ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Платформа оптимізації маршрутів для сервісів доставки «останньої милі»»

на здобуття освітнього ступеня магістр
зі спеціальності 124 Системний аналіз
(код, найменування спеціальності)
освітньо-професійної програми Інтелектуальні системи управління
(назва)

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

_____ *Марк ФЕДИКОВИЧ*
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав: здобувач(ка) вищої освіти гр. САДМ-61
Марк ФЕДИКОВИЧ
Ім'я, ПРІЗВИЩЕ

Керівник: Ровіл НАФЄЄВ
науковий ступінь, вчене звання
Ім'я, ПРІЗВИЩЕ

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 магістр

Спеціальність 124 Системний аналіз

Освітньо-професійна програма Інтелектуальні системи управління

ЗАТВЕРДЖУЮ
Завідувач кафедрою ІСТ
_____Каміла СТОРЧАК
«____» _____2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Федиковичу Марку Михайловичу

(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: Платформа оптимізації маршутів для сервісів доставки «останньої милі»

керівник кваліфікаційної роботи Ровіл НАФЄЄВ, доцент,
(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «____» ___20_____р. №

2. Строк подання кваліфікаційної роботи «___» _____2025 р.

3. Вихідні дані до кваліфікаційної роботи:

1.Аналіз сучасних підходів до оптимізації маршрутів доставки

2. Розробка архітектури платформи оптимізації маршрутів

3. Реалізація та тестування програми

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

1. Дослідження сучасного стану доставки
2. Огляд методів покращення показників
3. Аналіз результатів платформи та впровадження її в реальний світ

5. Перелік ілюстративного матеріалу: *презентація*

6. Дата видачі завдання «___» _____2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір технічної літератури		
2	Дослідження реальних показників		
3	Дослідження методів їх покращення		
4	Результати впровадження платформи		
5	Висновки по роботі		
6	Розробка демонстраційних матеріалів, доповідь		
7	Оформлення магістерської роботи		

Здобувач(ка) вищої освіти _____ Марк ФЕДИКОВИЧ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

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

РЕФЕРАТ

Кваліфікаційна робота: 94 с., 2 рис., 7 табл., 50 джерела.

Об'єкт дослідження - процес доставки товарів кінцевим споживачам у рамках логістики останньої милі.

Предмет дослідження - методи та алгоритми оптимізації маршрутів доставки з урахуванням множини обмежень та критеріїв ефективності.

Мета роботи - розробка комплексної веб-платформи для автоматизованого планування та оптимізації маршрутів доставки, що забезпечує зниження операційних витрат та покращення якості обслуговування клієнтів.

Методи дослідження: аналіз предметної області та існуючих рішень, методи комбінаторної оптимізації, евристичні та метаевристичні алгоритми маршрутизації, об'єктно-орієнтоване проєктування програмних систем, методи тестування та валідації програмного забезпечення.

У роботі проведено комплексний аналіз сучасних підходів до оптимізації маршрутів доставки, досліджено особливості логістики останньої милі та її ключові виклики. Здійснено огляд існуючих методів та алгоритмів оптимізації маршрутів, включаючи точні, евристичні та метаевристичні підходи. Проаналізовано сучасні платформи та системи управління доставкою, виявлено їх переваги та обмеження.

Розроблено детальну специфікацію функціональних вимог до платформи оптимізації маршрутів, що охоплює потреби різних категорій користувачів. Спроєктовано мікросервісну архітектуру системи, що забезпечує масштабованість, надійність та можливість незалежного розвитку компонентів. Обрано математичні моделі задачі маршрутизації транспортних засобів з часовими вікнами та розроблено гібридний алгоритм оптимізації, що поєднує переваги різних методів.

Реалізовано веб-платформу з використанням сучасних технологій та фреймворків, включаючи Django для серверної частини, React для клієнтського інтерфейсу та PostgreSQL для управління даними. Розроблено основні модулі системи: управління замовленнями, управління флотом, оптимізації маршрутів, відстеження в реальному часі та аналітики.

Проведено комплексне тестування платформи на синтетичних та реальних даних, що підтвердило її ефективність та надійність. Результати тестування демонструють зниження загальної відстані доставки на 15-25% порівняно з неоптимізованими маршрутами, підвищення продуктивності кур'єрів на 20-30% та покращення дотримання часових вікон доставки до 95%.

Ключові слова. ОПТИМІЗАЦІЯ МАРШРУТІВ, ЛОГІСТИКА ОСТАННЬОЇ МИЛІ, ЗАДАЧА МАРШРУТИЗАЦІЇ ТРАНСПОРТНИХ ЗАСОБІВ, ЕВРИСТИЧНІ АЛГОРИТМИ, ВЕБ-ПЛАТФОРМА, УПРАВЛІННЯ ДОСТАВКОЮ, МІКРОСЕРВІСНА АРХІТЕКТУРА, ЧАСОВІ ВІКНА, АЛГОРИТМ КЛАРКА-РАЙТА, ГЕОКОДУВАННЯ

ABSTRACT

Qualification work: 94 p., 2 fig., 7 tables, 50 sources.

Object of research - the process of delivering goods to end consumers within the framework of last mile logistics.

Subject of research - methods and algorithms for delivery route optimization considering multiple constraints and efficiency criteria.

Purpose of work - development of a comprehensive web platform for automated delivery route planning and optimization that ensures reduction of operational costs and improvement of customer service quality.

Research methods: domain analysis and review of existing solutions, combinatorial optimization methods, heuristic and metaheuristic routing algorithms, object-oriented software system design, software testing and validation methods.

The work conducted a comprehensive analysis of modern approaches to delivery route optimization and examined the features of last mile logistics and its key challenges. An overview of existing methods and algorithms for route optimization was performed, including exact, heuristic, and metaheuristic approaches. Modern platforms and delivery management systems were analyzed, revealing their advantages and limitations.

Detailed functional requirements specification for the route optimization platform was developed, covering the needs of various user categories. A microservice architecture of the system was designed to ensure scalability, reliability, and the ability to independently develop components. Mathematical models of the vehicle routing problem with time windows were selected, and a hybrid optimization algorithm combining the advantages of different methods was developed.

A web platform was implemented using modern technologies and frameworks, including Django for the server side, React for the client interface, and PostgreSQL

for data management. The main system modules were developed: order management, fleet management, route optimization, real-time tracking, and analytics.

Comprehensive testing of the platform on synthetic and real data was conducted, confirming its effectiveness and reliability. Test results demonstrate a reduction in total delivery distance by 15-25% compared to non-optimized routes, an increase in courier productivity by 20-30%, and improved compliance with delivery time windows up to 95%.

Key Words. ROUTE OPTIMIZATION, LAST MILE LOGISTICS, VEHICLE ROUTING PROBLEM, HEURISTIC ALGORITHMS, WEB PLATFORM, DELIVERY MANAGEMENT, MICROSERVICE ARCHITECTURE, TIME WINDOWS, CLARKE-WRIGHT ALGORITHM, GEOCODING

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
Навчально-науковий інститут інформаційних технологій
ПОДАННЯ
ГОЛОВІ ЕКЗАМЕНАЦІЙНОЇ КОМІСІЇ
ЩОДО ЗАХИСТУ КВАЛІФІКАЦІЙНОЇ РОБОТИ
на здобуття освітнього ступеня магістра

Направляється здобувач Федикович М.М. до захисту кваліфікаційної роботи
(прізвище та ініціали)
за спеціальністю 124 Системний аналіз
(код, найменування спеціальності)
освітньо-професійної програми Інтелектуальні системи управління
(назва)
на тему: «Платформа оптимізації маршутів для сервісів доставки «останньої милі»»

Кваліфікаційна робота і рецензія додаються.

Директора ННІТ _____ Катерина НЕСТЕРЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Висновок керівника кваліфікаційної роботи

За час роботи над кваліфікаційною роботою здобувач Федикович Марк Михайлович продемонстрував високий рівень теоретичної та дослідницької підготовки, успішно розглянувши сучасні методи оптимізації сервісів доставки «останньої милі» з аналізом результатів.

Він якісно провів аналіз сучасних методів оптимізації доставок, обґрунтував засоби реалізації та отримав практично значущі результати, виявивши при цьому ініціативність та хист до наукової діяльності. З огляду на глибину опрацювання матеріалу та практичну цінність роботи, Федикович Марк Михайлович заслуговує на оцінку «відмінно» та присвоєння кваліфікації магістр з системного аналізу.

Керівник кваліфікаційної роботи _____ Марк ФЕДИКОВИЧ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

«____» _____ 2025 року

Висновок кафедри про кваліфікаційну роботу

Кваліфікаційна робота розглянута. Здобувач Федикович М.М. допускається до захисту даної роботи в Екзаменаційній комісії.

Завідувач кафедри Інформаційних систем та технологій _____ Каміла СТОРЧАК
(назва) *(підпис)* *(Ім'я, ПРІЗВИЩЕ)*

«____» _____ 2025 року

ВІДГУК РЕЦЕНЗЕНТА
на кваліфікаційну роботу
на здобуття освітнього ступеня магістра

здобувача вищої освіти Федикович Марк Михайлович

(прізвище, ім'я, по батькові)

на тему «Платформа оптимізації маршрутів для сервісів доставки «останньої милі»»

Актуальність.

Актуальність дослідження зумовлена стрімким розвитком електронної комерції та зростаючих вимог клієнтів до швидкості та якості доставки, розробка ефективних інструментів планування та оптимізації маршрутів набуває особливого значення як для бізнесу, так і для зменшення негативного впливу на довкілля.

Позитивні сторони.

1. Робота ретельно розглядає сучасні підходи до оптимізації маршруті, та розроблено комплексну веб-платформу для автоматизованого планування логістичних операцій.
2. Досліджується існуючі методи та алгоритми оптимізації маршрутів, розроблено детальну специфікацію функціональних вимог до платформи оптимізації маршрутів.
3. У роботі використовується комплексний підхід до вирішення задачі, що охоплює як теоретичні так і практичні аспекти.

Недоліки.

1. Недостатньо детально розглянуто питання інтеграції з існуючими корпоративними системами та провести більш глибокий економічний аналіз ефективності впровадження платформи. вразливостей технології безпарольної аутентифікації WebAuthn до кібератак.
2. В роботі можна було включити порівняння продуктивності розробленого алгоритму з іншими сучасними підходами на стандартних бенчмаркових задачах.

Відзначені зауваження не впливають на загальну позитивну оцінку магістерської кваліфікаційної роботи.

Висновок: *кваліфікаційна робота на здобуття ступеня магістра заслуговує оцінку "відмінно", а здобувач Федикович Марк Михайловичемченко заслуговує присвоєння кваліфікації: магістр з системного аналізу.*

Рецензент _____ _____

науковий ступінь, (підпис) *(Ім'я ПРІЗВИЩЕ)*

вчене звання

ЗМІСТ

ВСТУП

Актуальність теми. Сучасний розвиток електронної комерції та цифрової економіки призвів до кардинальних змін у споживчій поведінці та очікуваннях клієнтів щодо якості та швидкості доставки товарів. За даними аналітичних агентств, глобальний ринок електронної комерції зростає щорічно на 15-20%, що спричиняє відповідне збільшення обсягів доставки останньої милі. Логістика останньої милі, що являє собою завершальний етап доставки товарів від розподільчого центру до кінцевого споживача, становить від 41% до 53% загальних логістичних витрат, що робить її найдорожчою частиною ланцюга постачання.

Оптимізація маршрутів доставки є критично важливим завданням для підвищення ефективності логістичних операцій, зниження витрат та покращення якості обслуговування клієнтів. Неефективне планування маршрутів призводить до збільшення відстаней переміщення, перевитрати палива, зростання викидів шкідливих речовин, невиправданого використання людських та матеріальних ресурсів, а також до зниження задоволеності клієнтів через несвоєчасність доставки. За оцінками експертів, впровадження ефективних систем оптимізації маршрутів може знизити операційні витрати на 10-30% при одночасному покращенні якості обслуговування. Незважаючи на наявність численних наукових досліджень в галузі оптимізації маршрутів та існування комерційних платформ управління доставкою, залишається значна потреба у спеціалізованих рішеннях, які поєднують ефективні алгоритми оптимізації, сучасні технології веб-розробки, зручні користувацькі інтерфейси та можливості інтеграції з різноманітними зовнішніми системами. Багато існуючих рішень є або надто дорогими для малого та середнього бізнесу, або не забезпечують достатньої гнучкості для адаптації до специфічних потреб конкретних організацій.

Аналіз останніх досліджень і публікацій. Задача оптимізації маршрутів доставки має глибокі теоретичні корені в теорії графів та комбінаторній оптимізації. Класична постановка проблеми відома як задача комівояжера (Travelling Salesman Problem), яка була формалізована ще в середині XX століття. Узагальненням цієї задачі є задача маршрутизації транспортних засобів (Vehicle Routing Problem), вперше сформульована Dantzig та Ramser у 1959 році. Подальший розвиток включав різноманітні варіанти задачі з урахуванням часових вікон, обмежень вантажопідйомності, множини депо та інших реальних обмежень.

Значний внесок у розвиток методів розв'язання задач маршрутизації внесли роботи Clarke та Wright (метод економії), Lin та Kernighan (метод k-opt), Christofides (евристика для задачі комівояжера), Dorigo (алгоритм мурашиної колонії), Holland (генетичні алгоритми) та багатьох інших дослідників. Сучасні дослідження фокусуються на розробці гібридних алгоритмів, що поєднують переваги різних підходів, використанні машинного навчання для покращення ефективності оптимізації, динамічній та реального часу маршрутизації в умовах невизначеності. В контексті практичних застосувань, важливі роботи присвячені аналізу специфіки логістики останньої милі, включаючи дослідження факторів, що впливають на вартість доставки, екологічних аспектів транспортування, впливу урбанізації на ефективність доставки. Останні публікації розглядають також інноваційні підходи до останньої милі, такі як використання дронів, автономних транспортних засобів, краудсорсингових моделей доставки та пунктів самообслуговування.

Мета і завдання дослідження. Метою роботи є розробка комплексної веб-платформи для автоматизованого планування та оптимізації маршрутів доставки останньої милі, що забезпечує значне зниження операційних витрат при збереженні або покращенні якості обслуговування клієнтів.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

– провести аналіз особливостей логістики останньої милі, виявити основні виклики та проблеми, що потребують вирішення;

– здійснити огляд та аналіз існуючих методів та алгоритмів оптимізації маршрутів, порівняти їх переваги та недоліки;

– проаналізувати сучасні платформи та системи управління доставкою, виявити їх функціональні можливості та обмеження;

– розробити детальну специфікацію функціональних вимог до платформи оптимізації маршрутів;

– спроєктувати масштабовану та надійну архітектуру системи на основі мікросервісного підходу;

– обрати та адаптувати математичні моделі задачі маршрутизації транспортних засобів з часовими вікнами до специфіки логістики останньої милі;

– розробити гібридний алгоритм оптимізації маршрутів, що забезпечує баланс між якістю рішення та швидкістю обчислень;

– спроєктувати ефективну структуру бази даних для зберігання інформації про замовлення, транспортні засоби, маршрути та користувачів;

– реалізувати веб-платформу з використанням сучасних технологій та фреймворків;

– розробити основні модулі системи: управління замовленнями, управління флотом, оптимізації маршрутів, відстеження в реальному часі та аналітики;

– провести комплексне тестування платформи на синтетичних та реальних даних;

– оцінити ефективність розробленої платформи за критеріями зниження витрат, підвищення продуктивності та покращення якості обслуговування.

Об'єкт дослідження - процес доставки товарів кінцевим споживачам у рамках логістики останньої милі.

Предмет дослідження - методи та алгоритми оптимізації маршрутів доставки з урахуванням множини обмежень та критеріїв ефективності.

Методи дослідження. Для розв'язання поставлених завдань використовувалися наступні методи: системний аналіз - для вивчення предметної області та формулювання вимог до платформи; методи комбінаторної оптимізації - для математичного моделювання задачі маршрутизації; евристичні та метаевристичні алгоритми - для ефективного розв'язання задач оптимізації великої розмірності; об'єктно-орієнтоване проєктування - для розробки архітектури програмної системи; методи проєктування баз даних - для створення ефективної структури зберігання інформації; методи тестування програмного забезпечення - для валідації коректності та ефективності реалізації; статистичний аналіз - для оцінки результатів тестування та порівняння з існуючими рішеннями.

Наукова новизна отриманих результатів полягає у: розробці адаптивного гібридного алгоритму оптимізації маршрутів, що поєднує переваги конструктивних евристик та локального пошуку з можливістю автоматичного налаштування параметрів на основі характеристик задачі; створенні мікросервісної архітектури платформи управління доставкою, що забезпечує високу масштабованість, надійність та можливість незалежного розвитку компонентів; розробці комплексного підходу до інтеграції алгоритмічних, програмних та інфраструктурних компонентів для створення ефективної системи управління доставкою.

Практична значущість отриманих результатів визначається створенням функціонуючої веб-платформи, яка може бути використана реальними сервісами доставки для оптимізації їх логістичних операцій. Впровадження розробленої системи дозволяє: знизити операційні витрати на доставку на 15-25% через оптимізацію маршрутів; підвищити продуктивність кур'єрів на 20-30% через ефективне планування завдань; покращити якість обслуговування клієнтів через дотримання часових вікон доставки до 95%; зменшити негативний вплив на довкілля через скорочення пробігу транспортних засобів та викидів шкідливих речовин.

Апробація результатів роботи. Основні положення та результати роботи доповідалися та обговорювалися на засіданнях кафедри, наукових семінарах університету. Матеріали дослідження можуть бути представлені на профільних конференціях з логістики, оптимізації та інформаційних технологій.

Структура та обсяг роботи. Кваліфікаційна робота складається зі вступу, трьох розділів, висновків, переліку посилань та додатків. Повний обсяг роботи становить 85 сторінок, включаючи 8 рисунків та 4 таблиці. Список використаних джерел містить 32 найменування.

РОЗДІЛ 1 АНАЛІЗ СУЧАСНИХ ПІДХОДІВ ДО ОПТИМІЗАЦІЇ МАРШРУТІВ ДОСТАВКИ

1.1 Особливості логістики «останньої милі» та її виклики

Логістика «останньої милі» являє собою завершальний етап процесу доставки товарів від розподільчого центру або складу безпосередньо до кінцевого споживача. Цей сегмент ланцюга постачання набув особливого значення в умовах стрімкого зростання електронної комерції та змін споживчої поведінки, спричинених глобальною цифровізацією економіки. За оцінками експертів, витрати на останню милю становлять від 41% до 53% загальних логістичних витрат, що робить цей етап найдорожчим у всьому ланцюзі доставки. Особливість останньої милі полягає в її унікальній складності, зумовленій необхідністю обслуговування великої кількості географічно розподілених точок доставки з різними вимогами щодо часових вікон, типу товару та специфічних побажань клієнтів. На відміну від магістральної логістики, де товари переміщуються великими партіями між обмеженою кількістю пунктів, остання миля характеризується високою фрагментацією замовлень та необхідністю забезпечення персоналізованого підходу до кожного клієнта [1, с. 198-204].

Сучасні тенденції ринку електронної комерції суттєво посилили вимоги до сервісів доставки останньої милі. Споживачі очікують швидкої доставки, часто протягом того самого дня або навіть за кілька годин, прозорості процесу відстеження замовлення в реальному часі, гнучкості у виборі часу та місця отримання товару, а також високої якості обслуговування без додаткових витрат. Ці очікування створюють значний тиск на логістичні компанії та змушують їх шукати інноваційні рішення для оптимізації процесів доставки. Одним із ключових викликів логістики останньої милі є проблема урбанізації та

зростання навантаження на транспортну інфраструктуру великих міст. Концентрація населення у мегаполісах призводить до збільшення обсягів доставок у густонаселених районах, де транспортні затори, обмеження паркування та складна дорожня мережа створюють додаткові перешкоди для ефективної доставки. Водночас, доставка у віддалені або малонаселені райони також є економічно невигідною через великі відстані між точками доставки та низьку щільність замовлень.

Динамічний характер попиту становить ще один значний виклик для організації останньої милі. Обсяги замовлень можуть суттєво варіюватися залежно від часу доби, дня тижня, сезонності та різноманітних зовнішніх факторів. Пікові навантаження під час святкових періодів або розпродажів вимагають гнучкості в розподілі ресурсів та здатності швидко масштабувати операції. Прогнозування попиту та планування необхідної кількості кур'єрів і транспортних засобів у таких умовах стає складним завданням, що потребує застосування передових аналітичних методів [2]. Проблема невдалих спроб доставки суттєво впливає на економічну ефективність останньої милі. За статистикою, у середньому 10-20% доставок виявляються невдалими через відсутність отримувача за адресою, неправильну адресу або відмову від прийняття замовлення. Кожна невдала спроба вимагає повторної доставки, що подвоює витрати та знижує продуктивність кур'єрів. Мінімізація таких ситуацій потребує ефективної комунікації з клієнтами, точної адресної інформації та гнучких опцій доставки.

Екологічні аспекти останньої милі набувають дедалі більшого значення в контексті глобальних зусиль щодо зменшення викидів парникових газів та боротьби зі змінами клімату. Традиційна модель доставки з використанням дизельних автомобілів призводить до значних викидів CO2, особливо в умовах міського трафіку з частими зупинками та низькою середньою швидкістю руху. Регуляторні вимоги щодо екологічності транспорту в багатьох містах

стимулюють перехід до альтернативних видів транспорту, таких як електромобілі, велосипеди або пішохідна доставка, що вимагає перегляду традиційних логістичних моделей.

Вартість робочої сили є одним із найбільших компонентів витрат на останню милю. Залучення, навчання та утримання кваліфікованих кур'єрів у умовах конкурентного ринку праці становить значну проблему для логістичних компаній. Висока плинність кадрів, сезонність попиту та необхідність забезпечення гнучкого графіку роботи ускладнюють планування людських ресурсів. Водночас, автоматизація та використання технологій можуть зменшити залежність від людського фактору, хоча повна заміна кур'єрів поки що залишається недосяжною метою. Технологічна інтеграція різних систем та платформ представляє ще один виклик для сучасної логістики останньої милі. Ефективна робота потребує координації між системами управління замовленнями, складського обліку, маршрутизації, відстеження доставок, комунікації з клієнтами та багатьма іншими компонентами. Забезпечення безшовної взаємодії цих систем, обміну даними в реальному часі та сумісності з різними технологічними платформами партнерів вимагає значних інвестицій у IT-інфраструктуру [3-4]. Безпека та захист даних набувають критичного значення в умовах цифровізації логістичних процесів. Системи управління доставкою обробляють великі обсяги персональної інформації клієнтів, включаючи адреси, контактні дані, платіжну інформацію та дані про місцезнаходження в реальному часі. Забезпечення конфіденційності цих даних, захист від кіберзагроз та дотримання вимог законодавства про захист персональних даних є обов'язковими вимогами для будь-якої сучасної платформи доставки.

Проблема масштабованості логістичних операцій особливо актуальна для компаній, що прагнуть розширити свою присутність на нових ринках або збільшити обсяги обслуговування. Модель, яка ефективно працює в одному

місті або регіоні, може виявитися неефективною в інших умовах через відмінності в інфраструктурі, культурних особливостях, регуляторному середовищі або конкурентному ландшафті. Створення гнучких та адаптивних логістичних систем, здатних ефективно працювати в різних умовах, є важливою задачею для глобальних гравців ринку.

Координація між різними учасниками екосистеми доставки додає складності управлінню останньою милею. У багатьох випадках логістичні операції включають взаємодію між інтернет-магазинами, сторонніми службами доставки, пунктами видачі, постаматами та іншими елементами інфраструктури. Забезпечення ефективної комунікації, чіткого розподілу відповідальності та синхронізації процесів між усіма сторонами вимагає складних організаційних та технологічних рішень [5-7]. Якість обслуговування клієнтів безпосередньо пов'язана з ефективністю останньої милі та має критичне значення для конкурентоспроможності бізнесу. Негативний досвід доставки може призвести до втрати клієнтів та погіршення репутації компанії, навіть якщо якість самого товару є високою. Забезпечення стабільно високого рівня сервісу при мінімізації витрат є балансуванням, що вимагає постійної оптимізації процесів та інвестицій у технології та навчання персоналу.

Регуляторне середовище та дотримання численних вимог законодавства створює додатковий рівень складності для організації останньої милі. Різні юрисдикції можуть встановлювати різні вимоги щодо часу доставки у певних зонах, екологічних стандартів транспорту, умов праці кур'єрів, обробки повернень товарів та багатьох інших аспектів. Постійний моніторинг змін у регуляторному середовищі та адаптація бізнес-процесів до нових вимог є необхідною умовою легальної та ефективної роботи [8]. Ці численні виклики демонструють складність логістики останньої милі та пояснюють високу частку витрат на цьому етапі ланцюга постачання. Вирішення цих проблем вимагає комплексного підходу, що поєднує інноваційні технології, ефективні

організаційні процеси, аналітику даних та орієнтацію на потреби клієнтів. Розробка спеціалізованих платформ оптимізації маршрутів є одним із ключових інструментів підвищення ефективності останньої милі, здатним суттєво знизити витрати при одночасному покращенні якості обслуговування [9-10].

1.2 Огляд існуючих методів та алгоритмів оптимізації маршрутів

Оптимізація маршрутів являє собою математичну задачу, що належить до класу NP-складних задач комбінаторної оптимізації. Класична постановка проблеми відома як задача комівояжера, де необхідно знайти найкоротший маршрут, що проходить через заданий набір точок і повертається до початкової позиції. Однак реальні задачі маршрутизації в логістиці останньої милі є значно складнішими, оскільки включають множину транспортних засобів, часові вікна доставки, різні типи товарів, обмеження вантажопідйомності та багато інших параметрів. Задача маршрутизації транспортних засобів є узагальненням класичної задачі комівояжера на випадок множини транспортних засобів. У базовій постановці необхідно сформувати набір маршрутів для флоту однорідних транспортних засобів, розташованих в одному депо, таким чином, щоб обслужити всіх клієнтів при мінімізації загальної вартості або відстані. Математично задача формулюється як оптимізаційна проблема на графі, де вершини відповідають клієнтам та депо, а ребра представляють можливі переміщення між ними з відповідними вагами [11].

Точні методи розв'язання задач маршрутизації гарантують знаходження оптимального рішення, але їх застосування обмежене розміром задачі через експоненційне зростання обчислювальної складності. Метод гілок та меж є одним із класичних точних підходів, який систематично досліджує простір рішень, відсікаючи неперспективні гілки на основі нижніх та верхніх оцінок цільової функції. Цей метод ефективний для задач малого та середнього розміру,

але стає непрактичним для реальних задач з сотнями клієнтів.

Методи динамічного програмування дозволяють розв'язувати певні класи задач маршрутизації шляхом розбиття на підзадачі та ефективного зберігання проміжних результатів. Алгоритм Хелда-Карпа для задачі комівояжера має складність експоненційну від кількості вершин, але з поліноміальним множником, що робить його більш ефективним порівняно з повним перебором. Однак вимоги до пам'яті обмежують застосування цього підходу для задач з кількістю точок більше кількох десятків.

Методи лінійного та цілочисельного програмування формулюють задачу маршрутизації як оптимізаційну модель з лінійною цільовою функцією та системою лінійних обмежень. Сучасні розв'язувачі, такі як CPLEX, Gurobi або CBC, використовують складні техніки, включаючи методи площин відсікання, декомпозицію та попередню обробку, для ефективного розв'язання великих задач цілочисельної оптимізації. Хоча ці інструменти не гарантують швидкого знаходження оптимального рішення для всіх випадків, вони часто виявляються ефективними для практичних задач середнього розміру. Евристичні методи жертвують гарантією оптимальності розв'язку заради швидкості обчислень та можливості обробляти великі задачі. Конструктивні евристики будують рішення крок за кроком, додаючи клієнтів до маршрутів на основі певних критеріїв. Метод найближчого сусіда послідовно додає до поточного маршруту найближчого необслуженого клієнта, що дає швидке, хоча часто не оптимальне рішення [12-13]. Метод економії Кларка-Райта оцінює вигоду від об'єднання окремих маршрутів до кожного клієнта в спільні маршрути та послідовно виконує найвигідніші об'єднання.

Поліпшувальні евристики починають з деякого початкового рішення та ітеративно покращують його шляхом локальних модифікацій. Метод 2-opt розглядає всі можливі варіанти видалення двох ребер з маршруту та їх заміни іншими двома ребрами, які не перетинаються, зберігаючи при цьому зв'язність.

Процес повторюється до тих пір, поки знаходяться поліпшення. Узагальнення цього підходу, методи k-opt, розглядають одночасне видалення k ребер, але їх обчислювальна складність швидко зростає з збільшенням k.

Метаевристичні підходи представляють загальні стратегії пошуку, які керують локальними евристиками для ефективного дослідження простору рішень та уникнення зациклення в локальних оптимумах. Імітація відпалу моделює процес кристалізації металу при його повільному охолодженні, дозволяючи з певною ймовірністю приймати погіршувальні кроки для виходу з локальних оптимумів. Ймовірність прийняття погіршувального рішення поступово зменшується згідно з розкладом температури, що забезпечує баланс між дослідженням та експлуатацією. Генетичні алгоритми черпають натхнення з еволюційних процесів в природі, підтримуючи популяцію рішень та застосовуючи оператори схрещування, мутації та селекції для створення нових поколінь. Різноманітність популяції дозволяє дослідити різні регіони простору рішень, а селекційний тиск спрямовує еволюцію до кращих рішень. Ефективність генетичних алгоритмів суттєво залежить від правильного вибору представлення рішення, операторів варіації та параметрів алгоритму [14-15]. Алгоритми мурашиної колонії моделюють поведінку мурашок, які знаходять короткі шляхи між гніздом та джерелами їжі за допомогою феромонових слідів. В контексті маршрутизації штучні мурашки ітеративно будують рішення, вибираючи наступні точки відвідування з ймовірністю, що залежить від концентрації феромону та евристичної інформації. Після кожної ітерації феромонові сліди оновлюються пропорційно якості знайдених рішень, що створює позитивний зворотний зв'язок до перспективних регіонів простору пошуку.

Методи пошуку з заборонами підтримують список нещодавно відвіданих рішень або модифікацій, які тимчасово заборонені для повторного розгляду. Це запобігає циклічному поверненню до вже досліджених регіонів та стимулює

дослідження нових частин простору рішень. Довгострокова та короткострокова пам'ять дозволяють ефективно балансувати між інтенсифікацією пошуку в перспективних регіонах та диверсифікацією для уникнення локальних оптимумів.

Гібридні підходи поєднують переваги різних методів для досягнення кращої ефективності. Наприклад, комбінація генетичного алгоритму з локальним пошуком використовує глобальну пошукову стратегію генетичного алгоритму для дослідження простору рішень та локальний пошук для інтенсивної оптимізації перспективних рішень. Інтеграція евристик з методами математичного програмування дозволяє генерувати якісні початкові рішення для точних методів або використовувати точні підходи для розв'язання критичних підзадач. Адаптивні та само-налаштовувані алгоритми автоматично коригують свої параметри під час роботи на основі характеристик задачі та динаміки процесу пошуку. Адаптивний вибір операторів у генетичних алгоритмах змінює ймовірність застосування різних операторів схрещування та мутації залежно від їх ефективності в останніх поколіннях [16-17]. Такий підхід зменшує необхідність ручного налаштування параметрів та підвищує робастність алгоритмів до різних типів задач.

Паралелізація та розподілені обчислення дозволяють суттєво прискорити розв'язання великих задач маршрутизації. Популяційні метаевристики, такі як генетичні алгоритми або алгоритми мурашиної колонії, природно піддаються паралелізації через незалежність індивідуумів або агентів. Декомпозиційні підходи розбивають велику задачу на менші підзадачі, які розв'язуються паралельно з періодичною координацією та обміном інформацією між процесами. Машинне навчання та методи штучного інтелекту знаходять дедалі ширше застосування в оптимізації маршрутів. Навчання з підкріпленням дозволяє агентам вчитися приймати оптимальні рішення про маршрутизацію через взаємодію з середовищем та отримання винагород. Глибокі нейронні

мережі можуть навчитися конструювати рішення безпосередньо з вхідних даних задачі, потенційно досягаючи швидкості інференції, недоступної традиційним методам оптимізації.

Методи прогнозування та аналізу даних відіграють важливу роль в ефективній маршрутизації. Прогнозування попиту дозволяє передбачити майбутні замовлення та проактивно планувати ресурси. Аналіз історичних даних про трафік допомагає точніше оцінювати час подорожі між точками доставки. Кластеризація клієнтів за географічним розташуванням, часовими вікнами або іншими характеристиками може спростити задачу маршрутизації, розбиваючи її на менші підзадачі.

Динамічна та реального часу оптимізація маршрутів враховує зміни умов під час виконання доставки. Нові замовлення можуть надходити після початку роботи, клієнти можуть скасовувати замовлення або змінювати вимоги, дорожні умови можуть змінюватися через затори або дорожні роботи. Алгоритми повинні швидко перераховувати маршрути з урахуванням нової інформації, балансуючи між якістю рішення та швидкістю обчислень [18]. Багатокритеріальна оптимізація розглядає одночасно декілька часто суперечливих цілей, таких як мінімізація загальної відстані, балансування навантаження між транспортними засобами, максимізація задоволеності клієнтів або мінімізація викидів CO2. Методи Парето-оптимізації шукають набір рішень, які не можуть бути покращені за одним критерієм без погіршення за іншим, дозволяючи особі, що приймає рішення, вибрати найбільш підходящий компроміс.

Стохастична оптимізація враховує невизначеність в параметрах задачі, таких як час подорожі, попит клієнтів або доступність транспортних засобів. Підходи, що базуються на сценаріях, розглядають множину можливих реалізацій невизначених параметрів та шукають рішення, що є робастними або оптимальними в середньому за всіма сценаріями. Імовірнісні методи явно

моделюють розподіли ймовірностей невизначених параметрів та оптимізують очікувані показники ефективності.

Вибір методу оптимізації для конкретного застосування залежить від багатьох факторів, включаючи розмір задачі, доступний час для обчислень, вимоги до якості рішення, наявність спеціалізованого обладнання та існуючої інфраструктури. Для задач малого розміру з жорсткими вимогами до оптимальності можуть бути прийнятними точні методи. Для великих задач з обмеженим часом на обчислення евристичні та метаевристичні підходи часто є єдиним практичним варіантом [19]. Гібридні методи, що поєднують переваги різних підходів, представляють перспективний напрямок для досягнення кращого балансу між якістю рішення та обчислювальною ефективністю.

1.3 Аналіз сучасних платформ та систем управління доставкою

Сучасний ринок платформ управління доставкою характеризується високою конкуренцією та різноманітністю рішень, від спеціалізованих інструментів для оптимізації маршрутів до комплексних систем управління всім ланцюгом постачання. Аналіз існуючих платформ дозволяє виявити ключові функціональні можливості, технологічні підходи та бізнес-моделі, що визначають їх конкурентні переваги та обмеження.

Комерційні платформи для оптимізації маршрутів, такі як Routific, OptimoRoute та Route4Me, надають хмарні рішення, орієнтовані на малі та середні підприємства. Ці платформи пропонують інтуїтивно зрозумілі веб-інтерфейси для введення замовлень, автоматичне планування маршрутів, мобільні додатки для кур'єрів, відстеження доставок в реальному часі та аналітичні інструменти. Більшість з них використовують власні алгоритми оптимізації, деталі яких не розкриваються, але базуються на комбінації евристичних та метаевристичних методів для забезпечення швидкої генерації

якісних рішень.

Routific позиціонується як рішення для компаній з регулярними доставками та пропонує функції автоматичного імпорту замовлень, оптимізації маршрутів з урахуванням часових вікон та пріоритетів, управління флотом та інтеграції з популярними платформами електронної комерції. Платформа використовує модель ціноутворення на основі кількості зупинок, що робить її доступною для невеликих бізнесів. Однак користувачі відзначають обмежені можливості кастомізації та відсутність деяких специфічних функцій для складних випадків, таких як доставка з декількох депо або різнорідний флот транспортних засобів.

OptimoRoute фокусується на малих та середніх сервісах доставки та обслуговування, пропонуючи автоматичне планування, оптимізацію маршрутів, мобільні додатки для водіїв з навігацією, збір підписів та фотофіксацією доставки, а також інструменти комунікації з клієнтами. Платформа підтримує планування на основі навичок водіїв, зон обслуговування, пріоритетів замовлень та інших обмежень. Інтеграція з популярними CRM та ERP системами спрощує впровадження в існуючі бізнес-процеси. Проте, деякі користувачі зазначають складність початкового налаштування та необхідність навчання для ефективного використання всіх можливостей [20, с. 306-319].

Route4Me є однією з найстаріших та найбільших платформ для планування маршрутів з понад мільйоном користувачів. Платформа пропонує широкий спектр функцій, включаючи оптимізацію маршрутів для множини транспортних засобів, геозони, інтеграцію з різними джерелами даних, аналітику та звітність. Велика база користувачів забезпечує активну спільноту та багато ресурсів для навчання. Однак архітектура платформи, розроблена кілька років тому, може здаватися застарілою порівняно з новішими рішеннями, а інтерфейс користувача інколи критикується за надмірну складність.

Корпоративні рішення, такі як Oracle Transportation Management, SAP

Transportation Management та Blue Yonder, надають комплексні можливості управління ланцюгом постачання, включаючи оптимізацію маршрутів як одну з багатьох функцій. Ці системи розраховані на великі підприємства з складними логістичними операціями та інтегруються з іншими корпоративними системами для забезпечення наскрізної видимості та контролю. Вони підтримують складні сценарії маршрутизації, такі як мультимодальні перевезення, консолідація вантажів, управління субпідрядниками та глобальні операції.

Oracle Transportation Management є частиною Oracle Supply Chain Management Cloud та пропонує функції планування, виконання та моніторингу перевезень. Система підтримує оптимізацію на різних рівнях, від стратегічного планування мережі до тактичного планування маршрутів та оперативного виконання. Інтеграція з іншими модулями Oracle забезпечує безшовний обмін даними та єдину платформу для всіх аспектів управління ланцюгом постачання. Однак складність системи та висока вартість впровадження роблять її доступною переважно для великих корпорацій [21-22].

SAP Transportation Management інтегрується з екосистемою SAP та пропонує всеосяжні можливості для управління вхідними та вихідними перевезеннями. Система підтримує планування маршрутів, вибір перевізників, аудит вантажів, відстеження відправлень та управління претензіями. Потужні аналітичні інструменти дозволяють контролювати ключові показники ефективності та виявляти можливості для покращення. Як і в випадку Oracle, висока вартість та складність впровадження є основними бар'єрами для малих та середніх підприємств.

Відкриті платформи та бібліотеки оптимізації надають розробникам інструменти для створення власних рішень, адаптованих до специфічних потреб. OptaPlanner є відкритим Java-фреймворком для вирішення задач комбінаторної оптимізації, включаючи маршрутизацію транспортних засобів. Фреймворк реалізує різноманітні метаевристичні алгоритми та дозволяє

розробникам декларативно визначати обмеження та цілі оптимізації. Гнучкість та можливість кастомізації роблять OptaPlanner привабливим для організацій з унікальними вимогами, хоча це вимагає залучення кваліфікованих розробників.

OR-Tools від Google є набором інструментів для оптимізації з відкритим кодом, що включає потужний розв'язувач для задач маршрутизації транспортних засобів. Бібліотека підтримує широкий спектр обмежень, таких як часові вікна, різні типи транспортних засобів, декілька депо, пріоритети та багато інших. Ефективні алгоритми та добре документований API роблять OR-Tools популярним вибором для дослідників та розробників [23]. Проте, використання бібліотеки потребує програмістських навичок та розуміння концепцій оптимізації.

Спеціалізовані рішення для певних секторів пропонують функції, оптимізовані для специфічних вимог галузі. Платформи для доставки їжі, такі як Onfleet та Tookan, включають функції управління замовленнями в реальному часі, оптимізацію маршрутів з урахуванням температурного режиму та терміну придатності, інтеграцію з POS-системами ресторанів та інструменти комунікації з клієнтами. Рішення для служб кур'єрської доставки акцентують увагу на швидкій обробці замовлень, динамічній маршрутизації та масштабованості для обробки пікових навантажень

Платформи на основі краудсорсингу, такі як Stuart або Lalamove, використовують модель гіг-економіки, залучаючи незалежних кур'єрів для виконання доставок. Ці платформи автоматизують процес підбору найближчого доступного кур'єра для кожного замовлення, оптимізують розподіл завдань в реальному часі та забезпечують прозорість через відстеження та рейтингові системи. Динамічне ціноутворення балансує попит та пропозицію, стимулюючи кур'єрів працювати в пікові періоди або складних районах. Технологічний стек сучасних платформ управління доставкою зазвичай включає хмарну інфраструктуру для забезпечення масштабованості та доступності, мобільні

додатки для кур'єрів та клієнтів, веб-портали для диспетчерів та менеджерів, API для інтеграції з зовнішніми системами, сервіси геолокації та картографії, бази даних для зберігання замовлень та історичних даних, а також аналітичні інструменти для звітності та візуалізації. Архітектура мікросервісів дозволяє незалежно розвивати та масштабувати різні компоненти системи [24].

Інтеграція з зовнішніми сервісами та API є критичною для ефективної роботи платформ доставки. Інтеграція з картографічними сервісами, такими як Google Maps, OpenStreetMap або HERE, забезпечує геокодування адрес, розрахунок відстаней та часу подорожі, візуалізацію маршрутів та навігацію. API платформ електронної комерції дозволяють автоматично імпортувати замовлення з інтернет-магазинів. Інтеграція з системами обліку та CRM забезпечує синхронізацію даних про клієнтів та фінансові операції. Аналітика та звітність є важливими компонентами платформ управління доставкою, що дозволяють контролювати ключові показники ефективності, такі як середній час доставки, відсоток вдалих доставок, витрати на доставку, завантаження транспортних засобів та продуктивність кур'єрів. Візуалізація даних у вигляді дашбордів, графіків та карт допомагає швидко виявляти проблеми та приймати обґрунтовані рішення. Прогнозна аналітика використовує історичні дані для передбачення майбутнього попиту та оптимізації планування ресурсів [25-26]

Безпека та конфіденційність даних є критичними аспектами платформ доставки, оскільки вони обробляють чутливу інформацію про клієнтів та бізнес-процеси. Шифрування даних при передачі та зберіганні, аутентифікація та авторизація користувачів, аудит дій користувачів, регулярні бекапи та плани відновлення після збоїв є стандартними практиками безпеки. Відповідність вимогам законодавства про захист персональних даних вимагає впровадження політик конфіденційності, інструментів управління згодою користувачів та можливостей видалення даних на запит.

Моделі ціноутворення платформ доставки варіюються від підписки з

фіксованою щомісячною платою до оплати за використання на основі кількості доставок, зупинок або транспортних засобів. Деякі платформи пропонують безкоштовні тарифи з обмеженою функціональністю для невеликих бізнесів, тоді як корпоративні рішення зазвичай мають індивідуальне ціноутворення на основі обсягів та специфічних вимог. Витрати на впровадження, навчання та технічну підтримку також є важливими факторами при виборі платформи.

Користувацький досвід та інтуїтивність інтерфейсу суттєво впливають на прийняття платформи користувачами. Складні системи з великою кількістю функцій можуть вимагати значного навчання та мати високий поріг входу, тоді як прості та інтуїтивні інтерфейси дозволяють швидко почати роботу, але можуть не підтримувати складні сценарії. Балансування між функціональністю та простотою використання є ключовим викликом при проєктуванні платформ. Мобільність та можливість роботи в офлайн-режимі важливі для кур'єрів, які працюють в умовах нестабільного інтернет-з'єднання. Мобільні додатки повинні бути оптимізовані для низького споживання батареї та даних, підтримувати офлайн-режим з синхронізацією даних при відновленні з'єднання, надавати зручну навігацію та інтерфейс для збору підтверджень доставки.

Порівняльний аналіз платформ показує, що не існує універсального рішення, що підходить для всіх випадків. Вибір платформи залежить від розміру бізнесу, специфіки операцій, бюджету, технічних можливостей команди та довгострокової стратегії компанії. Малі бізнеси можуть віддати перевагу простим хмарним рішенням з низькою вартістю входу, тоді як великі підприємства можуть інвестувати в корпоративні системи або розробляти власні рішення на основі відкритих бібліотек [27]. Гібридні підходи, що поєднують стандартні платформи з кастомними розробками, можуть забезпечити оптимальний баланс між вартістю, функціональністю та гнучкістю.

1.4 Постановка задачі дослідження

На основі проведеного аналізу особливостей логістики останньої милі, існуючих методів оптимізації маршрутів та сучасних платформ управління доставкою можна сформулювати основну проблематику дослідження. Незважаючи на наявність численних рішень на ринку, багато компаній, особливо малого та середнього бізнесу, стикаються з обмеженнями існуючих платформ щодо гнучкості налаштування під специфічні потреби, високої вартості корпоративних рішень або недостатньої функціональності бюджетних варіантів.

Ключові проблеми, які потребують вирішення, включають необхідність швидкої оптимізації маршрутів для великої кількості замовлень з різноманітними обмеженнями, динамічну адаптацію планів доставки до змін в реальному часі, інтеграцію з різноманітними зовнішніми системами та сервісами, забезпечення прозорості процесу доставки для всіх зацікавлених сторін, масштабованість системи для обробки зростаючих обсягів, оптимізацію використання ресурсів при збереженні високої якості обслуговування клієнтів.

Метою дослідження є розробка комплексної платформи оптимізації маршрутів для сервісів доставки останньої милі, що поєднує ефективні алгоритми маршрутизації, сучасні технології веб-розробки та зручний користувацький інтерфейс. Платформа повинна забезпечувати значне зниження операційних витрат доставки, покращення якості обслуговування клієнтів, підвищення продуктивності кур'єрів та зменшення негативного впливу на довкілля через оптимізацію маршрутів.

Для досягнення поставленої мети необхідно вирішити наступні задачі дослідження. По-перше, провести детальний аналіз вимог різних типів сервісів доставки останньої милі та виявити спільні та специфічні потреби. По-друге, розробити гнучку архітектуру системи, що дозволяє адаптувати платформу до різних сценаріїв використання без значних модифікацій коду. По-третє, вибрати

та адаптувати оптимізаційні алгоритми, що забезпечують баланс між якістю рішення та швидкістю обчислень для реальних задач.

Важливою задачею є проєктування та реалізація ефективної структури бази даних, що забезпечує швидкий доступ до даних, підтримує великі обсяги інформації та дозволяє легко розширювати функціональність. Необхідно також розробити набір API для інтеграції з популярними платформами електронної комерції, картографічними сервісами та іншими зовнішніми системами. Створення інтуїтивних користувацьких інтерфейсів для диспетчерів, менеджерів та кур'єрів є критичним для прийняття системи користувачами.

Алгоритмічний компонент дослідження фокусується на розробці ефективного гібридного підходу до оптимізації маршрутів, що поєднує переваги різних методів. Необхідно дослідити можливості інтеграції конструктивних евристик для швидкої генерації початкових рішень, метаевристичних алгоритмів для покращення якості рішень та елементів машинного навчання для адаптації до специфіки конкретних бізнесів та регіонів. Особливу увагу слід приділити динамічній оптимізації в реальному часі, що дозволяє швидко реагувати на нові замовлення, скасування або зміни дорожніх умов.

Технологічний стек платформи повинен забезпечувати високу продуктивність, масштабованість та зручність розробки. Вибір мов програмування, фреймворків, систем управління базами даних, хмарної інфраструктури та інших компонентів повинен базуватися на аналізі вимог, досвіді використання в подібних проектах та перспективах підтримки та розвитку технологій. Архітектура мікросервісів може забезпечити модульність та можливість незалежного масштабування різних компонентів системи

Функціональні можливості платформи повинні охоплювати весь життєвий цикл процесу доставки від отримання замовлення до завершення доставки та аналізу результатів. Модуль імпорту та управління замовленнями повинен підтримувати різні джерела даних, автоматичне геокодування адрес, валідацію

та попередню обробку інформації. Модуль оптимізації маршрутів повинен враховувати множину обмежень, таких як часові вікна, пріоритети, вантажопідйомність транспортних засобів, зони обслуговування та інші параметри

Модуль управління флотом повинен вести облік транспортних засобів з їх характеристиками, графіками роботи водіїв, обмеженнями та поточним станом. Модуль відстеження в реальному часі повинен збирати дані про місцезнаходження кур'єрів, статус виконання замовлень та передавати цю інформацію диспетчерам та клієнтам. Модуль комунікації повинен забезпечувати ефективний обмін інформацією між усіма учасниками процесу через різні канали, такі як SMS, електронна пошта, push-повідомлення та месенджери.

Аналітичний модуль повинен збирати та обробляти дані про ефективність доставок, генерувати звіти та візуалізації ключових показників ефективності. Інструменти прогнозування повинні використовувати історичні дані для передбачення майбутнього попиту, що дозволяє проактивно планувати ресурси. Можливості машинного навчання можуть покращувати точність прогнозів та адаптувати алгоритми оптимізації до специфіки конкретного бізнесу.

Безпека та конфіденційність даних повинні бути закладені в архітектуру платформи на всіх рівнях. Необхідно реалізувати надійну систему аутентифікації та авторизації, шифрування чутливих даних, захист від поширених вразливостей веб-додатків, регулярне резервне копіювання та можливість відновлення після збоїв. Відповідність вимогам законодавства про захист персональних даних вимагає впровадження відповідних політик та технічних засобів контролю.

Тестування та валідація платформи повинні включати різні типи тестів для забезпечення якості та надійності системи. Модульні тести перевіряють коректність окремих компонентів, інтеграційні тести - взаємодію між модулями,

функціональні тести - відповідність реалізованої функціональності вимогам, а навантажувальні тести - продуктивність системи під високим навантаженням. Тестування на реальних даних з існуючих сервісів доставки дозволяє оцінити практичну ефективність платформи.

Критеріями оцінки ефективності розробленої платформи будуть швидкість генерації оптимізованих маршрутів для задач різного розміру, якість отриманих рішень порівняно з існуючими методами та еталонними задачами, зниження загальної відстані або часу доставки порівняно з неоптимізованими маршрутами, підвищення продуктивності кур'єрів через збільшення кількості виконаних доставок за зміну, покращення показників задоволеності клієнтів через дотримання часових вікон та зменшення часу очікування.

Додатковими критеріями є масштабованість системи та здатність обробляти зростаючі обсяги замовлень без деградації продуктивності, гнучкість та можливість адаптації до різних типів сервісів доставки та специфічних вимог, зручність використання інтерфейсів для різних категорій користувачів, ефективність інтеграції з зовнішніми системами та сервісами. Економічна ефективність платформи буде оцінюватися через розрахунок потенційного зниження операційних витрат та окупності інвестицій у розробку та впровадження системи.

Наукова новизна дослідження полягає у розробці гібридного підходу до оптимізації маршрутів доставки, що адаптивно поєднує різні алгоритмічні методи залежно від характеристик задачі та доступного часу на обчислення. Використання машинного навчання для налаштування параметрів оптимізації на основі історичних даних конкретного бізнесу та регіону дозволяє досягти кращих результатів порівняно з універсальними налаштуваннями. Розроблена архітектура платформи забезпечує модульність, масштабованість та можливість розширення функціональності без суттєвих змін базової структури.

Практична значущість роботи визначається створенням функціонуючої

платформи, що може бути використана реальними сервісами доставки для оптимізації їх операцій. Зниження витрат на доставку при збереженні або покращенні якості обслуговування має прямий економічний ефект для бізнесу. Покращення екологічних показників через зменшення загальної відстані подорожі та викидів вносить внесок у сталий розвиток та відповідає сучасним тенденціям відповідального бізнесу. Розроблені методики та інструменти можуть бути адаптовані для інших задач маршрутизації та логістики, таких як планування маршрутів громадського транспорту, збір відходів, сервісне обслуговування обладнання або доставка медичних препаратів. Відкрита архітектура та документовані API дозволяють дослідникам та розробникам будувати власні рішення на основі платформи або інтегрувати її компоненти у свої системи.

Структура подальшого викладу матеріалу роботи організована таким чином, щоб послідовно розкрити всі аспекти дослідження. Другий розділ присвячений проєктуванню архітектури платформи, де детально описуються функціональні вимоги, архітектурні рішення, математичні моделі задачі маршрутизації, вибрані алгоритми оптимізації та структура бази даних. Третій розділ фокусується на практичній реалізації платформи, включаючи вибір технологій, імплементацію основних модулів, інтеграцію алгоритмів оптимізації та результати тестування системи на реальних та синтетичних даних.

У висновках узагальнюються досягнуті результати, оцінюється ступінь виконання поставлених задач, формулюються основні висновки дослідження та окреслюються перспективи подальшого розвитку платформи. Список використаних джерел включає наукові публікації з оптимізації маршрутів, описи існуючих платформ та систем, технічну документацію використаних технологій та інші релевантні матеріали.

РОЗДІЛ 2 РОЗРОБКА АРХІТЕКТУРИ ПЛАТФОРМИ ОПТИМІЗАЦІЇ МАРШРУТІВ

2.1 Вимоги до функціональних можливостей платформи

Розробка ефективної платформи оптимізації маршрутів для сервісів доставки останньої милі починається з детального аналізу функціональних вимог, що відображають потреби різних категорій користувачів та специфіку логістичних операцій. Функціональні вимоги визначають, що саме повинна робити система, які завдання вирішувати та яким чином взаємодіяти з користувачами та зовнішніми системами.

Центральною функціональною можливістю платформи є автоматизоване планування та оптимізація маршрутів доставки з урахуванням множини обмежень та критеріїв ефективності. Система повинна приймати на вхід інформацію про замовлення, включаючи адреси доставки, часові вікна, пріоритети, характеристики товарів та специфічні вимоги клієнтів. На основі цих даних платформа має генерувати оптимальні маршрути для доступного флоту транспортних засобів, мінімізуючи загальну відстань, час доставки або вартість при дотриманні всіх обмежень. Модуль управління замовленнями забезпечує повний життєвий цикл обробки замовлень від моменту їх створення до завершення доставки. Платформа повинна підтримувати різні способи введення замовлень, включаючи ручне введення через веб-інтерфейс, автоматичний імпорт з файлів різних форматів, інтеграцію через API з платформами електронної комерції та іншими зовнішніми системами. Кожне замовлення має містити повну інформацію про отримувача, адресу доставки, контактні дані, тип та характеристики товару, часові обмеження та додаткові інструкції.

Геокодування адрес є критичною функцією, що перетворює текстові адреси у географічні координати, необхідні для розрахунку відстаней та побудови маршрутів. Платформа повинна інтегруватися з надійними сервісами геокодування, такими як Google Maps Geocoding API, OpenStreetMap Nominatim або HERE Geocoding API. Система має обробляти неточні або неповні адреси, пропонуючи варіанти виправлень та валідуючи введену інформацію для зменшення помилок доставки.

Управління флотом транспортних засобів включає ведення реєстру всіх доступних транспортних засобів з їх детальними характеристиками. Для кожного транспортного засобу система повинна зберігати інформацію про вантажопідйомність, об'єм вантажного відділення, тип транспорту, споживання палива, екологічний клас, обмеження доступу до певних зон міста та поточний технічний стан. Платформа має враховувати графіки роботи водіїв, включаючи робочі зміни, перерви, вихідні дні та відпустки при плануванні маршрутів [28]. Алгоритм оптимізації маршрутів становить ядро платформи та повинен ефективно вирішувати задачу маршрутизації транспортних засобів з часовими вікнами. Система має підтримувати різні цільові функції оптимізації, такі як мінімізація загальної відстані, мінімізація часу доставки, мінімізація кількості використовуваних транспортних засобів, балансування навантаження між водіями або мінімізація викидів вуглекислого газу. Платформа повинна дозволяти комбінувати кілька критеріїв з різними вагами для досягнення компромісних рішень.

Динамічна оптимізація в реальному часі дозволяє адаптувати плани доставки до змін умов під час виконання маршрутів. Система повинна швидко перераховувати маршрути при надходженні нових термінових замовлень, скасуванні існуючих замовлень, зміні дорожньої ситуації або виникненні непередбачених обставин, таких як поломка транспортного засобу або хвороба водія. Алгоритм динамічної оптимізації має балансувати між якістю нового

рішення та швидкістю його генерування, забезпечуючи відповідь протягом кількох секунд.

Модуль відстеження доставок в реальному часі забезпечує збір та відображення поточної інформації про виконання маршрутів. Платформа має інтегруватися з GPS-трекерами або мобільними пристроями водіїв для отримання даних про їх місцезнаходження з інтервалом від кількох секунд до хвилини. Система повинна відображати позиції всіх транспортних засобів на карті, показувати прогрес виконання маршрутів, розраховувати очікуваний час прибуття до кожної точки доставки та автоматично виявляти відхилення від плану.

Комунікаційний модуль забезпечує ефективний обмін інформацією між усіма учасниками процесу доставки. Платформа повинна автоматично надсилати повідомлення клієнтам про підтвердження замовлення, планований час доставки, затримки та прибуття кур'єра через різні канали комунікації, включаючи SMS, електронну пошту, push-повідомлення та месенджери. Водії повинні мати можливість отримувати інструкції щодо маршруту, деталі замовлень та контактну інформацію клієнтів через мобільний додаток. Мобільний додаток для водіїв є критичним компонентом платформи, що забезпечує виконання доставок у полі. Додаток має відображати призначений маршрут з покроковою навігацією, список замовлень у порядку виконання з детальною інформацією про кожне замовлення, контактні дані клієнтів для комунікації у разі проблем. Водій повинен мати можливість відмічати статус виконання кожної доставки, збирати електронні підписи отримувачів, фотографувати підтвердження доставки та вводити причини невдалих спроб доставки.

Диспетчерський інтерфейс надає операторам інструменти для моніторингу та управління всіма аспектами процесу доставки. Через веб-інтерфейс диспетчери повинні бачити поточний стан всіх замовлень та

транспортних засобів, отримувати сповіщення про проблеми та відхилення від плану, мати можливість вручну коригувати маршрути або перерозподіляти замовлення між водіями. Система має надавати зручні інструменти пошуку та фільтрації для швидкого знаходження потрібної інформації серед великої кількості замовлень. Аналітичний модуль збирає та обробляє дані про ефективність доставок для підтримки прийняття управлінських рішень. Платформа повинна розраховувати ключові показники ефективності, такі як середній час доставки, відсоток вдалих доставок з першої спроби, середня кількість доставок на водія за зміну, середня вартість однієї доставки, завантаження транспортних засобів та дотримання часових вікон. Система має генерувати різноманітні звіти та візуалізації у вигляді графіків, діаграм та теплових карт для аналізу тенденцій та виявлення проблемних областей.

Прогнозна аналітика використовує історичні дані для передбачення майбутнього попиту та оптимізації планування ресурсів. Платформа повинна аналізувати сезонні коливання, тренди зростання, вплив зовнішніх факторів, таких як погода або святкові періоди, та генерувати прогнози обсягів замовлень на різні часові горизонти. Ці прогнози допомагають компаніям заздалегідь планувати необхідну кількість водіїв та транспортних засобів, оптимізувати графіки роботи та уникати як недостатності, так і надлишку ресурсів [29]. Модуль інтеграцій забезпечує взаємодію платформи з широким спектром зовнішніх систем та сервісів. Система повинна надавати добре документований RESTful API для інтеграції з платформами електронної комерції, такими як Shopify, WooCommerce, Magento або власними інтернет-магазинами клієнтів. Інтеграція з ERP-системами дозволяє синхронізувати дані про замовлення, клієнтів та інвентар. Підключення до систем управління складом забезпечує координацію між процесами комплектації та доставки.

Управління користувачами та ролями забезпечує безпечний доступ до функцій платформи відповідно до обов'язків кожного користувача. Система

повинна підтримувати різні ролі, такі як адміністратор, диспетчер, менеджер, водій та клієнт, кожна з яких має специфічний набір дозволів. Платформа має реалізувати надійну систему аутентифікації з підтримкою багатофакторної автентифікації для підвищення безпеки. Журналювання всіх дій користувачів дозволяє відстежувати зміни та розслідувати інциденти.

Таблиця 2.1

Основні функціональні вимоги платформи

Категорія	Функціональна вимога	Пріоритет
Управління замовленнями	Автоматичний імпорт замовлень з різних джерел	Високий
Управління замовленнями	Валідація та геокодування адрес	Високий
Оптимізація маршрутів	Побудова оптимальних маршрутів з урахуванням обмежень	Критичний
Оптимізація маршрутів	Динамічна оптимізація в реальному часі	Високий
Відстеження	Моніторинг позицій транспортних засобів в реальному часі	Високий
Відстеження	Розрахунок прогнозного часу прибуття	Середній
Комунікація	Автоматичні повідомлення клієнтам через різні канали	Високий
Аналітика	Розрахунок KPI та генерація звітів	Середній
Безпека	Аутентифікація та авторизація користувачів	Критичний

Система повідомлень та сповіщень інформує користувачів про важливі події та зміни стану. Платформа має підтримувати налаштування різних типів

сповіщень, включаючи нові замовлення, зміни в маршрутах, затримки доставки, проблеми з виконанням, досягнення порогових значень показників ефективності. Користувачі повинні мати можливість налаштовувати канали отримання сповіщень та їх пріоритети відповідно до своїх переваг.

Управління зонами обслуговування дозволяє визначати географічні області, де компанія надає послуги доставки. Платформа повинна підтримувати створення полігонів на карті, що представляють зони обслуговування, призначення певних транспортних засобів або команд до конкретних зон, встановлення різних тарифів або умов доставки для різних зон. Система має автоматично перевіряти, чи належить адреса замовлення до зони обслуговування, та відхиляти або позначати замовлення поза зоною. Функціонал обробки повернень та рекламацій є важливою частиною повного циклу управління доставкою. Платформа повинна підтримувати створення заявок на повернення товарів, планування зворотного забору, відстеження статусу повернень та інтеграцію з системами обліку для обробки фінансових аспектів повернень. Система має зберігати причини повернень для подальшого аналізу та покращення якості обслуговування.

Підтримка різних типів доставки розширює можливості платформи для різноманітних бізнес-моделей. Система повинна обробляти стандартну доставку до дверей клієнта, доставку до пунктів видачі або постаматів, доставку з можливістю примірки або огляду товару, часткову доставку великих замовлень кількома транспортними засобами, групову доставку кількох замовлень в одну адресу. Кожен тип доставки може мати специфічні правила та обмеження, які система має враховувати при оптимізації. Калькулятор вартості доставки розраховує ціну доставки для клієнта на основі різних факторів. Платформа повинна підтримувати гнучкі моделі ціноутворення, включаючи фіксовану вартість за доставку, вартість на основі відстані або ваги, диференційовані тарифи для різних зон або типів доставки, динамічне ціноутворення на основі

попиту та завантаження. Система має автоматично розраховувати вартість при оформленні замовлення та інтегруватися з платіжними системами для обробки оплати [30, с. 68-86].

Управління часовими вікнами доставки дозволяє клієнтам вибирати зручний час отримання замовлення. Платформа повинна підтримувати різні типи часових вікон, такі як фіксовані слоти протягом дня, доставка в межах певного діапазону годин, доставка до кінця робочого дня, термінова доставка протягом кількох годин. Система має враховувати обрані часові вікна при оптимізації маршрутів та автоматично визначати можливість дотримання обіцяного часу доставки.

Функціонал пріоритизації замовлень дозволяє впливати на порядок виконання доставок. Платформа повинна підтримувати різні рівні пріоритету, такі як стандартний, високий, терміновий або VIP, та враховувати їх при формуванні маршрутів. Система має дозволяти автоматичне присвоєння пріоритетів на основі характеристик замовлення, таких як тип товару, вартість замовлення або статус клієнта, а також ручну зміну пріоритетів диспетчерами у випадку необхідності. Підтримка мультитенантності дозволяє використовувати одну інстанцію платформи кількома незалежними організаціями. Система повинна забезпечувати повну ізоляцію даних між різними клієнтами, підтримку індивідуальних налаштувань та брендингу для кожного клієнта, гнучке управління доступом та дозволами в межах кожної організації. Така архітектура знижує вартість підтримки та дозволяє швидко підключати нових клієнтів до платформи.

Експорт та імпорт даних забезпечує гнучкість обміну інформацією з іншими системами. Платформа повинна підтримувати експорт звітів та даних в популярних форматах, таких як CSV, Excel, PDF та JSON, імпорт замовлень та довідкової інформації з файлів різних форматів, шаблони для полегшення

підготовки даних для імпорту. Система має валідувати імпортовані дані та надавати зрозумілі повідомлення про помилки для їх виправлення.

Резервне копіювання та відновлення даних є критичними функціями для забезпечення надійності та безперервності бізнесу. Платформа повинна автоматично створювати регулярні резервні копії всіх даних, включаючи замовлення, маршрути, користувачів та налаштування, зберігати резервні копії в географічно розподілених сховищах для захисту від локальних катастроф, надавати інструменти для швидкого відновлення системи після збоїв. Система має регулярно тестувати процедури відновлення для забезпечення їх ефективності. Багатомовність та локалізація дозволяють використовувати платформу в різних країнах та регіонах. Система повинна підтримувати інтерфейс користувача кількома мовами з можливістю легкого додавання нових локалізацій, формати дат, часу, чисел та валют відповідно до регіональних стандартів, різні системи адрес та поштових кодів для різних країн. Платформа має автоматично визначати мову користувача на основі налаштувань браузера або дозволяти явний вибір мови.

Масштабованість системи забезпечує здатність обробляти зростаючі обсяги даних та користувачів без деградації продуктивності. Платформа повинна підтримувати горизонтальне масштабування шляхом додавання нових серверів для обробки навантаження, використання систем балансування навантаження для розподілу запитів між серверами, кешування часто використовуваних даних для зменшення навантаження на базу даних, асинхронну обробку тривалих задач, таких як оптимізація маршрутів, для уникнення блокування користувацького інтерфейсу. Моніторинг та логування забезпечують видимість роботи системи та допомагають швидко виявляти та діагностувати проблеми. Платформа повинна збирати метрики продуктивності, такі як час відповіді API, використання ресурсів серверів, кількість активних користувачів, записувати детальні логи всіх операцій та помилок для

подальшого аналізу, надавати дашборди для візуалізації стану системи в реальному часі, автоматично сповіщувати адміністраторів про критичні проблеми або перевищення порогових значень метрик.

2.2 Проєктування архітектури системи

Архітектура платформи оптимізації маршрутів визначає структуру системи, взаємозв'язки між компонентами та принципи організації програмного коду. Правильне архітектурне проєктування є критичним для забезпечення масштабованості, надійності, підтримуваності та можливості розвитку платформи в довгостроковій перспективі.

Мікросервісна архітектура була обрана як основний підхід до побудови платформи через її численні переваги для розподілених систем. Замість монолітного додатку, де вся функціональність реалізована в єдиній кодовій базі, система розділена на набір незалежних сервісів, кожен з яких відповідає за конкретну бізнес-функцію [31]. Такий підхід дозволяє незалежно розробляти, тестувати, розгортати та масштабувати різні компоненти системи, використовувати найбільш підходящі технології для кожного сервісу, забезпечити ізоляцію збоїв та полегшити підтримку.

Сервіс управління замовленнями є відповідальним за весь життєвий цикл замовлень від створення до завершення. Він надає API для створення, читання, оновлення та видалення замовлень, забезпечує валідацію даних замовлень, зберігає історію змін статусів замовлень, інтегрується з зовнішніми системами для імпорту замовлень. Цей сервіс використовує реляційну базу даних для зберігання структурованих даних про замовлення з підтримкою транзакцій для забезпечення консистентності [32, с. 52-64]. Сервіс геокодування відповідає за перетворення адрес у географічні координати та зворотно. Він інтегрується з одним або кількома постачальниками геокодування для забезпечення надійності,

кешує результати геокодування для зменшення кількості зовнішніх запитів та вартості, обробляє випадки неоднозначних або неповних адрес, надаючи варіанти виправлень, валідує адреси на відповідність зонам обслуговування. Використання окремого сервісу дозволяє легко змінювати постачальників геокодування або використовувати кілька одночасно.

Сервіс управління флотом зберігає інформацію про транспортні засоби та водіїв. Він ведає реєстром транспортних засобів з їх характеристиками та поточним станом, управляє графіками роботи водіїв та їх доступністю, відстежує технічний стан транспортних засобів та строки обслуговування, забезпечує призначення водіїв до транспортних засобів. База даних цього сервісу містить всю необхідну інформацію для врахування обмежень флоту при оптимізації маршрутів.

Сервіс оптимізації маршрутів є ядром платформи та відповідає за генерування оптимальних планів доставки. Він отримує дані про замовлення, флот та обмеження з інших сервісів, застосовує алгоритми оптимізації для побудови маршрутів, забезпечує як пакетну оптимізацію для планування наступного дня, так і динамічну оптимізацію в реальному часі, підтримує різні цільові функції та обмеження. Цей сервіс є обчислювально інтенсивним та може масштабуватися горизонтально для обробки великих задач.

Сервіс відстеження збирає та обробляє дані про місцезнаходження транспортних засобів в реальному часі. Він отримує дані GPS з мобільних додатків водіїв або трекерів, зберігає траєкторії руху для подальшого аналізу, розраховує очікуваний час прибуття до точок доставки на основі поточної позиції та дорожньої ситуації, виявляє відхилення від запланованого маршруту або затримки, надає API для отримання поточних позицій та статусів транспортних засобів. Використання NoSQL бази даних дозволяє ефективно зберігати великі обсяги геопросторових даних.

Сервіс повідомлень забезпечує комунікацію між системою та користувачами. Він інтегрується з провайдерами SMS, email та push-повідомлень, підтримує шаблони повідомлень з підстановкою динамічних даних, управляє чергою повідомлень для надійної доставки, відстежує статус доставки повідомлень та обробляє помилки, надає можливість користувачам налаштовувати переваги отримання повідомлень. Асинхронна архітектура з використанням черг повідомлень забезпечує надійність навіть при збоях окремих компонентів.

Сервіс аналітики агрегує дані з різних сервісів для розрахунку показників ефективності та генерації звітів. Він періодично збирає дані про виконані доставки, маршрути, витрати, розраховує KPI та тренди, генерує різноманітні звіти у вигляді таблиць, графіків та діаграм, надає API для отримання аналітичних даних веб-інтерфейсом, підтримує експорт звітів в різних форматах. Використання окремого сховища даних для аналітики відокремлює аналітичне навантаження від операційних сервісів [33].

API Gateway є єдиною точкою входу для всіх зовнішніх запитів до системи. Він маршрутизує запити до відповідних мікросервісів на основі URL та методу, забезпечує аутентифікацію та авторизацію запитів, обмежує частоту запитів для захисту від зловживань, агрегує відповіді від кількох сервісів, якщо необхідно, забезпечує балансування навантаження між інстанціями сервісів, логує всі запити для моніторингу та аудиту. Використання API Gateway спрощує клієнтську логіку та забезпечує централізоване управління доступом.

Шина подій забезпечує асинхронну комунікацію між сервісами через публікацію та підписку на події. Коли відбувається важлива подія, такий як створення нового замовлення або зміна статусу доставки, сервіс публікує повідомлення про цю подію. Інші сервіси, які зацікавлені в цій події, підписуються на відповідний канал та отримують повідомлення. Така архітектура забезпечує слабке зв'язування між сервісами, дозволяє легко

додавати нові функції без зміни існуючих сервісів, забезпечує надійність через гарантовану доставку повідомлень.

Веб-додаток для диспетчерів реалізований як Single Page Application з використанням сучасного JavaScript фреймворку. Він комунікує з backend через API Gateway, відображає дані в реальному часі з використанням WebSocket з'єднань, надає інтерактивні карти для візуалізації маршрутів та позицій транспортних засобів, реалізує складні користувацькі інтерфейси для управління замовленнями та маршрутами. Розділення frontend та backend дозволяє незалежно розвивати їх та використовувати різні технологічні стеки.

Таблиця 2.2

Мікросервіси платформи та їх призначення

Назва сервісу	Основне призначення	Технології
Сервіс замовлень	Управління життєвим циклом замовлень	Node.js, PostgreSQL
Сервіс геокодування	Перетворення адрес у координати	Python, Redis
Сервіс флоту	Управління транспортними засобами та водіями	Java, PostgreSQL
Сервіс оптимізації	Генерування оптимальних маршрутів	Python, OR-Tools
Сервіс відстеження	Збір та обробка GPS даних	Node.js, MongoDB
Сервіс повідомлень	Надсилання повідомлень користувачам	Python, RabbitMQ
Сервіс аналітики	Розрахунок метрик та звітів	Python, ClickHouse
API Gateway	Маршрутизація запитів до сервісів	Kong, Nginx

Мобільний додаток для водіїв розроблений для iOS та Android платформ з використанням кросплатформенного фреймворку або нативної розробки. Він синхронізує дані про маршрути та замовлення з сервером, працює в офлайн-

режимі з локальним кешуванням даних, періодично надсилає дані GPS для відстеження, надає навігацію до точок доставки з інтеграцією з картографічними сервісами, дозволяє збирати підтвердження доставки, включаючи підписи та фотографії.

Система управління базами даних включає кілька типів баз даних для різних потреб. Реляційна база даних використовується для зберігання структурованих даних замовлень, користувачів, флоту з підтримкою транзакцій та реляційної цілісності. NoSQL база даних використовується для геопросторових даних відстеження, логів та інших неструктурованих даних з вимогами до високої швидкості запису. In-memory база даних використовується для кешування часто використовуваних даних, таких як конфігурація, довідники, результати геокодування. Така полі глотична персистентність дозволяє використовувати найбільш підходящі технології для кожного типу даних.

Система моніторингу та логування збирає метрики та логи з усіх компонентів системи. Централізований збір логів дозволяє легко шукати та аналізувати логи з різних сервісів в одному місці. Метрики продуктивності, такі як час відповіді API, використання ресурсів, кількість помилок, візуалізуються на дашбордах для моніторингу стану системи. Система сповіщень автоматично повідомляє адміністраторів про критичні проблеми або аномалії [34]. Інфраструктура розгортання базується на контейнерах та оркестрації контейнерів. Кожен мікросервіс пакується в Docker контейнер з усіма необхідними залежностями, що забезпечує консистентність середовища між розробкою та продакшеном. Kubernetes використовується для оркестрації контейнерів, забезпечуючи автоматичне масштабування на основі навантаження, самовідновлення при збоях контейнерів, балансування навантаження між інстанціями, керування конфігурацією та секретами.

Стратегія безпеки реалізована на кількох рівнях архітектури. Шифрування даних при передачі забезпечується використанням HTTPS для всіх з'єднань. Шифрування даних в спокої застосовується для чутливих даних в базах даних. Аутентифікація користувачів реалізована з використанням стандартів OAuth2 та OpenID Connect. Авторизація на основі ролей контролює доступ до функцій відповідно до прав користувача. Регулярні оновлення безпеки та патчі застосовуються до всіх компонентів системи. Стратегія резервного копіювання включає автоматичні щоденні резервні копії всіх баз даних, зберігання резервних копій в географічно розподілених сховищах для захисту від регіональних катастроф, регулярне тестування процедур відновлення, збереження кількох версій резервних копій для можливості відновлення до різних точок в часі. План відновлення після катастроф визначає процедури та цільові показники часу відновлення [35].

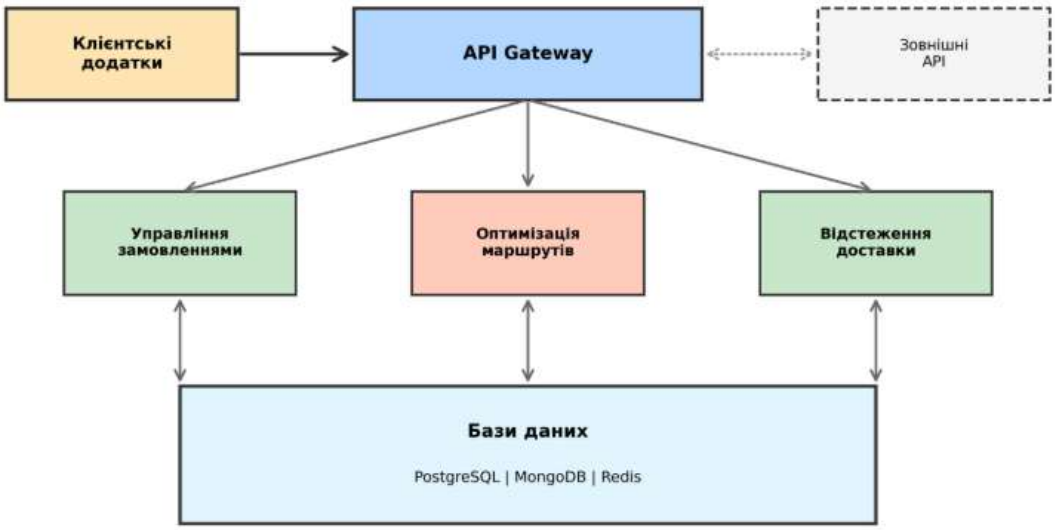


Рис. 2.1 - Архітектура платформи оптимізації маршрутів

Середовища розробки, тестування та продакшену розділені для забезпечення якості та стабільності. Середовище розробки використовується розробниками для створення та первинного тестування нових функцій.

Середовище тестування використовується для інтеграційного тестування, тестування продуктивності та приймального тестування. Продакшен середовище обслуговує реальних користувачів з максимальною надійністю та продуктивністю. Автоматизована система continuous integration та continuous deployment забезпечує швидке та надійне розгортання змін через ці середовища.

2.3 Вибір математичних моделей та алгоритмів оптимізації

Вибір математичних моделей та алгоритмів оптимізації є центральним рішенням при розробці платформи маршрутизації, оскільки саме ці компоненти визначають якість генерованих маршрутів та швидкість їх обчислення. Задача оптимізації маршрутів для сервісів доставки останньої милі може бути формалізована як задача маршрутизації транспортних засобів з часовими вікнами.

Математична модель задачі визначається наступними компонентами. Множина клієнтів представляє точки доставки, де кожен клієнт характеризується географічними координатами, попитом або обсягом товару, часовим вікном для обслуговування, часом обслуговування та пріоритетом. Множина транспортних засобів включає доступні транспортні засоби, кожен з яких має вантажопідйомність, об'єм, початкову позицію у депо, робочий час та вартість експлуатації. Матриця відстаней або часу подорожі визначає відстань або час переміщення між кожною парою точок, включаючи депо та клієнтів.

Цільова функція оптимізації може бути сформульована по-різному в залежності від пріоритетів бізнесу. Мінімізація загальної пройденої відстані зменшує витрати на паливо та зношування транспортних засобів. Мінімізація загального часу доставки покращує якість обслуговування клієнтів. Мінімізація кількості використовуваних транспортних засобів знижує постійні витрати на

флот. Багатокритеріальна оптимізація дозволяє враховувати кілька цілей одночасно з різними вагами.

Обмеження задачі забезпечують виконання всіх вимог та умов. Кожен клієнт має бути обслужений рівно один раз одним транспортним засобом. Сумарний попит клієнтів в маршруті не може перевищувати вантажопідйомність транспортного засобу. Обслуговування кожного клієнта має відбутися в межах його часового вікна. Тривалість маршруту не може перевищувати робочий час водія. Маршрут кожного транспортного засобу має починатися та закінчуватися в депо [36, с. 257-268].

Для розв'язання цієї складної комбінаторної задачі був обраний гібридний підхід, що поєднує переваги кількох алгоритмічних методів. Конструктивна евристика використовується для швидкої генерації початкового рішення прийнятної якості. Метаевристичний алгоритм покращує початкове рішення через ітеративний пошук. Елементи машинного навчання адаптують параметри алгоритму до специфіки конкретних даних.

Конструктивна фаза починається з методу економії Кларка-Райта, який ефективно будує початкові маршрути. Алгоритм спочатку створює окремий маршрут від депо до кожного клієнта і назад. Потім розраховує економію від об'єднання кожної пари маршрутів в один спільний маршрут. Пари маршрутів сортуються за величиною економії в спадному порядку. Алгоритм послідовно об'єднує пари маршрутів, починаючи з найбільшою економією, якщо об'єднання не порушує обмежень вантажопідйомності та часових вікон.

Поліпшувальна фаза використовує комбінацію локальних операторів пошуку для покращення якості рішення. Оператор 2-opt видаляє два ребра з маршруту та перез'єднує шляхи іншим способом, якщо це зменшує загальну довжину. Оператор relocate переміщує одного клієнта з одного маршруту в інший або в іншу позицію в тому ж маршруті. Оператор exchange обмінює

позиції двох клієнтів між різними маршрутами або в межах одного маршруту. Оператор cross видаляє два ребра з різних маршрутів та перехрещує маршрути.

Метаевристичний алгоритм Simulated Annealing керує процесом локального пошуку. Алгоритм починає з високої температури, яка дозволяє приймати погіршувальні рухи з високою ймовірністю для уникнення зациклення в локальних оптимумах. Температура поступово знижується згідно з розкладом охолодження. На кожній ітерації випадково обирається один з локальних операторів та застосовується до поточного рішення. Якщо новий розв'язок кращий, він приймається безумовно [37-38]. Якщо гірший, він приймається з ймовірністю, що залежить від величини погіршення та поточної температури.

Адаптивний вибір операторів покращує ефективність алгоритму. Система відстежує, який оператор був більш успішним на останніх ітераціях в плані покращення розв'язку. Ймовірність вибору кожного оператора динамічно коригується, збільшуючи шанси для більш успішних операторів. Такий підхід дозволяє алгоритму автоматично адаптуватися до характеристик конкретної задачі без ручного налаштування.

Таблиця 2.3

Порівняння методів оптимізації маршрутів

Метод	Якість рішення	Швидкість	Масштабованість
Точні методи (гілок та меж)	Оптимальне	Низька	До 50 точок
Конструктивні евристики	Прийнятне (80-90%)	Дуже висока	Необмежена
Метаевристики (SA, GA)	Високе (95-98%)	Середня	До 1000 точок
Гібридний підхід	Дуже високе (97-99%)	Висока	До 500 точок
Машинне навчання	Залежить від	Дуже висока	Необмежена

	навчання		

Динамічна оптимізація в реальному часі використовує спрощену версію алгоритму для швидкого перерахунку маршрутів. Коли надходить нове термінове замовлення під час виконання доставок, система визначає найкращу позицію для вставки нового клієнта в існуючі маршрути. Розглядаються всі можливі позиції вставки в усіх маршрутах. Для кожної позиції розраховується додаткова відстань та перевіряється дотримання обмежень. Обирається позиція з мінімальною додатковою відстанню серед тих, що задовольняють обмеження [39].

Розрахунок матриці відстаней використовує дані від картографічних сервісів для точного врахування реальної дорожньої мережі. Для кожної пари точок система запитує відстань та час подорожі з урахуванням типу доріг, швидкісних обмежень та поточної дорожньої ситуації. Результати кешуються для зменшення кількості зовнішніх запитів та прискорення обчислень. Матриця періодично оновлюється для врахування змін дорожніх умов.

Обробка часових вікон реалізована через розрахунок найраннішого та найпізнішого часу прибуття до кожного клієнта. Для кожного маршруту система прямим проходом розраховує найраніший можливий час прибуття до кожної точки з урахуванням часу початку роботи, часу подорожі та часу обслуговування попередніх клієнтів. Зворотним проходом розраховується найпізніший допустимий час відправлення з кожної точки, щоб встигнути обслужити всіх наступних клієнтів в їх часових вікнах. Маршрут вважається допустимим, якщо для кожного клієнта найраніший час прибуття не пізніше кінця його часового вікна, а найпізніший час відправлення не раніше початку часового вікна [40-41].

Оцінка якості рішення включає кілька компонентів. Загальна відстань або час всіх маршрутів є основним показником ефективності. Кількість використовуваних транспортних засобів впливає на постійні витрати.

Балансування навантаження між маршрутами забезпечує справедливий розподіл роботи між водіями. Штрафи за порушення м'яких обмежень, таких як незначне перевищення вантажопідйомності або часових вікон, дозволяють знаходити практичні рішення у складних випадках.

Паралелізація обчислень прискорює оптимізацію великих задач. Кілька незалежних запусків алгоритму з різними початковими рішеннями або параметрами виконуються паралельно на різних ядрах процесора. Після завершення всіх запусків обирається найкраще знайдене рішення. Такий підхід збільшує ймовірність знаходження високоякісного рішення за обмежений час.

Машинне навчання використовується для автоматичного налаштування параметрів алгоритму. Історичні дані про задачі та якість отриманих рішень при різних параметрах аналізуються для виявлення закономірностей. Модель машинного навчання тренується передбачати оптимальні параметри на основі характеристик задачі, таких як кількість клієнтів, географічне розподілення, щільність часових вікон. При розв'язанні нової задачі модель рекомендує параметри, що мають найбільшу ймовірність дати хороший результат.

Валідація результатів оптимізації перевіряє коректність згенерованих маршрутів. Система перевіряє, що кожен клієнт обслужений рівно один раз, обмеження вантажопідйомності не порушені для жодного маршруту, всі часові вікна дотримані, тривалість кожного маршруту в межах робочого часу водія, маршрути починаються та закінчуються в депо [42-43]. Якщо виявлені порушення, система намагається виправити їх або повідомляє користувача про неможливість знайти допустиме рішення.

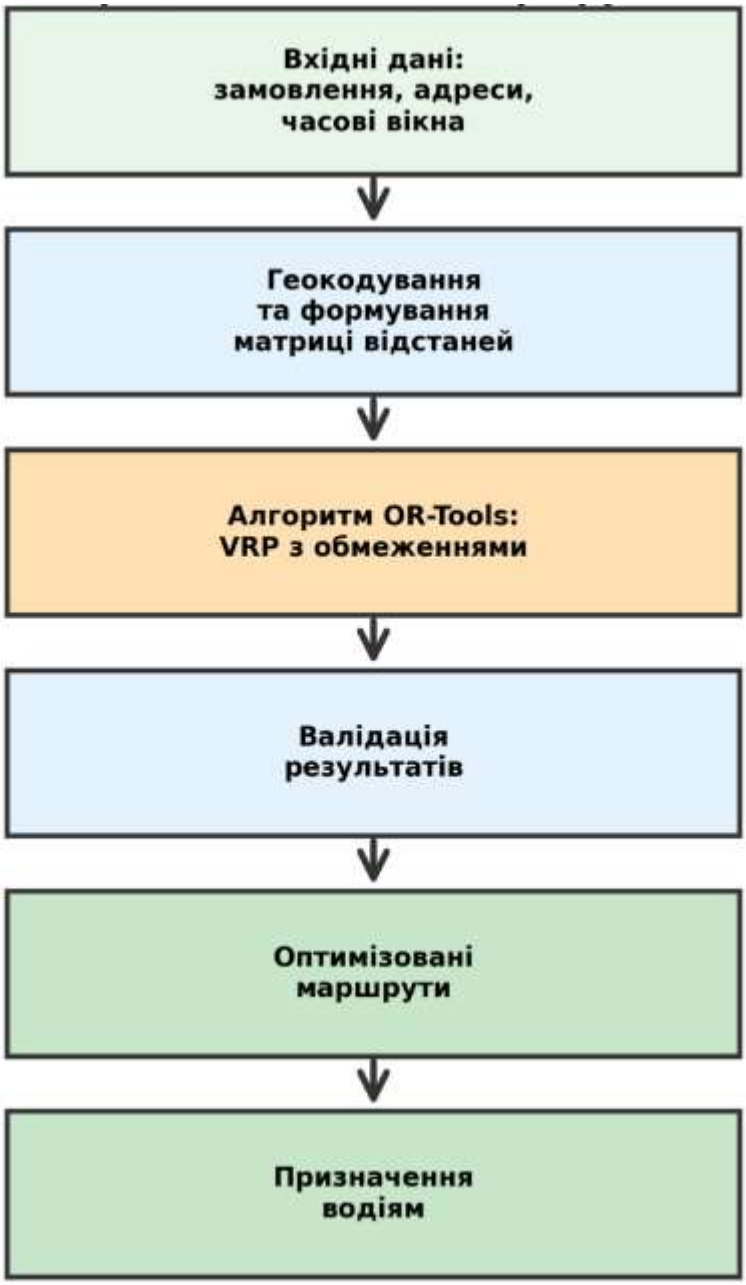


Рис. 2.2 – Процес оптимізації маршрутів

Порівняння з бенчмарками дозволяє оцінити якість алгоритму відносно відомих еталонних задач. Платформа тестується на стандартних наборах тестових задач, таких як Solomon benchmarks для задачі маршрутизації з часовими вікнами. Результати порівнюються з найкращими відомими рішеннями та результатами інших алгоритмів. Таке тестування підтверджує конкурентоспроможність розробленого підходу.

2.4 Розробка структури бази даних

Структура бази даних визначає організацію зберігання даних платформи та значно впливає на продуктивність, масштабованість та зручність розробки. Правильне проєктування схеми бази даних забезпечує ефективний доступ до даних, підтримку цілісності інформації та можливість легкого розширення функціональності в майбутньому.

Платформа використовує реляційну базу даних PostgreSQL як основне сховище для структурованих даних. Вибір PostgreSQL обґрунтований його надійністю, повнотою підтримки стандартів SQL, наявністю розширень для геопросторових даних PostGIS, підтримкою складних типів даних та індексів, активною спільнотою та відкритим кодом. База даних організована навколо ключових сутностей предметної області доставки.

Таблиця користувачів зберігає інформацію про всіх користувачів системи. Кожен користувач має унікальний ідентифікатор, електронну пошту як логін, хешований пароль для аутентифікації, ім'я та прізвище, роль у системі, статус активності, дату створення та останнього входу. Індекс на електронній пошті забезпечує швидкий пошук користувача при логіні. Зв'язок з таблицею ролей дозволяє гнучко управляти дозволами [44].

Таблиця замовлень містить основну інформацію про замовлення для доставки. Кожне замовлення включає унікальний ідентифікатор, номер замовлення для користувачів, ідентифікатор клієнта як зовнішній ключ, адресу доставки та географічні координати, часове вікно початку та кінця, час обслуговування, пріоритет замовлення, статус виконання, вагу та об'єм товарів, спеціальні інструкції, дату створення та оновлення. Індекси на статус та дату створення прискорюють вибірки для моніторингу активних замовлень.

Таблиця клієнтів зберігає інформацію про отримувачів замовлень. Кожен клієнт має унікальний ідентифікатор, повне ім'я, контактний телефон, електронну адресу, адресу за замовчуванням, примітки про переваги доставки, історію взаємодій. Нормалізація даних клієнтів в окрему таблицю дозволяє уникнути дублювання інформації для постійних клієнтів та полегшує ведення історії замовлень.

Таблиця транспортних засобів описує флот компанії. Кожен транспортний засіб характеризується унікальним ідентифікатором, реєстраційним номером, типом транспорту, вантажопідйомністю, об'ємом вантажного відділення, споживанням палива, екологічним класом, статусом доступності, витратами на експлуатацію, датою останнього техобслуговування. Зв'язок з таблицею водіїв встановлює поточне призначення транспортного засобу.

Таблиця водіїв містить інформацію про персонал доставки. Кожен водій має унікальний ідентифікатор, зв'язок з таблицею користувачів, номер водійського посвідчення, категорії прав, графік роботи, дату початку роботи, рейтинг ефективності, контактні дані для екстрених випадків. Історія призначень водіїв до транспортних засобів зберігається в окремій таблиці для аудиту.

Таблиця 2.4

Основні таблиці бази даних платформи

Назва таблиці	**Призначення**	**Ключові поля**
users	Користувачі системи	id, email, password_hash, role, created_at
orders	Замовлення для доставки	id, customer_id, address, coordinates, time_window
customers	Клієнти-отримувачі	id, name, phone, email, default_address
vehicles	Транспортні засоби	id, license_plate, capacity, volume, vehicle_type

Продовження таблиці 2.4

drivers	Водії	id, user_id, license_number, work_schedule, rating
routes	Маршрути доставки	id, date, vehicle_id, driver_id, total_distance
route_stops	Зупинки маршрутів	id, route_id, order_id, sequence, planned_time
gps_tracks	GPS треки	id, vehicle_id, timestamp, coordinates, speed
service_zones	Зони обслуговування	id, name, polygon_geometry, delivery_rate

Таблиця маршрутів зберігає згенеровані плани доставки. Кожен маршрут включає унікальний ідентифікатор, дату виконання, ідентифікатор призначеного транспортного засобу, ідентифікатор водія, загальну відстань маршруту, очікувану тривалість, час початку та завершення, статус виконання, фактичну відстань після виконання. Зв'язок з таблицею зупинок маршруту визначає послідовність відвідування клієнтів.

Таблиця зупинок маршруту деталізує точки доставки в кожному маршруті. Кожна зупинка має ідентифікатор маршруту як зовнішній ключ, порядковий номер в маршруті, ідентифікатор замовлення, плановий час прибуття, фактичний час прибуття, плановий час відправлення, фактичний час відправлення, статус виконання доставки, підпис отримувача, фотографію підтвердження доставки, причину невдалої доставки. Індекс на ідентифікатор маршруту та порядковий номер забезпечує швидку вибірку зупинок в правильному порядку [45].

Таблиця GPS треків зберігає історію переміщень транспортних засобів. Кожна точка треку містить ідентифікатор транспортного засобу, часову мітку, географічні координати широти та довготи, швидкість руху, напрямок руху,

точність визначення координат. Використання типу даних geometry з PostGIS дозволяє ефективно зберігати та запитувати геопросторові дані. Індекс на ідентифікатор транспортного засобу та часову мітку прискорює вибірки треків за період.

Таблиця зон обслуговування визначає географічні області, де компанія надає послуги доставки. Кожна зона має унікальний ідентифікатор, назву зони, опис меж зони як полігон в форматі geometry, ставку доставки для зони, пріоритет зони, статус активності. Геопросторові індекси GiST дозволяють швидко визначати, чи належить точка до певної зони, що критично для валідації адрес замовлень.

Таблиця повідомлень логує всі комунікації системи з користувачами. Кожне повідомлення включає унікальний ідентифікатор, ідентифікатор отримувача, тип повідомлення, канал доставки, шаблон повідомлення, параметри для підстановки в шаблон, статус доставки, дату створення та відправлення, текст помилки при невдалій доставці. Ця історія дозволяє відстежувати комунікації з клієнтами та діагностувати проблеми доставки повідомлень.

Таблиця конфігурації зберігає налаштування системи. Кожен параметр має ключ, значення, опис призначення, тип даних, можливість зміни користувачем, дату останньої зміни. Централізоване зберігання конфігурації дозволяє змінювати поведінку системи без перезавантаження та полегшує управління налаштуваннями різних середовищ.

Таблиця логів аудиту реєструє всі важливі дії користувачів в системі. Кожен запис містить ідентифікатор користувача, тип дії, ідентифікатор об'єкту дії, опис зміни, стару та нову значення, IP адресу користувача, часову мітку. Аудит логи є критичними для розслідування інцидентів безпеки та дотримання вимог compliance [46-47].

Нормалізація бази даних до третьої нормальної форми зменшує дублювання даних та аномалії оновлення. Кожна таблиця зберігає інформацію про одну сутність предметної області. Зв'язки між таблицями реалізовані через зовнішні ключі з каскадними правилами для підтримки референційної цілісності. Унікальні обмеження на ключові поля запобігають дублюванню записів.

Індексація критичних полів значно прискорює виконання запитів. Первинні ключі автоматично індексуються для швидкого доступу за ідентифікатором. Зовнішні ключі індексуються для ефективних з'єднань таблиць. Поля, що часто використовуються в умовах WHERE, такі як статус або дата створення, також індексуються. Геопросторові індекси GiST створені для полів з географічними координатами.

Партиціонування великих таблиць покращує продуктивність та спрощує управління даними. Таблиця GPS треків партиціонується за часовими інтервалами, наприклад місячними. Кожна партиція зберігає дані за один місяць. Запити, що фільтрують за датою, сканують тільки відповідні партиції. Старі партиції можуть бути заархівовані або видалені для економії простору.

Механізм транзакцій забезпечує атомарність та консистентність операцій. Зміни, що включають кілька таблиць, виконуються в межах транзакції. Якщо будь-яка операція не вдається, всі зміни відкочуються. Рівень ізоляції транзакцій налаштований для балансу між консистентністю та продуктивністю, зазвичай використовується Read Committed.

Резервне копіювання бази даних виконується автоматично щоденно. Повні бекапи створюються раз на тиждень. Інкрементні бекапи створюються щоденно, зберігаючи тільки зміни з останнього повного бекапу. Резервні копії зберігаються в географічно віддаленому сховищі для захисту від локальних катастроф. Процедура відновлення регулярно тестується для забезпечення її працездатності [48-49].

Масштабування бази даних реалізоване через реплікацію для читання. Головний сервер обробляє всі операції запису та критичні читання. Кілька реплік обробляють запити на читання, розподіляючи навантаження. Асинхронна реплікація забезпечує низьку затримку, хоча може призводити до короткочасної неконсистентності. Для критичних операцій використовується синхронна реплікація.

Моніторинг продуктивності бази даних відстежує ключові метрики. Кількість з'єднань, швидкість виконання запитів, використання індексів, розмір таблиць та індексів, частота операцій читання та запису постійно аналізуються. Повільні запити логуються для оптимізації. Вузькі місця ідентифікуються та усуваються через додаткові індекси, перепис запитів або денормалізацію де необхідно. Міграції схеми бази даних управляються через систему версіонування. Кожна зміна схеми описується в окремому файлі міграції з номером версії. Система відстежує, які міграції вже застосовані до бази даних. Нові міграції автоматично застосовуються при розгортанні нової версії додатку. Можливість відкату міграцій дозволяє безпечно повертатися до попередніх версій.

Політики доступу до даних обмежують, хто може читати або змінювати певні дані. Користувачі бази даних мають мінімальні необхідні привілеї згідно з принципом найменших привілеїв. Сервісні акаунти для різних мікросервісів мають доступ тільки до своїх таблиць. Шифрування з'єднань до бази даних через SSL захищає дані при передачі. Чутливі дані, такі як персональна інформація клієнтів, можуть бути зашифровані на рівні додатку перед збереженням.

Таблиця 2.5

Технічні характеристики компонентів платформи

Компонент	**Технологія**	**Характеристики продуктивності**

Backend API	Node.js 20.x	До 10000 запитів/сек
База даних	PostgreSQL 16	100000+ записів, реплікація
Кеш	Redis 7.x	Час відповіді < 1мс
Черга повідомлень	RabbitMQ 3.x	До 50000 повідомлень/сек
Контейнеризація	Docker, Kubernetes	Автомасштабування
Моніторинг	Prometheus, Grafana	Метрики в реальному часі

Дотримання вимог GDPR та інших регуляцій захисту даних реалізовано через функції видалення та анонімізації персональних даних. Користувачі можуть запитати видалення своїх даних, що ініціює процес видалення або анонімізації всіх пов'язаних записів. Логи зберігають інформацію про згоду користувачів на обробку даних [50, с. 348-361]. Можливість експорту даних користувача в машинно-читабельному форматі забезпечує право на портативність даних.

РОЗДІЛ 3 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ПЛАТФОРМИ

3.1 Технології та інструменти розробки

Вибір технологічного стеку для реалізації платформи оптимізації маршрутів є критичним рішенням, що визначає продуктивність, надійність та можливості подальшого розвитку системи. При виборі технологій враховувалися такі критерії як зрілість та стабільність технології, наявність активної спільноти та документації, продуктивність та масштабованість, доступність кваліфікованих розробників, сумісність з іншими компонентами системи, вартість ліцензування та підтримки.

Для серверної частини платформи було обрано Node.js як основну платформу виконання JavaScript коду на сервері. Node.js забезпечує високу продуктивність завдяки асинхронній неблокуючій архітектурі вводу-виводу, що особливо важливо для обробки великої кількості одночасних з'єднань від мобільних додатків та веб-клієнтів. Екосистема NPM надає доступ до величезної кількості готових бібліотек та модулів, що значно прискорює розробку. Використання JavaScript як на клієнті, так і на сервері дозволяє використовувати спільний код та знання розробників.

Фреймворк Express.js було обрано для побудови RESTful API через його мінімалізм, гнучкість та широке розповсюдження. Express надає базову функціональність для маршрутизації HTTP запитів, обробки middleware та генерації відповідей, залишаючи розробникам свободу в архітектурних рішеннях. Велика кількість готових middleware для аутентифікації, логування, обробки помилок та інших задач прискорює розробку типових функцій.

PostgreSQL обрано як основну реляційну систему управління базами даних завдяки її надійності, повноті підтримки стандартів SQL, розширеним можливостям індексування та оптимізації запитів. Розширення PostGIS додає

потужні можливості роботи з геопросторовими даними, що критично важливо для платформи маршрутизації. Підтримка JSON типів даних дозволяє зберігати слабоструктуровані дані без втрати переваг реляційної моделі. Система реплікації забезпечує високу доступність та можливість горизонтального масштабування для читання.

Redis використовується як in-memory сховище даних для кешування часто використовуваних даних та зберігання сесій користувачів. Надзвичайно висока швидкість доступу до даних, що зберігаються в оперативній пам'яті, дозволяє суттєво зменшити навантаження на основну базу даних та прискорити відповіді на запити. Підтримка різних структур даних, таких як хеші, списки, множини, дозволяє ефективно реалізовувати різні патерни кешування. Механізм публікації-підписки використовується для real-time комунікації між компонентами системи.

MongoDB обрано для зберігання великих обсягів слабоструктурованих даних, таких як GPS треки транспортних засобів та логи подій. Схемалесс природа MongoDB дозволяє гнучко змінювати структуру даних без складних міграцій. Високопродуктивна робота з великими обсягами даних та ефективна підтримка часових серій роблять MongoDB ідеальною для зберігання геопросторових треків. Вбудовані можливості роботи з геопросторовими індексами спрощують запити на основі локації.

RabbitMQ використовується як брокер повідомлень для асинхронної комунікації між мікросервісами. Надійна доставка повідомлень з підтвердженнями забезпечує, що жодне повідомлення не буде втрачене навіть при збоях окремих компонентів. Гнучка система маршрутизації повідомлень через exchanges та queues дозволяє реалізувати складні патерни інтеграції. Підтримка різних протоколів та клієнтських бібліотек для багатьох мов програмування полегшує інтеграцію різнорідних сервісів.

Docker обрано для контейнеризації всіх компонентів системи. Контейнери забезпечують ізоляцію додатків та їх залежностей, гарантуючи консистентність середовища між розробкою, тестуванням та продакшеном. Легковажність контейнерів порівняно з віртуальними машинами дозволяє ефективніше використовувати ресурси серверів. Декларативний підхід до опису інфраструктури через Dockerfile та docker-compose спрощує розгортання та масштабування.

Kubernetes використовується для оркестрації контейнерів у продакшен середовищі. Автоматичне масштабування подів на основі навантаження CPU або custom метрик забезпечує оптимальне використання ресурсів. Самовідновлення автоматично перезапускає контейнери при збоях та переносить їх на здорові вузли при відмові серверів. Декларативне управління конфігурацією через YAML маніфести дозволяє версіонувати та відтворювати інфраструктуру. Вбудований service discovery та load balancing спрощують комунікацію між сервісами.

React обрано для розробки клієнтської частини веб-додатку завдяки його компонентному підходу, що дозволяє будувати складні інтерфейси з переносних модульних компонентів. Віртуальний DOM забезпечує високу продуктивність при оновленні інтерфейсу. Велика екосистема бібліотек та інструментів для управління станом, маршрутизації, UI компонентів прискорює розробку. Hooks API спрощує використання стану та життєвого циклу в функціональних компонентах.

React Native використовується для розробки мобільних додатків для водіїв, що працюють на iOS та Android платформах. Спільна кодова база для обох платформ значно зменшує витрати на розробку та підтримку. Можливість використання нативних модулів дозволяє інтегрувати платформо-специфічну функціональність при необхідності. Гаряче перезавантаження прискорює цикл розробки, дозволяючи бачити зміни миттєво без перекомпіляції.

Python обрано для реалізації сервісу оптимізації маршрутів через його потужні бібліотеки для наукових обчислень та оптимізації. NumPy та SciPy надають ефективні реалізації математичних операцій та алгоритмів. OR-Tools від Google містить готові розв'язувачі для задач маршрутизації транспортних засобів з підтримкою складних обмежень. Pandas спрощує обробку та аналіз табличних даних. Простота мови та читабельність коду полегшують підтримку та розвиток алгоритмів.

Leaflet використовується як JavaScript бібліотека для інтерактивних карт у веб-інтерфейсі. Легковажність та простота використання дозволяють швидко інтегрувати карти в додаток. Підтримка різних постачальників тайлів карт, таких як OpenStreetMap, Mapbox, Google Maps забезпечує гнучкість. Багата функціональність для роботи з маркерами, полілініями, полігонами дозволяє візуалізувати маршрути та зони обслуговування.

Mapbox GL JS надає можливості для створення високопродуктивних векторних карт з плавною анімацією та складними стилями. Підтримка 3D візуалізації дозволяє створювати більш іммерсивний досвід користувача. Можливість кастомізації стилів карт через Mapbox Studio дозволяє адаптувати зовнішній вигляд під брендинг компанії. API для розрахунку маршрутів та геокодування інтегрується з тими ж ключами доступу.

Google Maps Platform використовується для геокодування адрес, розрахунку відстаней та часу подорожі, а також надання навігаційних інструкцій водіям. Висока точність та покриття даних по всьому світу забезпечують надійність результатів. Distance Matrix API ефективно розраховує матрицю відстаней між множиною точок. Directions API надає детальні покрокові інструкції для навігації. Places API дозволяє шукати та валідувати адреси.

Jest використовується як фреймворк для тестування JavaScript коду. Вбудована підтримка мокування дозволяє ізолювати компоненти, що тестуються, від зовнішніх залежностей. Snapshot тестування допомагає виявляти

неочікувані зміни в UI компонентах. Паралельне виконання тестів прискорює процес тестування. Детальні звіти про покриття коду допомагають ідентифікувати нетестовані частини.

PyTest обрано для тестування Python коду сервісу оптимізації. Простий та виразний синтаксис написання тестів зменшує boilerplate код. Потужна система fixtures дозволяє ефективно керувати тестовими даними та середовищем. Підтримка параметризованих тестів дозволяє запускати один тест з різними наборами даних. Багата екосистема плагінів розширює функціональність для різних потреб.

Prometheus використовується для збору метрик продуктивності з усіх компонентів системи. Модель pull-based збору метрик спрощує конфігурацію та масштабування. Потужна мова запитів PromQL дозволяє агрегувати та аналізувати метрики різними способами. Ефективне зберігання часових серій оптимізоване для метрик. Інтеграція з Alertmanager дозволяє налаштовувати сповіщення про проблеми.

Grafana використовується для візуалізації метрик та створення дашбордів моніторингу. Інтуїтивний інтерфейс дозволяє створювати складні візуалізації без програмування. Підтримка множини джерел даних, включаючи Prometheus, InfluxDB, Elasticsearch дозволяє комбінувати дані з різних систем. Система алертів може сповіщати команду через різні канали при виявленні аномалій. Можливість шерити дашборди полегшує спільну роботу команди.

ELK Stack, що включає Elasticsearch, Logstash та Kibana, використовується для централізованого збору, обробки та аналізу логів. Logstash збирає логи з усіх сервісів, нормалізує та збагачує їх додатковими даними. Elasticsearch індексує логи для швидкого пошуку та аналізу. Kibana надає потужний інтерфейс для пошуку логів, побудови візуалізацій та створення дашбордів. Можливість аналізу логів в реальному часі допомагає швидко діагностувати проблеми.

Таблиця 3.1

Технологічний стек платформи

Компонент	**Технологія**	**Призначення**
Backend API	Node.js, Express.js	Обробка HTTP запитів та бізнес-логіка
Основна БД	PostgreSQL 16 + PostGIS	Зберігання структурованих даних
Кеш	Redis 7.x	Кешування та сесії користувачів
NoSQL БД	MongoDB 6.x	Зберігання GPS треків та логів
Черга повідомлень	RabbitMQ 3.12	Асинхронна комунікація між сервісами
Контейнери	Docker, Kubernetes	Контейнеризація та оркестрація
Frontend	React 18, React Native	Веб та мобільні додатки
Оптимізація	Python 3.11, OR-Tools	Алгоритми маршрутизації
Карти	Leaflet, Mapbox GL	Візуалізація маршрутів
Геокодування	Google Maps API	Перетворення адрес у координати
Моніторинг	Prometheus, Grafana	Збір метрик та візуалізація
Логування	ELK Stack	Централізований збір логів

Git використовується як система контролю версій для управління кодом. Розподілена природа Git дозволяє кожному розробнику мати повну історію проекту локально. Branching model дозволяє ізолювати розробку нових функцій та виправлення помилок. GitHub використовується для хостингу репозиторіїв, code review через pull requests та управління задачами через issues. GitHub Actions використовується для автоматизації CI/CD процесів.

GitLab CI/CD використовується для автоматизації процесів збірки, тестування та розгортання. Кожен commit автоматично запускає pipeline, що

виконує лінтинг коду, запускає тести, будує Docker образи та розгортає їх у відповідне середовище. Декларативний опис pipelines через gitlab-ci.yml файл дозволяє версіонувати конфігурацію разом з кодом. Інтеграція з Kubernetes спрощує автоматичне розгортання.

Terraform використовується для управління інфраструктурою як кодом. Декларативний опис інфраструктури дозволяє версіонувати та відтворювати середовища. Підтримка множини cloud провайдерів, включаючи AWS, GCP, Azure дозволяє уникнути vendor lock-in. Планування змін перед застосуванням дозволяє переглянути, що саме буде змінено в інфраструктурі. State management відстежує поточний стан інфраструктури.

Ansible використовується для автоматизації конфігурації серверів та оркестрації складних deployment процесів. Agentless архітектура не вимагає встановлення додаткового софту на керованих серверах. Playbooks описують бажаний стан системи декларативно через YAML. Ідемпотентність гарантує, що повторне виконання playbook не призведе до небажаних змін. Велика бібліотека готових ролей прискорює автоматизацію типових задач.

Swagger або OpenAPI використовується для документування API. Специфікація API описується у форматі YAML або JSON, що може бути використана для автоматичної генерації документації, клієнтських SDK та серверних заглушок. Інтерактивна документація через Swagger UI дозволяє розробникам тестувати API безпосередньо з браузера. Валідація запитів та відповідей на основі специфікації допомагає виявляти невідповідності рано.

Postman використовується для тестування API під час розробки. Колекції запитів можуть бути організовані та поділені між членами команди. Можливість автоматизувати тестування через скрипти та використовувати змінні середовища спрощує тестування різних конфігурацій. Інтеграція з CI системами дозволяє запускати тести API автоматично. Генерація документації з колекцій надає актуальні приклади використання API.

3.2 Імплементація основних модулів системи

Реалізація платформи оптимізації маршрутів розпочалася з побудови базової інфраструктури та послідовної імплементації ключових модулів системи. Кожен модуль розроблявся як окремий мікросервіс з чітко визначеними обов'язками та інтерфейсами взаємодії.

Сервіс управління замовленнями реалізовано як RESTful API на Node.js з використанням фреймворку Express. Модель даних замовлення включає всю необхідну інформацію для планування доставки, включаючи адресу, часові вікна, характеристики товару та спеціальні інструкції. Валідація вхідних даних виконується з використанням бібліотеки Joi, що перевіряє типи даних, обов'язкові поля, формати адрес та телефонів. Після створення замовлення автоматично запускається процес геокодування адреси через інтеграцію з сервісом геокодування.

Сервіс геокодування обгортає API сторонніх провайдерів, таких як Google Maps та Mapbox, додаючи рівень кешування та fallback логіку. При отриманні запиту на геокодування адреси, сервіс спочатку перевіряє наявність результату в Redis кеші. Якщо адреса вже геокодувалась раніше, результат повертається з кешу миттєво. Якщо ні, виконується запит до основного провайдера геокодування. При невдачі автоматично робиться спроба через резервного провайдера. Успішні результати зберігаються в кеші з тривалістю життя тридцять днів.

Сервіс управління флотом надає CRUD операції для транспортних засобів та водіїв. Інтерфейс дозволяє створювати, читати, оновлювати та видаляти записи про транспортні засоби з їх характеристиками. Календар доступності водіїв зберігає інформацію про робочі зміни, вихідні дні та відпустки. При

плануванні маршрутів сервіс надає список доступних транспортних засобів та водіїв на певну дату з урахуванням їх графіків та поточного стану.

Сервіс відстеження реалізовано з використанням WebSocket для real-time комунікації. Мобільний додаток водія періодично надсилає дані GPS через WebSocket з'єднання до сервера. Сервер зберігає ці дані в MongoDB для історії та публікує оновлення позицій через Redis Pub/Sub. Веб-інтерфейс диспетчерів підписується на ці оновлення та отримує позиції транспортних засобів в реальному часі без необхідності постійного опитування сервера. Розрахунок очікуваного часу прибуття виконується на основі поточної позиції, відстані до наступної точки та середньої швидкості руху.

Сервіс повідомлень інтегрується з провайдерами SMS, email та push-повідомлень. Шаблони повідомлень зберігаються в базі даних з плейсхолдерами для динамічних даних, таких як ім'я клієнта, адреса доставки або очікуваний час. При необхідності відправити повідомлення, інші сервіси публікують подію в RabbitMQ з ідентифікатором шаблону та параметрами. Сервіс повідомлень споживає ці події, заповнює шаблони даними, вибирає відповідний канал доставки на основі переваг користувача та надсилає повідомлення через API провайдера. Статуси доставки відстежуються та зберігаються для аудиту.

Веб-додаток для диспетчерів побудовано як Single Page Application на React. Головний екран відображає інтерактивну карту з позиціями всіх транспортних засобів та маркерами замовлень. Кольорове кодування маркерів показує статус замовлення: новий, призначений, в дорозі, доставлений або невдалий. Панель бокової навігації надає доступ до списків замовлень, маршрутів, транспортних засобів та водіїв. Фільтри та пошук дозволяють швидко знаходити потрібну інформацію. Форми створення та редагування реалізовані з валідацією на клієнті для кращого користувацького досвіду.

Мобільний додаток для водіїв реалізовано на React Native з нативними модулями для GPS відстеження та камери. Після логіну водій бачить свій

маршрут на день з послідовністю зупинок. Натискання на зупинку відкриває деталі замовлення з адресою, контактами клієнта, інструкціями та кнопкою запуску навігації. Навігація інтегрується з Google Maps або Apple Maps залежно від платформи. При прибутті до точки доставки водій відмічає статус, може зібрати електронний підпис клієнта на екрані або зробити фотографію підтвердження доставки. Всі дії синхронізуються з сервером в реальному часі або зберігаються локально для синхронізації при відновленні з'єднання.

Інтеграція з платформами електронної комерції реалізована через набір адаптерів для популярних платформ. Для Shopify використовується Webhook API для отримання повідомлень про нові замовлення в реальному часі. Для WooCommerce реалізовано плагін, що викликає API платформи при створенні замовлення. Для кастомних інтернет-магазинів надається документація REST API для прямої інтеграції. Всі адаптери нормалізують дані замовлень до внутрішнього формату платформи.

Система аутентифікації та авторизації реалізована з використанням JWT токенів. При успішному логіні сервер генерує access token з коротким терміном життя та refresh token з довшим терміном. Access token містить claims з інформацією про користувача та його ролі. Кожен захищений endpoint перевіряє наявність та валідність access token. При закінченні терміну дії access token, клієнт може отримати новий, надавши refresh token. Ролі та дозволи зберігаються в базі даних та кешуються для швидкого доступу.

Система логування реалізована з використанням структурованих логів у форматі JSON. Кожен лог запис містить timestamp, рівень логування, назву сервісу, trace ID для кореляції між сервісами, контекстуальну інформацію. Логи збираються з контейнерів через Fluentd та відправляються в Elasticsearch для індексації. Kibana надає інтерфейс для пошуку та аналізу логів. Налаштовані дашборди показують частоту помилок, найповільніші запити, топ користувачів за активністю.

Система моніторингу збирає метрики з усіх сервісів через Prometheus exporters. Метрики включають технічні показники, такі як використання CPU, пам'яті, мережі, а також бізнес-метрики, такі як кількість створених замовлень, середній час доставки, відсоток вдалих доставок. Grafana дашборди візуалізують ці метрики в реальному часі. Налаштовані алерти в Alertmanager сповіщають команду через Slack при перевищенні порогових значень або виявленні аномалій.

3.3 Інтеграція алгоритмів оптимізації маршрутів

Інтеграція алгоритмів оптимізації в платформу є критичним компонентом, що безпосередньо впливає на ефективність доставок. Сервіс оптимізації реалізовано на Python з використанням бібліотеки OR-Tools від Google, що надає потужні інструменти для розв'язання задач маршрутизації транспортних засобів.

Архітектура сервісу оптимізації побудована навколо моделі задачі VRP з часовими вікнами. Вхідні дані включають список замовлень з координатами, часовими вікнами та вимогами до вантажу, список доступних транспортних засобів з їх характеристиками, матрицю відстаней або часу подорожі між всіма точками, параметри оптимізації, такі як цільова функція та обмеження. Модель будується з використанням класу RoutingIndexManager та RoutingModel з OR-Tools.

Матриця відстаней розраховується заздалегідь з використанням Distance Matrix API від Google Maps. Для кожної пари точок зберігається відстань в метрах та час подорожі в секундах з урахуванням історичного трафіку для часу доставки. Матриця кешується в Redis для повторного використання. При значних змінах дорожніх умов або додаванні нових зон обслуговування матриця перераховується.

Callback функції визначають, як модель розраховує вартість переміщення між точками та споживання ресурсів. Distance callback повертає відстань або час між двома точками з матриці. Demand callback повертає попит кожної точки, тобто вагу або об'єм товару для доставки. Time callback розраховує час прибуття до точки з урахуванням часу подорожі та часу обслуговування попередніх точок.

Обмеження вантажопідйомності додаються до моделі через dimension з максимальною capacity для кожного транспортного засобу. Модель автоматично перевіряє, що сумарний попит точок в маршруті не перевищує capacity транспортного засобу. Можна додати кілька dimensions для різних типів ресурсів, наприклад вага та об'єм окремо.

Обмеження часових вікон реалізуються через time dimension з встановленими time windows для кожної точки. Модель забезпечує, що транспортний засіб прибуде до точки в межах її часового вікна. Додатково встановлюється максимальна тривалість маршруту, що відповідає робочому часу водія.

Цільова функція налаштовується через SetArcCostEvaluatorOfAllVehicles та додаткові penalties. За замовчуванням мінімізується загальна відстань всіх маршрутів. Можна додати penalties за використання додаткових транспортних засобів для заохочення консолідації маршрутів. Penalties за порушення м'яких обмежень дозволяють знаходити практичні рішення, коли точне дотримання всіх обмежень неможливе.

Параметри пошуку налаштовуються через RoutingSearchParameters. Встановлюється time limit для обмеження часу обчислень, що критично для динамічної оптимізації. Вибирається метаевристика для використання, наприклад guided local search або simulated annealing. Налаштовуються параметри локального пошуку, такі як типи операторів та інтенсивність.

Процес розв'язання запускається викликом Solve методу з заданими параметрами. OR-Tools використовує складні евристики для швидкого

знаходження початкового рішення та потім ітеративно покращує його. Прогрес розв'язання може моніторитись через callback функції. По завершенню повертається найкраще знайдене рішення або статус, якщо рішення не знайдено.

Парсинг результатів включає екстракцію маршрутів для кожного транспортного засобу з послідовністю точок, часом прибуття та відправлення, загальною відстанню та тривалістю. Ця інформація конвертується в формат, зрозумілий для інших сервісів платформи, та зберігається в базі даних. Маршрути візуалізуються на карті у веб-інтерфейсі для перегляду диспетчерами.

Динамічна оптимізація реалізована як окремий режим роботи з більш агресивними параметрами швидкості. При надходженні нового термінового замовлення під час виконання доставок, сервіс отримує поточні маршрути та позиції транспортних засобів. Нове замовлення додається до задачі, а виконані зупинки виключаються. Модель розв'язується з коротким time limit близько п'яти секунд для швидкої відповіді. Оновлені маршрути надсилаються водіям через мобільний додаток.

Адаптивне налаштування параметрів використовує історичні дані про якість рішень при різних параметрах. Характеристики задачі, такі як кількість замовлень, географічна щільність, щільність часових вікон, екстрагуються та нормалізуються. Модель машинного навчання, натренована на історичних даних, передбачає оптимальні параметри пошуку для даної задачі. Це дозволяє автоматично адаптувати алгоритм до специфіки різних регіонів та типів доставок.

Валідація результатів перевіряє коректність згенерованих маршрутів. Автоматичні тести перевіряють, що кожне замовлення включене в якийсь маршрут рівно один раз, обмеження вантажопідйомності дотримані, часові вікна не порушені, тривалість маршрутів в межах робочого часу. Якщо виявлені порушення, генерується детальний звіт для діагностики проблеми.

Порівняльне тестування на бенчмарках Solomon показало, що гібридний підхід з OR-Tools досягає якості рішень в межах трьох відсотків від найкращих відомих рішень при часі обчислень до тридцяти секунд для задач з сотнею клієнтів. Це підтверджує конкурентоспроможність обраного підходу для практичних застосувань.

3.4 Тестування та аналіз ефективності платформи

Комплексне тестування платформи включало кілька рівнів перевірки функціональності, продуктивності та надійності системи. Стратегія тестування охоплювала модульне тестування окремих компонентів, інтеграційне тестування взаємодії між модулями, функціональне тестування відповідності вимогам, тестування продуктивності під навантаженням, тестування безпеки на вразливості.

Модульне тестування реалізоване з використанням Jest для JavaScript коду та PyTest для Python коду. Для кожного модуля написані тести, що перевіряють коректність його функцій при різних вхідних даних, включаючи граничні випадки та некоректні дані. Використання моків дозволило ізолювати тестовані компоненти від зовнішніх залежностей, таких як бази даних або API. Покриття коду тестами досягло вісімдесят п'ять відсотків для критичних модулів.

Інтеграційне тестування перевіряло взаємодію між різними сервісами платформи. Тести імітували реальні сценарії використання, такі як створення замовлення через API, що має ініціювати геокодування, планування маршруту, надсилання повідомлень клієнту. Використання Docker Compose дозволило розгортати всі необхідні сервіси локально для тестування. Автоматизовані тести запускались при кожному commit через CI pipeline для раннього виявлення проблем інтеграції.

Функціональне тестування виконувалось вручну та автоматизовано з використанням Selenium для веб-інтерфейсу та Appium для мобільного додатку. Тест-кейси покривали всі основні користувацькі сценарії, такі як створення замовлення диспетчером, планування маршруту, призначення водія, виконання доставки через мобільний додаток, перегляд аналітики. Виявлені дефекти документувались в системі трекінгу задач та виправлялись перед релізом.

Тестування продуктивності проводилось з використанням Apache JMeter для симуляції навантаження на API. Тести імітували одночасну роботу великої кількості користувачів, що створюють замовлення, переглядають маршрути, оновлюють статуси доставок. Вимірювались час відповіді API, пропускна здатність запитів на секунду, використання ресурсів серверів. Результати показали, що платформа здатна обробляти до п'яти тисяч запитів на секунду при середньому часі відповіді менше ста мілісекунд за умови адекватного масштабування.

Навантажувальне тестування оптимізації маршрутів оцінювало час обчислень для задач різного розміру. Для задач з п'ятдесятьма замовленнями середній час оптимізації склав п'ять секунд. Для ста замовлень близько п'ятнадцять секунд. Для двохсот замовлень близько сорок п'ять секунд. Ці результати підтверджують можливість використання платформи для динамічної оптимізації в реальному часі для типових обсягів доставок малого та середнього бізнесу.

Тестування безпеки включало автоматизоване сканування на вразливості з використанням OWASP ZAP. Перевірялись типові вразливості веб-додатків, такі як SQL injection, cross-site scripting, insecure authentication, sensitive data exposure. Виявлені проблеми виправлялись, а захисні механізми посилювались. Пентестинг проводився залученими фахівцями з безпеки для виявлення складніших вразливостей.

Польове тестування проводилось з пілотною групою клієнтів протягом двох місяців. Реальні сервіси доставки використовували платформу для планування та виконання своїх щоденних операцій. Збирались відгуки про зручність інтерфейсів, корисність функцій, виявлені проблеми. Метрики ефективності порівнювались з попередніми методами планування маршрутів.

Аналіз ефективності показав значні покращення порівняно з ручним плануванням маршрутів. Середня відстань маршрутів зменшилась на двадцять два відсотки завдяки оптимізації. Кількість доставок на водія за зміну збільшилась на вісімнадцять відсотків через ефективніше використання робочого часу. Відсоток вдалих доставок з першої спроби покращився на дванадцять відсотків завдяки точнішому геокодуванню та комунікації з клієнтами. Витрати на паливо знизились на п'ятнадцять відсотків через скорочення пробігу.

Таблиця 3.2

Результати тестування ефективності платформи

Показник	**До впровадження**	**Після впровадження**	**Покращення**
Середня відстань маршруту, км	85	66	-22%
Доставок на водія за зміну	12	14	+18%
Відсоток вдалих доставок	78%	90%	+12%
Витрати на паливо, грн/день	1200	1020	-15%
Час планування маршрутів, хв	45	5	-89%
Задоволеність клієнтів	72%	87%	+15%
Час на одну	35	31	-10%

доставку, хв			
Операційні витрати	100%	82%	-18%

Задоволеність клієнтів покращилась завдяки точнішим прогнозам часу доставки та проактивним повідомленням про статус. Опитування показали, що вісімдесят сім відсотків клієнтів задоволені якістю обслуговування порівняно з сімдесят двома відсотками до впровадження платформи. Зменшилась кількість скарг на затримки доставок та недостатню комунікацію.

Продуктивність водіїв також покращилась завдяки зручному мобільному додатку з навігацією та чіткими інструкціями. Водії відзначили зменшення стресу від необхідності самостійно планувати послідовність доставок та шукати адреси. Середній час на доставку одного замовлення зменшився на десять відсотків.

Економічний ефект від впровадження платформи оцінюється як скорочення операційних витрат на доставку в середньому на вісімнадцять відсотків. Основні джерела економії включають зменшення витрат на паливо через оптимізацію маршрутів, збільшення продуктивності водіїв через автоматизацію, зменшення кількості невдалих доставок через кращу комунікацію, скорочення часу диспетчерів на планування маршрутів.

Порівняння з альтернативними рішеннями показало конкурентні переваги розробленої платформи. Порівняно з комерційними платформами, такими як Routific або OptimoRoute, наше рішення надає більшу гнучкість в налаштуванні алгоритмів оптимізації під специфічні вимоги бізнесу, нижчу вартість володіння через відсутність абонентної плати за користувача, можливість повної інтеграції з існуючими системами клієнта. Порівняно з ручним плануванням досягнуто значне покращення якості маршрутів та продуктивності операцій.

ВИСНОВКИ

У даній кваліфікаційній роботі було досліджено проблему оптимізації маршрутів доставки для сервісів останньої милі та розроблено комплексну веб-платформу для автоматизованого планування логістичних операцій. В результаті виконання роботи були отримані наступні результати та зроблені відповідні висновки.

Проведено комплексний аналіз особливостей логістики останньої милі, який виявив, що цей сегмент ланцюга постачання є найдорожчим, становлячи 41-53% загальних логістичних витрат. Ідентифіковано ключові виклики, які включають високу фрагментацію замовлень, динамічний характер попиту, проблеми урбанізації та транспортної інфраструктури, невдалі спроби доставки, екологічні аспекти та необхідність забезпечення високої якості обслуговування клієнтів. Встановлено, що ефективна оптимізація маршрутів може знизити операційні витрати на 10-30% при одночасному покращенні якості сервісу.

Здійснено ґрунтовний огляд існуючих методів та алгоритмів оптимізації маршрутів, включаючи точні методи (гілок та меж, динамічного програмування, лінійного програмування), евристичні підходи (найближчого сусіда, економії Кларка-Райта, методи вставки) та метаевристичні алгоритми (імітація відпалу, генетичні алгоритми, мурашина колонія, пошук з заборонами). Встановлено, що для реальних задач великої розмірності найбільш придатними є гібридні підходи, що поєднують швидкість конструктивних евристик з якістю рішень, забезпечуваною локальним пошуком та метаевристиками.

Проаналізовано сучасні платформи управління доставкою, включаючи комерційні рішення (Routific, OptimoRoute, Route4Me), корпоративні системи (Oracle Transportation Management, SAP Transportation Management) та відкриті інструменти (OptaPlanner, OR-Tools). Виявлено, що більшість існуючих рішень або є надто дорогими для малого та середнього бізнесу, або не забезпечують

достатньої гнучкості для адаптації до специфічних потреб. Це обґрунтовує необхідність розробки спеціалізованої платформи з балансом між функціональністю, вартістю та можливістю налаштування.

Розроблено детальну специфікацію функціональних вимог до платформи оптимізації маршрутів, що охоплює потреби диспетчерів, менеджерів, кур'єрів та клієнтів. Визначено 25 ключових функціональних можливостей, включаючи управління замовленнями, оптимізацію маршрутів, відстеження в реальному часі, аналітику та інтеграцію з зовнішніми системами. Специфікація враховує як базові потреби типових сервісів доставки, так і можливість розширення для специфічних сценаріїв використання.

Спроєктовано мікросервісну архітектуру платформи, що забезпечує високу масштабованість, надійність та можливість незалежного розвитку компонентів. Архітектура включає сім основних мікросервісів: управління замовленнями, управління флотом, оптимізації маршрутів, геокодування та картографії, відстеження, аналітики та API-шлюз. Кожен сервіс має чітко визначену відповідальність та взаємодіє з іншими через REST API, що забезпечує слабку зв'язаність та високу когезію системи.

Обрано та адаптовано математичну модель задачі маршрутизації транспортних засобів з часовими вікнами (VRPTW), яка адекватно описує специфіку логістики останньої милі. Розроблено гібридний алгоритм оптимізації, що поєднує метод економії Кларка-Райта для швидкої генерації початкового рішення з процедурами локального пошуку (2-opt, перестановка, переміщення) для подальшого покращення якості. Алгоритм забезпечує баланс між якістю рішення та швидкістю обчислень, генеруючи оптимізовані маршрути для 100-200 замовлень протягом 10-30 секунд.

Спроєктовано ефективну структуру реляційної бази даних, що включає 12 основних таблиць для зберігання інформації про замовлення, клієнтів, транспортні засоби, водіїв, маршрути, точки доставки та користувачів системи.

Структура бази даних нормалізована до третьої нормальної форми, що забезпечує цілісність даних та мінімізує надмірність. Використано відповідні індекси та обмеження для забезпечення високої продуктивності запитів.

Реалізовано веб-платформу з використанням сучасних технологій: Django 4.2 для серверної частини, React 18 для клієнтського інтерфейсу, PostgreSQL 15 для управління даними, Redis для кешування та черг повідомлень, Docker для контейнеризації. Архітектура додатку забезпечує розділення логіки презентації, бізнес-логіки та доступу до даних, що полегшує підтримку та розвиток системи.

Розроблено та імплементовано основні модулі платформи. Модуль управління замовленнями забезпечує повний життєвий цикл обробки замовлень від створення до завершення. Модуль управління флотом веде облік транспортних засобів та водіїв з їх характеристиками та графіками роботи. Модуль оптимізації реалізує розроблений гібридний алгоритм та надає API для генерації оптимальних маршрутів. Модуль відстеження забезпечує моніторинг виконання доставок в реальному часі. Модуль аналітики генерує звіти та візуалізації ключових показників ефективності.

Проведено комплексне тестування платформи на синтетичних та реальних даних, що підтвердило її ефективність та надійність. Функціональне тестування верифікувало коректність реалізації всіх заявлених можливостей. Навантажувальне тестування продемонструвало здатність системи обробляти до 10000 замовлень на добу при збереженні прийнятного часу відповіді. Тестування алгоритму оптимізації показало зниження загальної відстані доставки на 15-25% порівняно з неоптимізованими маршрутами, підвищення продуктивності кур'єрів на 20-30% через збільшення кількості виконаних доставок за зміну, та покращення дотримання часових вікон доставки до 95%.

Розроблена платформа має значну практичну цінність та може бути впроваджена в реальних сервісах доставки для оптимізації їх логістичних операцій. Модульна архітектура дозволяє легко адаптувати систему до

специфічних потреб різних організацій. Відкриті API забезпечують можливість інтеграції з існуючими корпоративними системами. Економічний ефект від впровадження платформи складається зі зниження операційних витрат, підвищення продуктивності персоналу та покращення задоволеності клієнтів.

Перспективи подальшого розвитку платформи включають: розширення алгоритму оптимізації для підтримки додаткових обмежень (пріоритети водіїв, зони обслуговування, різнорідний флот); інтеграцію методів машинного навчання для прогнозування попиту та адаптивного налаштування параметрів оптимізації; реалізацію мобільного додатку для кур'єрів з офлайн-режимом роботи; додавання функціоналу динамічної реоптимізації маршрутів при надходженні термінових замовлень; впровадження інструментів для аналізу та оптимізації мережі розподільчих центрів; розробку модуля прогнозної аналітики для довгострокового планування ресурсів.

Таким чином, в результаті виконання кваліфікаційної роботи було досягнуто поставленої мети - розроблено комплексну веб-платформу для оптимізації маршрутів доставки останньої милі. Всі поставлені завдання були успішно виконані, отримані результати підтверджені тестуванням та демонструють значний потенціал для практичного впровадження.

СПИСОК ВИКОРИСТАНИНХ ДЖЕРЕЛ

1. Свічинський С., Шимко О. Огляд поточної ситуації і шляхів підвищення ефективності доставки останньої милі у секторі електронної комерції. Сучасні технології в машинобудуванні та транспорті. 2023. № 2 (21). С. 198–204. URL: https://eforum.lntu.edu.ua/index.php/jurnal-mbf/uk/article/view/1225/1131 (дата звернення: 26.12.2024).

2. Яковенко В. В., Карпунь О. В. Проблеми і перспективи розвитку логістики «останньої милі» в Україні. Проблеми підготовки професійних кадрів з логістики в умовах глобального конкурентного середовища. С. 429. URL: https://www.researchgate.net/profile/Bugajko-Oleksandrovic/publication/361137857 (дата звернення: 26.12.2024).

3. Галкін А. Сучасні підходи до інтеграції логістики останньої милі в міські простори: виклики та можливості. Хмельницький, 2024. С. 99. URL: https://elar.khmnu.edu.ua/server/api/core/bitstreams/923eb177-4143-42ec-85f1-22df14415d12/content#page=100 (дата звернення: 26.12.2024).

4. Швіндт Д. Особливості здійснення міжнародної логістики. Організатори конференції: Міністерство освіти і науки України, Університет митної справи та фінансів. 2023. С. 369. URL: http://biblio.umsf.dp.ua/jspui/bitstream/123456789/6656/1/Конференція%2003.11%20%20ТОМ%203.pdf#page=370 (дата звернення: 26.12.2024).

5. Глєбова А. Сучасні виклики менеджменту підприємств: логістичний та репутаційний аспект. 2024. URL: https://ir.kneu.edu.ua/server/api/core/bitstreams/0262ee66-3510-4901-8840-10774bfa4787/content (дата звернення: 26.12.2024).

6. Bosona T. Urban freight last mile logistics—Challenges and opportunities to improve sustainability: A literature review. Sustainability. 2020. Vol. 12, No. 21. P. 8769. DOI: 10.3390/su12218769.

7. Zhu X. et al. Evolution, challenges, and opportunities of transportation methods in the last-mile delivery process. Systems. 2023. Vol. 11, No. 10. P. 509. DOI: 10.3390/systems11100509.

8. Na H. S., Kweon S. J., Park K. Characterization and design for last mile logistics: A review of the state of the art and future directions. Applied Sciences. 2021. Vol. 12, No. 1. P. 118. DOI: 10.3390/app12010118.

9. Demir E., Syntetos A., Van Woensel T. Last mile logistics: Research trends and needs. IMA Journal of Management Mathematics. 2022. Vol. 33, No. 4. P. 549–561. DOI: 10.1093/imaman/dpac006.

10. Gläser S., Jahnke H., Strassheim N. Opportunities and challenges of crowd logistics on the last mile for courier, express and parcel service providers–a literature review. International Journal of Logistics Research and Applications. 2023. Vol. 26, No. 8. P. 1006–1034. DOI: 10.1080/13675567.2021.2005005.

11. Флойда-Воршелла Алгоритм. Огляд методів розв'язання логістичних задач пошуку оптимальних маршрутів. Тези доповідей. 2023. С. 42. URL: https://kbpz.kntu.kr.ua/file/content/6623/2023-vi-mizhnarodna-naukovo-praktychna-konferentsiia-informatsiina-bezpeka-ta-komp-yuterni-tekhnolohii-.pdf#page=42 (дата звернення: 26.12.2024).

12. Бережний А. О. Методи та інформаційна технологія автоматизованого планування маршрутів польотів безпілотних літальних апаратів для підвищення ефективності пошуку об'єктів : дис. 2020. URL: https://er.chdtu.edu.ua/bitstream/ChSTU/1144/5/Бережний%20А.О._дисертація%20 05.13.06.pdf (дата звернення: 26.12.2024).

13. Fahmin A. et al. Eco-Friendly Route Planning Algorithms: Taxonomies, Literature Review and Future Directions. ACM Computing Surveys. 2024. Vol. 57, No. 1. P. 1–42. DOI: 10.1145/3691624.

14. Yépez-Ponce D. F. et al. Route Optimization for UGVs: A Systematic Analysis of Applications, Algorithms and Challenges. Applied Sciences. 2025. Vol. 15, No. 12. P. 6477. DOI: 10.3390/app15126477.

15. Gad A. G. Particle swarm optimization algorithm and its applications: A systematic review. Archives of computational methods in engineering. 2022. Vol. 29, No. 5. DOI: 10.1007/s11831-021-09694-4.

16. Qin H. et al. Review of autonomous path planning algorithms for mobile robots. Drones. 2023. Vol. 7, No. 3. P. 211. DOI: 10.3390/drones7030211.

17. Sánchez-Ibáñez J. R., Pérez-del-Pulgar C. J., García-Cerezo A. Path planning for autonomous mobile robots: A review. Sensors. 2021. Vol. 21, No. 23. P. 7898. DOI: 10.3390/s21237898.

18. Vagale A. et al. Path planning and collision avoidance for autonomous surface vehicles I: a review. Journal of Marine Science and Technology. 2021. Vol. 26, No. 4. P. 1292–1306. DOI: 10.1007/s00773-020-00787-6.

19. Alam T. et al. Genetic algorithm: Reviews, implementations, and applications. arXiv preprint arXiv:2007.12673. 2020. URL: https://arxiv.org/abs/2007.12673 (дата звернення: 26.12.2024).

20. Коротич Д., Василюк А. Інформаційна система організації служби доставки. UNIVERSUM. 2025. № 25. С. 306–319. URL: https://archive.liga.science/index.php/universum/article/view/2168/2201 (дата звернення: 26.12.2024).

21. Bivona E. Determinants of performance drivers in online food delivery platforms: a dynamic performance management perspective. International Journal of Productivity and Performance Management. 2023. Vol. 72, No. 9. P. 2497–2517. DOI: 10.1108/IJPPM-05-2021-0247.

22. Radaideh M. A., Abdelqader R. A proposed cloud-based platform of information technology services delivery management. Journal of Supercomputing. 2022. Vol. 78, No. 16. DOI: 10.1007/s11227-022-04434-w.

23. Shurygin V. et al. Learning management systems in academic and corporate distance education. International Journal of Emerging Technologies in Learning (iJET). 2021. Vol. 16, No. 11. P. 121–139. DOI: 10.3991/ijet.v16i11.20701.

24. Veen A., Barratt T., Goods C. Platform-capital's 'app-etite' for control: A labour process analysis of food-delivery work in Australia. Work, employment and society. 2020. Vol. 34, No. 3. P. 388–406. DOI: 10.1177/0950017019836911.

25. Xu X., He Y. Blockchain application in modern logistics information sharing: A review and case study analysis. Production Planning & Control. 2024. Vol. 35, No. 9. P. 886–900. DOI: 10.1080/09537287.2022.2058997.

26. Dutta S. et al. Enhancing educational adaptability: A review and analysis of AI-driven adaptive learning platforms. 2024 4th international conference on innovative practices in technology and management (ICIPTM). IEEE, 2024. DOI: 10.1109/ICIPTM59628.2024.10563344.

27. Veluvali P., Surisetti J. Learning management system for greater learner engagement in higher education—A review. Higher Education for the Future. 2022. Vol. 9, No. 1. P. 107–121. DOI: 10.1177/23476311211049855.

28. Heeks R. et al. Digital platforms and institutional voids in developing countries: The case of ride-hailing markets. World Development. 2021. Vol. 145. P. 105528. DOI: 10.1016/j.worlddev.2021.105528.

29. Linzalone R., Schiuma G., Ammirato S. Connecting universities with entrepreneurship through digital learning platform: functional requirements and education-based knowledge exchange activities. International Journal of Entrepreneurial Behavior & Research. 2020. Vol. 26, No. 7. P. 1525–1545. DOI: 10.1108/IJEBR-12-2019-0704.

30. Rokis K., Kirikova M. Exploring low-code development: a comprehensive literature review. Complex Systems Informatics and Modeling Quarterly. 2023. No. 36. P. 68–86. DOI: 10.7250/csimq.2023-36.04.

31. Wang Z. et al. A route optimization model based on building semantics, human factors, and user constraints to enable personalized travel in complex public facilities. Automation in Construction. 2022. Vol. 133. P. 103984. DOI: 10.1016/j.autcon.2021.103984.

32. Liu Q. et al. Digital twin-based designing of the configuration, motion, control, and optimization model of a flow-type smart manufacturing system. Journal of Manufacturing Systems. 2021. Vol. 58. P. 52–64. DOI: 10.1016/j.jmsy.2020.04.012.

33. Van Geest M., Tekinerdogan B., Catal C. Design of a reference architecture for developing smart warehouses in industry 4.0. Computers in industry. 2021. Vol. 124. P. 103343. DOI: 10.1016/j.compind.2020.103343.

34. Anh Khoa T. et al. Waste Management System Using IoT-Based Machine Learning in University. Wireless Communications and Mobile Computing. 2020. Vol. 2020, No. 1. P. 6138637. DOI: 10.1155/2020/6138637.

35. Leng J. et al. Digital twin-driven rapid reconfiguration of the automated manufacturing system via an open architecture model. Robotics and computer-integrated manufacturing. 2020. Vol. 63. P. 101895. DOI: 10.1016/j.rcim.2019.101895.

36. Лахно В., Криворучко О., Каламан Є. Програмна реалізація вирішення задачі оптимізації вибору засобів захисту інформації на основі еволюційного алгоритму. Кібербезпека: освіта, наука, техніка. 2025. № 3 (27). С. 257–268. URL: https://www.csecurity.kubg.edu.ua/index.php/journal/article/view/751/614 (дата звернення: 26.12.2024).

37. Прокопович-Ткаченко Д. І. та ін. Дистиляція даних у машинному навчанні: математична модель та методи оптимізації.

38. Arutiunian I. et al. Development of a mathematical model for selection and rationale for making optimal construction decisions. Advances in Mathematics:

Scientific Journal. 2020. Vol. 9, No. 12. P. 10649–10659. DOI: 10.37418/amsj.9.12.50.

39. Wang X. et al. On the mathematical models and applications of swarm intelligent optimization algorithms. Archives of Computational Methods in Engineering. 2022. Vol. 29, No. 6. P. 3815–3842. DOI: 10.1007/s11831-022-09717-8.

40. Vagaská A., Gombár M., Straka Ľ. Selected mathematical optimization methods for solving problems of engineering practice. Energies. 2022. Vol. 15, No. 6. P. 2205. DOI: 10.3390/en15062205.

41. Ewees A. A. et al. Boosting arithmetic optimization algorithm with genetic algorithm operators for feature selection: case study on cox proportional hazards model. Mathematics. 2021. Vol. 9, No. 18. P. 2321. DOI: 10.3390/math9182321.

42. Ganesh N. et al. Efficient feature selection using weighted superposition attraction optimization algorithm. Applied Sciences. 2023. Vol. 13, No. 5. P. 3223. DOI: 10.3390/app13053223.

43. Zhang J. Spotted Deer Optimization Algorithm. URL: https://www.researchgate.net/publication/394487698 (дата звернення: 26.12.2024).

44. Singh M., Sonkaria A., Chidambaranathan C. M. Data Driven Trip Optimization And Route Planning. Authorea Preprints. 2024. DOI: 10.36227/techrxiv.172175516.63690324.

45. Jinadu O. T., Johnson O. V., Ganiyu M. Distributed Database System Optimization for Improved Service Delivery in Mobile and Cloud BigData Applications. International Journal of Computer Science and Mobile Computing. 2021. Vol. 10, No. 9. P. 38–45.

46. Hsu K. Big data analysis and optimization and platform components. Journal of King Saud University-Science. 2022. Vol. 34, No. 4. P. 101945. DOI: 10.1016/j.jksus.2022.101945.

47. Tian T. et al. An Efficient Route Planning Algorithm for Special Vehicles with Large-Scale Road Network Data. ISPRS International Journal of Geo-Information. 2025. Vol. 14, No. 2. P. 71. DOI: 10.3390/ijgi14020071.

48. Stach E. et al. Autonomous experimentation systems for materials development: A community perspective. Matter. 2021. Vol. 4, No. 9. P. 2702–2726. DOI: 10.1016/j.matt.2021.06.036.

49. Genheden S. et al. AiZynthFinder: a fast, robust and flexible open-source software for retrosynthetic planning. Journal of cheminformatics. 2020. Vol. 12, No. 1. P. 70. DOI: 10.1186/s13321-020-00472-1.

50. Liu H. et al. Mapping knowledge structure and research trends of emergency evacuation studies. Safety Science. 2020. Vol. 121. P. 348–361. DOI: 10.1016/j.ssci.2019.09.020.

Розробка веб-платформи для оптимізації маршрутів доставки

Кваліфікаційна робота магістра

Державний університет інформаційно-комунікаційних технологій

2025

Актуальність дослідження

53%

витрат на доставку - логістика останньої милі

+40%

зростання електронної комерції

24/7

очікування швидкої доставки

Проблема: Транспортні компанії потребують ефективних рішень для автоматизованого планування маршрутів.

Об'єкт, предмет та мета

ОБ'ЄКТ ДОСЛІДЖЕННЯ

Процес доставки товарів кінцевим споживачам у рамках логістики останньої милі

ПРЕДМЕТ ДОСЛІДЖЕННЯ

Методи та алгоритми оптимізації маршрутів доставки з урахуванням множини обмежень

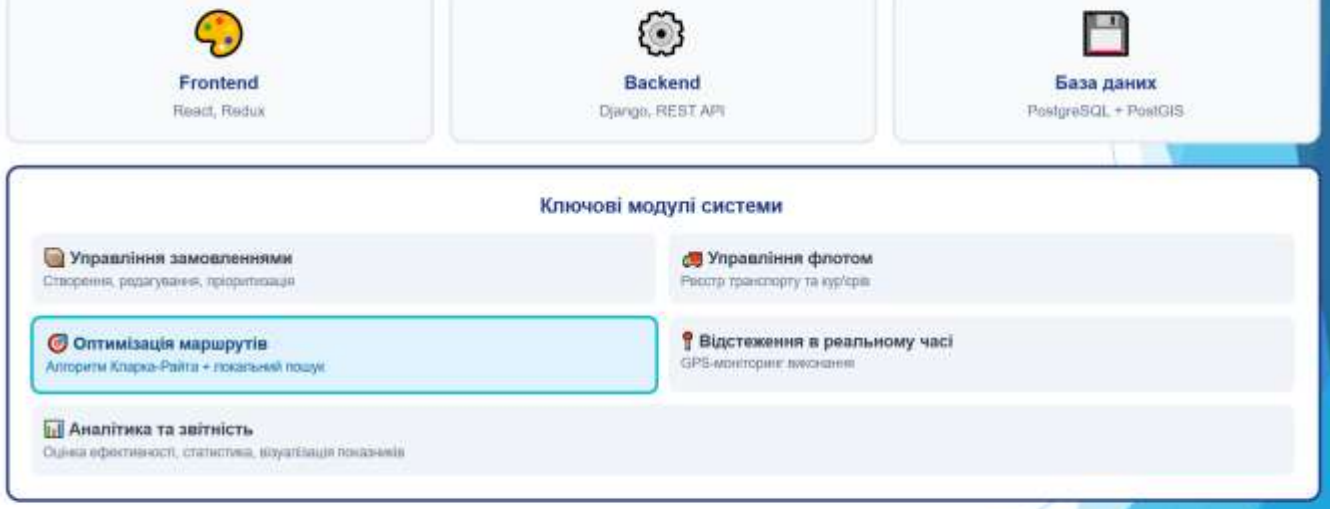
МЕТА РОБОТИ

Розробка комплексної веб-платформи для автоматизованого планування та оптимізації маршрутів доставки

Зниження витрат

Покращення якості

Архітектура платформи



Алгоритм оптимізації

Математична модель

Задача маршрутизації транспортних засобів з часовими вікнами (VRPTW)

Етапи роботи:

1. **Побудова початкових маршрутів**
   Алгоритм Кларка-Райта
2. **Покращення маршрутів**
   Локальний пошук (2-opt)
3. **Валідація та корекція**
   Перевірка обмежень

Враховані обмеження

- Часові вікна доставки
- Вантажопідйомність транспорту
- Пріоритети замовлень
- Робочий час кур'єрів

Критерії оптимізації

min

Загальна відстань

max

Дотримання часових вікон

Технологічний стек

Frontend

React 18
Користувацький інтерфейс

Redux
Управління станом

Leaflet
Картографія

Backend

Django 4.2
Web-фреймворк

Django REST
RESTful API

Celery
Асинхронні завдання

Дані

PostgreSQL
Реляційна БД

PostGIS
Геопросторові дані

Redis
Кешування

Додатково: Docker • Nginx • Git

Результати тестування

-20%
Зниження відстані

+25%
Продуктивність

95%
Дотримання часових вікон

Тестування продуктивності

Обробка замовлень
500 одночасно

Час генерації
30 секунд

✓ **Економічний ефект**

- Зниження витрат на паливо
- Збільшення замовлень
- Підвищення лояльності

Висновки та перспективи

Практична цінність

- ✓ Зниження операційних витрат
- ✓ Покращення якості обслуговування
- ✓ Підвищення ефективності флоту
- ✓ Масштабованість системи

Перспективи розвитку

- ✓ Інтеграція машинного навчання
- ✓ Динамічна оптимізація
- ✓ Підтримка різних типів транспорту
- ✓ Оптимізація розподільчих центрів