DATABASE

Nataliia Lishchyna
Lutsk National Technical University, Associate Professor at the Department of Software Engineering, n.lishchyna@lutsk-ntu.com.ua
Roman Ozirskyi
Lutsk National Technical University, Student of the Group IPZ-41, ozirsky.r1807@lntu.edu.ua

The paper outlines the rationale for developing a web application to automate request processing in a logistics company. The system enables fast order submission, route visualization, and centralized data storage, which improves operational efficiency and enhances customer service in freight transportation.

Keywords: *logistics, automation, transportation, requests, route, database, data processing, customer service.*

Активне впровадження цифрових технологій, зокрема в сфері автоматизації та обробки даних, відкриває нові можливості для підвищення ефективності логістичних процесів. З огляду на зростаючі вимоги до швидкості обробки замовлень, прозорості взаємодії та комфорту клієнтів, створення спеціалізованих вебзастосунків є раціональним рішенням. Застосування інтерактивних карт і цифрових форм подання заявок істотно полегшує роботу диспетчерів, забезпечує точнішу координацію маршрутів і підвищує якість сервісу.

У цьому контексті особливої актуальності набуває впровадження інструментів, що дозволяють швидко та зручно

подавати логістичні запити в автоматизованому режимі. Такий підхід не лише підвищує якість обслуговування, а й зменшує навантаження на персонал і мінімізує помилки, що виникають під час ручного опрацювання інформації.

У сучасних умовах бізнес-середовища, де час обробки запиту часто визначає рівень клієнтської задоволеності, вебзастосунки стають невід'ємним елементом конкурентоспроможності компаній. Зокрема, інтерфейс для подачі заявок, який забезпечує точне введення, візуалізацію маршрутів та зворотний зв'язок у реальному часі, є важливим для логістичних процесів. Створення вебзастосунку для логістичної компанії є логічним кроком у напрямку спрощення процесу подання заявок, підвищення точності обробки інформації та зручності для користувачів. Гнучкий формат введення дозволяє враховувати декілька логістичних точок, супутні умови транспортування та додаткові інструкції, що робить систему придатною для різноманітних сценаріїв доставки. Це не лише зменшує ймовірність помилок, а й підвищує рівень обслуговування і контроль над кожною заявкою. Інтерфейс майбутнього вебзастосунку доцільно передбачити інтуїтивно зрозумілим і адаптованим до різних типів пристроїв, що дозволить користувачам легко орієнтуватися в системі без необхідності попереднього навчання. Такий підхід сприятиме ефективному залученню клієнтів, оперативній роботі персоналу та забезпеченню зручного доступу до ключових функцій незалежно від місця чи часу використання. Зручність і доступність інтерфейсу є важливою умовою для досягнення високої швидкості обробки запитів і узгодженості дій між усіма учасниками логістичного процесу. Однією з важливих функціональних переваг такої системи є візуалізація маршрутів, що дає змогу покращити планування перевезень, уникнути неточностей у координації й підвищити загальну прозорість логістичної діяльності. Інтеграція з каналами оперативного сповіщення може значно

скоротити час реагування, забезпечуючи миттєву передачу нових заявок відповідальним особам.

Окрім зручного процесу подання інформації, система повинна містити засоби аналітичного опрацювання даних. Щоб вона дозволяла формувати звіти за визначені періоди, аналізувати динаміку обробки замовлень, відстежувати статуси виконання, виявляти повторні звернення клієнтів та оцінювати навантаження на логістичні ресурси. Такі можливості сприятимуть прийняттю обґрунтованих управлінських рішень та покращенню операційної ефективності. За потреби, дані можуть бути адаптовані для подальшого використання в інших системах обліку чи аналітики, що дозволяє інтегрувати систему в ширший інформаційний контекст діяльності підприємства.

Доцільність впровадження автоматизованої системи для обробки логістичних заявок підтверджується низкою важливих факторів: зменшенням часу на обробку запитів, підвищенням прозорості операцій, створенням надійного цифрового архіву, покращенням планування маршрутів та оперативним інформуванням відповідальних осіб. Система позитивно впливає як на внутрішні бізнес-процеси, так і на зовнішню взаємодію з клієнтами, формуючи сучасний та технологічний імідж компанії. У довгостроковій перспективі функціонал системи може бути розширений за рахунок додаткових можливостей, таких як автоматичний розподіл завдань, розрахунок часу доставки, просторове групування логістичних точок, підтримка кількох мов, а також інтеграція з внутрішніми інформаційними системами підприємства.

Розроблена система виступає як комплексний інструмент, що охоплює ключові етапи логістичного циклу – від подання заявки до аналізу результатів її обробки. Такий підхід дозволяє гармонізувати внутрішні процеси компанії, усунути дублювання функцій між працівниками, а також підвищити прозорість і контрольованість на кожному етапі взаємодії з клієнтом. Інтерфейс доцільно проектувати з урахуванням потреб кінцевого користувача, що сприятиме зниженню бар'єру входу

та зменшенню потреби в додатковому навчанні персоналу. Функціональне наповнення рішення відповідає сучасним вимогам логістичного ринку, де гнучкість, швидкість і точність виконання відіграють вирішальну роль. Завдяки автоматизованому збору та обробці даних компанія отримує можливість оперативно реагувати на зміни в обсягах замовлень, краще планувати навантаження на транспортні засоби та швидше ухвалювати управлінські рішення. Крім оперативної вигоди, система закладає фундамент для довготривалих стратегічних переваг — можливості масштабування під нові регіони чи клієнтські сегменти, інтеграцію з іншими цифровими платформами, накопичення та обробку аналітичних даних для прогнозування попиту. Усе це формує передумови для побудови сталої, сучасної та конкурентоспроможної логістичної моделі ведення бізнесу.

Такий вебзастосунок можна розглядати як приклад ефективного цифрового рішення, що здатне забезпечити автоматизацію ключових процесів, покращення обслуговування клієнтів та загальне підвищення ефективності управління логістикою. Її впровадження сприятиме підвищенню ефективності компанії, зниженню витрат і забезпеченню контролю над всіма етапами обробки заявки.

Список використаних джерел

1. Vue.js. *Vue.js – The Progressive JavaScript Framework | Vue.js*. URL: https://vuejs.org/ (дата звернення: 30.04.2025).
2. Node.js – Run JavaScript Everywhere. *Node.js – Run JavaScript Everywhere*. URL: https://nodejs.org/en (дата звернення: 01.05.2025).
3. MySQL: MySQL Documentation. *MySQL: Developer Zone*. URL: https://dev.mysql.com/doc/ (дата звернення: 07.05.2025).
4. Telegram bot API. *Telegram APIs*. URL: https://core.telegram.org/bots/api (дата звернення: 03.05.2025).
5. Leaflet – an open-source JavaScript library for interactive maps. *Leaflet – a JavaScript library for interactive maps*. URL: https://leafletjs.com/ (дата звернення: 02.05.2025).