

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____ Комп'ютерних наук
(повна назва)
Кафедра _____ Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____ перший (бакалаврський)

Інтелектуальна система багатфакторної оптимізації
топології складського зберігання на основі предикативної аналітики попиту
(тема)

Виконав:
здобувач _____ четвертого _____ року навчання,
групи _____ ІТШ-22-1

_____ Нікіта ОПАРА
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки

(код і повна назва спеціальності)

Тип програми _____ освітньо-професійна

Освітня програма _____ Штучний інтелект

(повна назва освітньої програми)

Керівник _____ ас. Сергій ГРИНЬОВ
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ

(підпис)

_____ Лариса ЧАЛА
(власне ім'я, прізвище)

2026 р.

Харківський національний університет радіоелектроніки

Факультет _____ Комп'ютерних наук _____

Кафедра _____ Штучного інтелекту _____

Рівень вищої освіти _____ перший (бакалаврський) _____

Спеціальність _____ 122 Комп'ютерні науки _____
(код і повна назва)

Тип програми _____ освітньо-професійна _____

Освітня програма _____ Штучний інтелект _____
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____

(підпис)

« _____ » _____ 20 ____ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві _____ Опарі Нікіті Олександровичу _____
(прізвище, ім'я, по батькові)

1. Тема роботи Інтелектуальна система багатофакторної оптимізації топології складського зберігання на основі предикативної аналітики попиту

затверджена наказом університету від 5 червня 2026 р. № 488Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 7 липня 2026 р.

3. Вихідні дані до роботи Науково-технічні публікації з питань складської логістики, оптимізації розміщення товарів, прогнозування попиту та автоматизації складських процесів, дані Інтернет-джерел, документація з Java, Spring Boot, PostgreSQL, Flyway, Tribuo, Redis, Apache Kafka, Elasticsearch, матеріали щодо алгоритмів маршрутизації, машинного навчання та розроблення веб-застосунків.

4. Перелік питань, що потрібно опрацювати в роботі _____

1) Аналіз предметної галузі складської логістики

2) Обґрунтування вибору технологій, методів та моделей

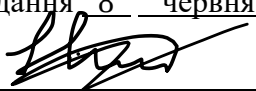
3) Проектування інтелектуальної системи оптимізації розміщення товарів

4) Програмна реалізація та тестування системи

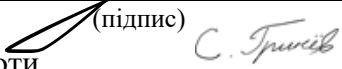
КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / термін виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	08.06.2026	виконано
2	Аналіз предметної галузі складської логістики	09.06.2026	виконано
3	Огляд існуючих методів та програмних засобів	11.06.2026	виконано
4	Проектування інтелектуальної системи оптимізації розміщення товарів	14.06.2026	виконано
5	Програмна реалізація веб-застосунку	21.06.2026	виконано
6	Тестування розробленої системи та аналіз результатів	24.06.2026	виконано
7	Написання пояснювальної записки	26.06.2026	виконано
8	Перевірка на академічний плагіат	27.06.2026	виконано
9	Нормоконтроль	29.06.2026	виконано
10	Підготовка презентації та доповіді	30.06.2026	виконано
11	Попередній захист	01.07.2026	виконано
12	Рецензування	03.07.2026	виконано
13	Захист перед ЕК	07.07.2026	виконано

Дата видачі завдання 8 червня 20 26 р.

Здобувач 

(підпис)

Керівник роботи 

(підпис)

ас. Сергій ГРИНЬОВ

(посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 77 с., 23 рис., 21 табл., 1 дод., 25 джерел.

АЛГОРИТМ ДЕЙКСТРИ, ВЕБЗАСТОСУНОК, МАШИННЕ НАВЧАННЯ, ОПТИМІЗАЦІЯ РОЗМІЩЕННЯ, ПРОГНОЗУВАННЯ ПОПИТУ, СКЛАДСЬКА ЛОГІСТИКА, ШТУЧНИЙ ІНТЕЛЕКТ.

Об'єкт дослідження – процеси розміщення, зберігання, переміщення та оптимізації палетизованих товарів у складських інформаційних системах.

Предмет дослідження – методи та програмні засоби формування рекомендацій щодо розміщення палет і контейнерів на складі з урахуванням структури складу, параметрів місць зберігання, характеристик товарів, маршрутних відстаней, історії попиту та обмежень складської топології.

Мета роботи – розробка інтелектуальної вебсистеми для оптимізації розміщення товарів на складі, яка забезпечує підтримку прийняття рішень під час приймання, розміщення, маршрутизації та планування переміщень.

У роботі використано методи системного аналізу, об'єктно-орієнтованого проєктування, реляційного моделювання даних, багатофакторного оцінювання місць зберігання, графової маршрутизації та машинного навчання для аналізу історії попиту.

У результаті роботи розроблено вебзастосунок, який підтримує створення структури складу, ведення довідника товарів, приймання палет, формування рекомендацій щодо розміщення, прогнозування попиту, розрахунок маршрутів, планування переміщень, журнал операцій та аудит подій. Практична значимість роботи полягає у можливості використання системи як основи для автоматизації складських процесів на складах малого та середнього масштабу.

ABSTRACT

Bachelor's thesis contains: 77 pp., 23 fig., 21 tabl., 1 ann., 25 references.

DEMAND FORECASTING, DIJKSTRA'S ALGORITHM, INTELLIGENT SYSTEM, MACHINE LEARNING, PALLET PLACEMENT, WAREHOUSE LOGISTICS, WAREHOUSE OPTIMIZATION, WEB APPLICATION.

The object of study is the processes of placement, storage, movement, and optimization of palletized goods in warehouse information systems.

The subject of study is the methods and software tools for generating recommendations on container and pallet placement in a warehouse, taking into account storage location parameters, product characteristics, distances, demand history, and warehouse layout constraints.

The aim of the thesis is to develop an intelligent web-based system for optimizing warehouse goods placement. The system supports decision-making during container receiving, placement, routing, and relocation planning.

The research uses system analysis, object-oriented design, relational database modeling, storage location scoring, graph-based routing, and machine learning methods for analyzing demand history and forecasting product demand.

As a result, a web application was developed that supports warehouse layout generation, product directory management, pallet receiving, placement recommendation generation, demand forecasting, route calculation, relocation planning, operation history, and audit events. The practical value of the system lies in improving the organization of warehouse operations and creating a foundation for further development of intelligent warehouse management tools.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів	9
Вступ.....	10
1 Аналіз предметної галузі складської логістики	13
1.1 Складська логістика та процеси складського обліку	13
1.2 Проблема розміщення товарів і прогнозування попиту	14
1.3 Огляд програмних рішень і постановка задачі	15
1.4 Операційний цикл палети та фактори ефективності	16
1.4.1 Фактори, що впливають на якість розміщення товарів	17
1.4.2 Показники ефективності складського розміщення	18
1.5 Узагальнення результатів аналізу предметної галузі	19
2 Обґрунтування вибору технологій, методів та моделей	21
2.1 Загальні вимоги до технологічних рішень	21
2.2 Архітектура вебзастосунку та серверна платформа	21
2.2.1 Java та Spring Boot	21
2.2.2 PostgreSQL, JPA та Flyway	22
2.3 Методи рекомендацій, маршрутизації та прогнозування	23
2.3.1 Алгоритм маршрутизації	23
2.3.2 Прогнозування попиту	23
2.3.3 Обмеження алгоритмічних підходів і розвиток	24
2.4 Інфраструктура, безпека та інтерфейс системи	25
2.4.1 Spring Security та рольова модель	26
2.4.2 Серверний вебінтерфейс	27
2.5 Порівняння алгоритмічних підходів	27
2.6 Обґрунтування прийнятих технологічних та алгоритмічних рішень	28
3 Проєктування інтелектуальної системи оптимізації розміщення товарів	30
3.1 Вимоги до системи та користувацькі сценарії	30
3.2 Архітектура системи та структура бази даних	30
3.3 Модель предметної галузі	31

3.4 Алгоритмічні модулі оптимізації	32
3.4.1 Прогнозування попиту на товари.....	32
3.4.2 Маршрутизація та планування переміщень	32
3.5 Деталізація архітектури та сценарії користувачів	32
3.6 Концептуальна модель даних і алгоритм рекомендації.....	34
3.7 Валідація, журнал операцій та аудит	37
4 Програмна реалізація та тестування системи.....	39
4.1 Структура проєкту та серверна частина	39
4.2 Складські модулі та облік палет.....	40
4.3 Рекомендації щодо розміщення.....	43
4.4 Прогнозування попиту, маршрутизація та оптимізаційні плани	45
4.5 Вебінтерфейс адміністратора й оператора	49
4.6 Тестування системи та аналіз результатів.....	51
4.7 Додаткові реалізовані сценарії	53
4.7.1 Імпорт історії попиту.....	54
4.7.2 Адміністрування користувачів	56
4.7.3 Довідник товарів і коригування даних	57
4.7.4 Операторський пошук палет.....	59
4.8 Події, аудит та API системи	60
4.8.1 API та серверні контролери	61
4.8.2 Перевірка якості сервісів.....	62
4.9 Експлуатаційні обмеження та підготовка до розвитку	64
4.9.1 Міграції бази даних	65
4.9.2 Ручні сценарії перевірки вебінтерфейсу.....	67
4.9.3 Підготовка до інтеграції зі сканером штрихкодів	69
4.9.4 Підсумок програмної реалізації	69
4.9.5 Документаційна сторінка проєкту та API	70
4.9.6 Ризики впровадження та способи їх зменшення	71
Висновки	73
Перелік джерел посилання	74

Додаток А Відомість кваліфікаційної роботи	77
--	----

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БД – база даних;

КР – кваліфікаційна робота;

AI – Artificial Intelligence – штучний інтелект;

API – Application Programming Interface – інтерфейс прикладного програмування;

CART – Classification and Regression Tree – алгоритм дерева класифікації та регресії;

CRUD – Create, Read, Update, Delete – базові операції створення, читання, оновлення та видалення даних;

DTO – Data Transfer Object – об’єкт передавання даних;

ERP – Enterprise Resource Planning – система планування ресурсів підприємства;

JPA – Jakarta Persistence API – специфікація для об’єктно-реляційного відображення даних у Java;

ML – Machine Learning – машинне навчання;

REST – Representational State Transfer – архітектурний стиль взаємодії вебсервісів;

SKU – Stock Keeping Unit – складська одиниця обліку товару;

SQL – Structured Query Language – мова структурованих запитів;

UI – User Interface – користувацький інтерфейс;

WMS – Warehouse Management System – система управління складом.

ВСТУП

Сучасні складські системи працюють в умовах постійного зростання обсягів товарообігу, підвищення вимог до швидкості обробки замовлень і необхідності раціонального використання складського простору. Для підприємств торгівлі, виробництва та логістики склад уже не є лише місцем фізичного зберігання товарів. Він перетворюється на важливий елемент інформаційної інфраструктури, від ефективності якого залежить швидкість виконання замовлень, рівень витрат і стабільність операційних процесів [1], [2], [3].

Однією з ключових проблем складської логістики є вибір місця для розміщення товарів після їх приймання. На практиці таке рішення часто приймається оператором вручну, з урахуванням доступності місць, габаритів товару, досвіду працівника або поточної ситуації на складі. Такий підхід є зрозумілим для невеликих складів, але зі збільшенням кількості палет, артикулів і місць зберігання він стає менш ефективним.

Актуальність теми роботи зумовлена необхідністю автоматизації процесів прийняття рішень у складських системах. Для підвищення ефективності роботи складу недостатньо лише зберігати інформацію про товари та місця зберігання. Система повинна аналізувати параметри складу, характеристики товарів, історію попиту, маршрутні відстані та можливість подальших переміщень.

У межах кваліфікаційної роботи розглядається задача розробки інтелектуальної вебсистеми багатофакторної оптимізації топології складського зберігання на основі предиктивної аналітики попиту. Основна ідея полягає в тому, що розміщення товарів має визначатися не лише фактом наявності вільного місця, а й сукупністю факторів: відстанню до місця зберігання, фізичними обмеженнями комірки, характеристиками палети, можливістю об'єднання однакових товарів і прогнозованою популярністю товару.

Метою кваліфікаційної роботи є розробка інтелектуальної вебсистеми оптимізації розміщення товарів на складі з підтримкою прогнозування попиту, формування рекомендацій щодо розміщення контейнерів, маршрутизації та планування переміщень.

Для досягнення поставленої мети необхідно проаналізувати предметну галузь складської логістики, дослідити підходи до автоматизації складських процесів, обґрунтувати вибір технологій, методів і моделей, спроектувати архітектуру вебзастосунку, реалізувати основні програмні модулі та провести тестування системи.

Об'єктом дослідження є процеси розміщення, зберігання, переміщення та оптимізації палетизованих товарів у складських інформаційних системах. Предметом дослідження є методи та програмні засоби формування рекомендацій щодо розміщення контейнерів і палет з урахуванням структури складу, параметрів місць зберігання, характеристик товарів, історії попиту та маршрутних відстаней.

Практична цінність роботи полягає у створенні програмного рішення, яке може бути використане як основа для автоматизації складських процесів на складах малого та середнього масштабу. У поточній версії система перевіряє бізнес-логіку без фізичного сканера штрихкодів: сценарій сканування імітується ручним введенням номерів палет і кодів місць зберігання. У перспективі ці поля можуть бути підключені до сканера, що працює як пристрій введення.

Під час розроблення такої системи важливо враховувати, що складська логістика поєднує фізичні операції та інформаційні процеси. Одна й та сама дія оператора має матеріальний результат у вигляді переміщення палети та цифровий результат у вигляді зміни статусу, оновлення місця зберігання, створення запису руху й фіксації відповідної події в системі. Саме тому програмне рішення повинно не лише показувати довідники, а й підтримувати повний операційний цикл.

Робота орієнтована на сценарій складу малого або середнього масштабу, де складська топологія може бути формалізована у вигляді проходів, рядів, рівнів і місць зберігання. Для такого класу складів особливо важливими є простота впровадження, зрозумілий вебінтерфейс, можливість поступового нарощування функціональності та відсутність залежності від дорогого промислового обладнання на ранньому етапі.

У поточній реалізації апаратне сканування штрихкодів не використовується, оскільки фізичний сканер не був доступний під час розроблення. Однак інтерфейс побудовано так, щоб поля введення номера палети та коду місця зберігання могли надалі приймати дані від сканера, який працює як клавіатурний пристрій введення. Це дозволяє розглядати ручне введення як тимчасову умовність, а не як принципове обмеження архітектури.

Таким чином, у роботі розглядається не абстрактна задача оптимізації, а приклад прикладної інформаційної системи, у якій поєднуються довідники, бізнес-правила, історія операцій, алгоритми маршрутизації, прогнозування попиту та вебінтерфейс для двох ролей користувачів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ СКЛАДСЬКОЇ ЛОГІСТИКИ

1.1 Складська логістика та процеси складського обліку

Складська логістика є важливою складовою логістичної системи підприємства, оскільки забезпечує приймання, зберігання, переміщення, облік і подальше відвантаження товарів. Від ефективності складських процесів залежить швидкість виконання замовлень, рівень операційних витрат, якість обслуговування клієнтів і загальна продуктивність підприємства [4], [5].

У сучасних умовах склад виконує не лише функцію фізичного зберігання продукції. Він є інформаційним середовищем, у якому постійно змінюються дані про товари, залишки, місця зберігання, переміщення, замовлення та попит. Через це ручне управління складом поступово втрачає ефективність.

Автоматизація складської логістики дає змогу підвищити точність обліку, зменшити кількість помилок, прискорити виконання типових операцій і забезпечити прозорість переміщення товарів.

До основних процесів складської логістики належать приймання товарів, перевірка їх характеристик, розміщення у місцях зберігання, внутрішні переміщення, контроль залишків, інвентаризація та підготовка товарів до подальшого використання або відвантаження.

Після надходження товару на склад необхідно зафіксувати його основні параметри: артикул, назву, кількість, вагу, габарити, висоту палети або контейнера. Ці дані використовуються для перевірки можливості розміщення товару у конкретному місці зберігання.

Процес розміщення товару не повинен зводитися лише до пошуку першої вільної комірки. Раціональне рішення має враховувати фізичні обмеження місця зберігання, характеристики товару, можливість

об'єднання однакових товарів, відстань до комірки та потенційну частоту звернення до товару в майбутньому [6], [7], [8].

1.2 Проблема розміщення товарів і прогнозування попиту

Однією з основних проблем складської логістики є нераціональне розміщення товарів. Вона виникає тоді, коли під час вибору місця зберігання враховується лише поточна доступність комірки, але не беруться до уваги відстань, популярність товару, габаритні обмеження, вантажопідйомність, можливість об'єднання однакових товарів і майбутні переміщення.

На практиці це може призводити до ситуації, коли товари з високим попитом розміщуються у віддалених частинах складу, а менш затребувані товари займають зручніші місця. Унаслідок цього оператор витрачає більше часу на переміщення, збільшується навантаження на персонал і знижується загальна продуктивність складу.

Отже, задача розміщення товарів є багатофакторною. Для її розв'язання доцільно використовувати систему, яка аналізує параметри товару, стан складу, характеристики місць зберігання та історичні дані про попит.

Прогнозування попиту є одним із перспективних напрямів підвищення ефективності складських процесів [9], [10]. Якщо система має дані про те, які товари замовляються частіше, вона може враховувати цю інформацію під час розміщення товарів на складі.

Для прогнозування попиту необхідно накопичувати дані про замовлення, артикул товару, кількість замовлених одиниць і дату операції. На основі цих даних можна визначати популярність товарів, виявляти зміну попиту в часі та формувати прогнозні оцінки.

Використання прогнозування попиту не замінює класичні складські правила, а доповнює їх. Система все одно повинна перевіряти вагові та габаритні обмеження, доступність місць і коректність операцій.

1.3 Огляд програмних рішень і постановка задачі

Для автоматизації складських процесів використовуються системи управління складом, або WMS. Такі системи зазвичай підтримують приймання товарів, адресне зберігання, контроль залишків, інвентаризацію, переміщення, відбір замовлень і формування звітності.

Перевагою готових WMS-рішень є широкий функціонал і можливість використання в промислових умовах. Водночас такі системи часто є складними для впровадження, потребують налаштування під конкретне підприємство та можуть бути надлишковими для складів малого або середнього масштабу.

Порівняння типових WMS-рішень і розроблюваної системи наведено в таблиці 1.1.

Таблиця 1.1 – Порівняння типових WMS-рішень і розроблюваної системи

Критерій	Типові WMS-рішення	Розроблювана система
Облік товарів	Підтримується	Підтримується
Адресне зберігання	Підтримується	Підтримується
Рекомендації розміщення	Зазвичай базуються на правилах	Формуються з урахуванням кількох факторів
Прогнозування попиту	Доступне не в усіх системах	Передбачено як окремий напрям оптимізації
Маршрутизація	Залежить від конкретної платформи	Реалізується на основі графової моделі складу
Гнучкість алгоритмів	Обмежена готовою системою	Можлива зміна та розвиток алгоритмів

Таким чином, розробка власної системи є обґрунтованою, оскільки вона дозволяє поєднати базові функції складського обліку з інтелектуальними методами підтримки прийняття рішень.

Метою кваліфікаційної роботи є розробка інтелектуальної вебсистеми багатофакторної оптимізації топології складського зберігання на основі предиктивної аналітики попиту.

Для досягнення поставленої мети необхідно розв'язати такі задачі: проаналізувати предметну галузь складської логістики; обґрунтувати вибір технологій, методів і моделей; спроектувати модель складу, товарів, контейнерів і місць зберігання; реалізувати механізм формування рекомендацій; додати аналіз попиту, маршрутизацію та планування переміщень; розробити вебінтерфейс для адміністратора й оператора; провести тестування системи.

1.4 Операційний цикл палети та фактори ефективності

Для коректного проектування системи важливо описати життєвий цикл палети як основної одиниці складського зберігання. Палета потрапляє до системи після приймання, отримує власний номер, пов'язується з артикулом товару та переходить у стан очікування розміщення. На цьому етапі вона ще фізично не закріплена за конкретним місцем, тому система повинна запропонувати оператору подальшу дію.

Після отримання рекомендації оператор може розмістити палету у вільному місці або об'єднати її з уже наявною палетою такого самого товару. У разі підтвердження операції змінюються статус палети, статус місця зберігання, а також створюється запис руху. Такий підхід дозволяє відновити історію операцій і зрозуміти, хто, коли та з якої причини виконав переміщення.

Окремим етапом життєвого циклу є оптимізаційне переміщення. Воно відрізняється від первинного розміщення тим, що палета вже перебуває на

складі, але система визначає можливість покращити її позицію. Наприклад, товар із високим прогнозованим попитом може бути переміщений ближче до входу або до зручнішого місця виконання операцій.

З точки зору інформаційної системи кожна зміна стану повинна бути атомарною: не можна допустити ситуацію, коли палета отримала нове місце, але місце зберігання не було оновлене, або запис руху не потрапив до історії. Саме тому складські операції доцільно реалізовувати через сервісний рівень із транзакційними межами.

Типові стани палети в системі наведено в таблиці 1.2.

Таблиця 1.2 – Типові стани палети в системі

Стан	Зміст стану	Типова дія оператора
WAITING_FOR_PLACEMENT	Палета прийнята, але ще не розміщена	Отримати рекомендацію та підтвердити місце
STORED	Палета фізично закріплена за місцем зберігання	Переглянути, перемістити або використати в оптимізації
REMOVED	Палета виведена з активного обліку	Перевірити причину списання або видалення
MERGED	Палета об'єднана з іншою палетою такого самого товару	Перевірити цільову палету та історію операції

Наведені стани не є лише відображенням інтерфейсу. Вони задають правила допустимих переходів, впливають на доступність місць зберігання та забезпечують контроль коректності операцій.

1.4.1 Фактори, що впливають на якість розміщення товарів

Якість розміщення товарів визначається не одним показником, а сукупністю взаємопов'язаних факторів. Найочевиднішим є відстань від

точки входу або основного робочого маршруту до місця зберігання. Чим частіше товар використовується, тим сильніше відстань впливає на сумарний час роботи оператора.

Другим фактором є фізична сумісність товару та місця. Місце зберігання має обмеження за максимальною вагою та висотою, а палета має власні параметри. Якщо система не перевіряє ці обмеження, оператор може отримати рекомендацію, яку неможливо виконати фізично або яка створює ризик порушення правил зберігання.

Третім фактором є фрагментація запасів. Якщо однаковий товар розміщений у багатьох різних місцях, ускладнюється контроль залишків і збільшується кількість переміщень. Тому в системі доцільно спочатку перевіряти можливість об'єднання палет, а вже потім шукати нове місце.

Четвертим фактором є зміна попиту в часі. Товар, який був малопопулярним у минулому місяці, може стати важливим у наступному періоді через сезонність або зміну структури замовлень. Через це система повинна використовувати не лише поточний залишок, а й історію попиту.

Нарешті, важливим є фактор надійності даних. Якщо оператор неправильно ввів вагу, висоту або номер товару, навіть якісний алгоритм сформує некоректну рекомендацію. Тому в інтерфейсі мають бути передбачені перевірки, довідники та можливість коригування даних адміністратором.

1.4.2 Показники ефективності складського розміщення

Для оцінювання результатів роботи системи необхідно визначити показники, за якими можна порівнювати різні варіанти розміщення. У межах роботи основну увагу приділено показникам, які можна обчислити на основі доступних даних без підключення зовнішньої ERP-системи.

Одним із ключових показників є середня відстань від входу до місць зберігання палет, для яких є попит. Якщо товари з високою частотою

звернень розміщені ближче, оператор витрачає менше часу на типові операції. Цей показник добре підходить для демонстрації ефекту оптимізаційних планів.

Іншим показником є частка палет, для яких система має дані про попит. Якщо історія замовлень недостатня, прогнозна модель не може повноцінно впливати на рекомендації. Тому якість попиту є не менш важливою, ніж якість алгоритму.

Також доцільно враховувати кількість операцій об'єднання, кількість переміщень, частку відхилених рекомендацій і кількість помилкових або непідтверджених дій. Ці показники дають змогу оцінити не лише математичну ефективність, а й зручність роботи користувачів із системою.

Показники оцінювання ефективності складського розміщення наведено в таблиці 1.3.

Таблиця 1.3 – Показники оцінювання ефективності складського розміщення

Показник	Що характеризує	Використання в роботі
Середня відстань	Середню довжину маршруту до палет із попитом	Оцінювання поточної ефективності складу
Частка палет із попитом	Наявність історичних даних для прийняття рішень	Перевірка готовності до прогнозування
Кількість переміщень	Обсяг роботи, потрібний для покращення топології	Формування оптимізаційного плану
Частка об'єднань	Рівень зменшення фрагментації однакових товарів	Оцінювання ефекту merge-first підходу

1.5 Узагальнення результатів аналізу предметної галузі

Аналіз предметної галузі показав, що складська логістика є не лише задачею фізичного переміщення товарів, а й задачею управління даними.

Кожна операція повинна бути зафіксована, перевірена, пов'язана з користувачем і відображена в поточному стані складу.

Найбільш суттєвою проблемою є вибір місця зберігання в умовах багатьох обмежень. Рішення має враховувати не тільки доступність комірки, а й габарити, вагу, відстань, можливість об'єднання та прогнозований попит. Через це задача має багатофакторний характер.

Отже, для розв'язання поставленої задачі доцільно розробити вебсистему, яка поєднує функції WMS, алгоритмічну підтримку рекомендацій, прогнозування попиту та засоби аудиту. Така система може бути використана як основа для подальшої інтеграції з апаратним сканером і зовнішніми корпоративними системами.

Результати аналізу також показують, що ефективність складської системи залежить від узгодженості між фізичною структурою складу та її цифровим відображенням. Якщо інформація про місця зберігання, палети, товари, обмеження ваги й висоти або історію попиту є неповною, система не зможе формувати достатньо обґрунтовані рекомендації. Тому під час подальшого проєктування необхідно передбачити не лише алгоритм вибору місця, а й механізми перевірки даних, фіксації операцій, аудиту змін і можливості поступового розширення функціональності системи.

2 ОБҐРУНТУВАННЯ ВИБОРУ ТЕХНОЛОГІЙ, МЕТОДІВ ТА МОДЕЛЕЙ

2.1 Загальні вимоги до технологічних рішень

Під час розробки інтелектуальної системи оптимізації складського зберігання необхідно враховувати вимоги до надійності, масштабованості, підтримуваності та можливості подальшого розвитку. Система повинна зберігати структуровані дані про склад, товари, палети, користувачів, переміщення та результати оптимізації.

Вибір технологій здійснювався з урахуванням стабільності платформи, підтримки реляційної моделі даних, наявності засобів для побудови вебзастосунків, інтеграції з інструментами машинного навчання та можливості розширення системи.

2.2 Архітектура вебзастосунку та серверна платформа

Для реалізації системи обрано веб-орієнтовану архітектуру. Такий підхід є доцільним, оскільки користувачі можуть працювати з системою через браузер без встановлення окремого клієнтського програмного забезпечення.

У системі виділено дві основні ролі: адміністратор складу та оператор складу. Адміністратор керує структурою складу, товарами, користувачами, попитом і оптимізаційними планами, а оператор виконує приймання, розміщення та переміщення палет.

2.2.1 Java та Spring Boot

Для реалізації серверної частини обрано Java, оскільки ця мова широко використовується для створення корпоративних інформаційних

систем, має розвинену екосистему бібліотек і добре підходить для застосунків зі складною бізнес-логікою [11].

Фреймворк Spring Boot обрано як основу вебзастосунку, оскільки він надає готові інструменти для створення контролерів, роботи з базою даних, авторизації користувачів, валідації даних, конфігурування та тестування [12].

2.2.2 PostgreSQL, JPA та Flyway

Для зберігання основних даних системи обрано PostgreSQL. Предметна область складської логістики має чітко структуровану природу: склади, місця зберігання, товари, палети, користувачі, рекомендації та плани оптимізації мають визначені зв'язки між собою [13].

Spring Data JPA та Hibernate використовуються для роботи з об'єктно-реляційним відображенням, а Flyway забезпечує контрольовану еволюцію структури бази даних через міграції [14], [15].

Обґрунтування вибору основних технологій наведено в таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору основних технологій

Технологія	Призначення	Причина вибору
Java	Серверна логіка	Стабільність, типізація, придатність для корпоративних систем
Spring Boot	Основа вебзастосунку	Швидка розробка, модульність, інтеграція компонентів
PostgreSQL	Основна база даних	Надійне зберігання структурованих даних
Spring Data JPA	Робота з даними	Спрощення доступу до реляційної БД
Flyway	Міграції БД	Контрольована зміна структури бази даних
Tribuo	Машинне навчання	Інтеграція прогнозування попиту в Java-екосистемі

2.3 Методи рекомендацій, маршрутизації та прогнозування

Основною задачею системи є формування рекомендацій щодо розміщення палет. Для цього використано багатофакторний підхід, оскільки вибір місця залежить від доступності комірки, вагових і габаритних обмежень, відстані до місця, стану складу та можливості об'єднання палет з однаковим товаром.

Окремо враховано принцип пріоритету об'єднання. Якщо на складі вже є палета з таким самим артикулом і об'єднання не порушує обмежень місця зберігання, така дія зменшує фрагментацію запасів і спрощує подальший облік.

2.3.1 Алгоритм маршрутизації

Складську топологію можна подати у вигляді графа, де вершинами є логічні точки складу або місця зберігання, а ребрами – можливі переходи між ними. Для пошуку найкоротшого маршруту обрано алгоритм Дейкстри, оскільки відстані між точками складу є невід'ємними [16], [17].

2.3.2 Прогнозування попиту

Для підвищення ефективності розміщення товарів доцільно враховувати історію попиту. Якщо система знає, які товари замовляються частіше, вона може використовувати цю інформацію під час формування рекомендацій щодо розміщення. Товари з вищою очікуваною частотою звернення доцільно розміщувати ближче до основних маршрутів або до точки входу, оскільки це зменшує сумарну відстань переміщення оператора.

У межах роботи прогнозування попиту розглядається як допоміжний інструмент підтримки прийняття рішень, а не як єдиний критерій оптимізації. Система повинна спочатку перевіряти обов'язкові обмеження:

доступність місця, допустиму вагу, висоту, відповідність складу та можливість виконання операції. Лише після цього прогнозований попит може впливати на пріоритетність місць зберігання або доцільність подальшого переміщення палети.

Джерелом даних для прогнозування є історія замовлень, яка містить дату операції, артикул товару та кількість замовлених одиниць. На основі цих даних можна визначити частоту звернення до кожного товару, оцінити його популярність і сформувати навчальний набір для моделі машинного навчання [9], [10].

Для реалізації машинного навчання в системі обрано Tribuo. Ця бібліотека дозволяє працювати з моделями без виходу за межі Java-екосистеми, що спрощує інтеграцію прогнозного модуля із серверною частиною застосунку. Завдяки цьому навчання моделі, збереження результатів і подальше використання прогнозу можуть виконуватися в межах однієї технологічної платформи [18], [19].

Важливо, що результат прогнозування не замінює бізнес-правила складської системи. Навіть якщо товар має високий прогнозований попит, він не може бути розміщений у комірці, яка не відповідає його вазі, висоті або поточному стану. Тому прогноз попиту використовується як додатковий фактор у багатофакторній оцінці, що поєднує фізичні обмеження, маршрутні відстані та очікувану частоту звернення до товару.

2.3.3 Обмеження алгоритмічних підходів і розвиток

Обрані технології та методи відповідають задачам кваліфікаційної роботи, однак мають певні обмеження. Система передбачає, що структура складу формалізована, а місця зберігання мають відомі параметри. Якість рекомендацій залежить від коректності даних про товари, палети, вагу, висоту та доступність місць.

У поточній версії апаратну інтеграцію зі сканером штрихкодів не реалізовано через відсутність фізичного обладнання. Сценарій сканування імітується ручним введенням номерів палет і кодів місць. У перспективі ці поля можуть працювати зі сканером, який передає відсканований код як клавіатурне введення.

2.4 Інфраструктура, безпека та інтерфейс системи

Окрім основної серверної частини та бази даних, система використовує інфраструктурні компоненти, які забезпечують надійність і спостережуваність. До них належать Redis, Apache Kafka, Elasticsearch, Mailpit і засоби моніторингу на основі Micrometer та Prometheus.

Redis використовується як опційний компонент для кешування та розподілених блокувань. У складських системах важливо уникати одночасного виконання однакових фонових задач, наприклад навчання моделі або оцінювання оптимізації. Розподілене блокування дозволяє запобігти дублюванню таких операцій у разі запуску кількох екземплярів застосунку [20].

Apache Kafka використовується для подій, які виникають після складських операцій. У системі реалізовано transactional outbox: спочатку подія записується в PostgreSQL у межах транзакції основної операції, а потім окремий диспетчер може доставити її до Kafka. Такий підхід зменшує ризик втрати подій [21].

Elasticsearch застосовується для індексації та пошуку аудиторських подій. Оскільки події мають JSON-навантаження та можуть містити різні типи операцій, пошуковий індекс зручніший для перегляду аудиту, ніж складні запити до кількох реляційних таблиць [22].

Mailpit використовується як локальний інструмент для перевірки e-mail-повідомлень. У реальному середовищі такий компонент може бути

замінений на корпоративний SMTP-сервер, але під час розроблення він дозволяє тестувати повідомлення без зовнішньої поштової інфраструктури.

Узагальнене призначення використаних інфраструктурних компонентів наведено в таблиці 2.2.

Таблиця 2.2 – Призначення інфраструктурних компонентів

Компонент	Роль у системі	Причина вибору
Redis	Кешування та розподілені блокування	Швидка робота з ключами, просте підключення через Spring
Kafka	Доставка доменних подій	Підтримка асинхронної обробки та масштабування
Elasticsearch	Індексація та пошук аудиту	Зручний пошук за подіями та JSON-навантаженням
Micrometer / prometheus	Метрики та спостережуваність	Стандартний підхід для spring boot застосунків
Mailpit	Локальна перевірка E-mail	Можливість тестування без зовнішньої пошти

2.4.1 Spring Security та рольова модель

Складська система працює з операціями, які мають різний рівень відповідальності. Наприклад, оператор може приймати та розміщувати палети, але не повинен створювати облікові записи адміністраторів або змінювати критичні параметри складу. Через це в системі використано рольову модель доступу.

Spring Security обрано тому, що він є стандартним інструментом безпеки для застосунків Spring. Він дозволяє централізовано описувати правила доступу, інтегрувати автентифікацію з користувацькою моделлю, обмежувати маршрути вебінтерфейсу та захищати API-контролери [23].

У системі виділено ролі ROOT_ADMIN, ADMIN та OPERATOR. ROOT_ADMIN має найвищі права та використовується для початкового

адміністрування. ADMIN відповідає за управління складом, товарами, користувачами, моделями попиту та оптимізацією. OPERATOR працює з практичними операціями складу.

Такий розподіл ролей відповідає реальному поділу відповідальності на складі. Він зменшує ризик випадкових змін у довідниках, обмежує доступ до адміністративних функцій і робить інтерфейс зрозумілішим для кожної категорії користувачів.

2.4.2 Серверний вебінтерфейс

Для інтерфейсу обрано підхід із серверним рендерингом сторінок за допомогою Thymeleaf. Такий варіант є доцільним для кваліфікаційної роботи, оскільки дозволяє зосередитися на бізнес-логіці складської системи без необхідності розробляти окремий клієнтський застосунок.

Серверний вебінтерфейс добре підходить для адміністративних панелей і операторських робочих місць, де основними діями є заповнення форм, перегляд таблиць, підтвердження операцій і відкриття сторінок із результатами. Складна клієнтська SPA-архітектура у такому сценарії не потрібна.

Водночас застосунок зберігає чіткий поділ між контролерами, сервісами та репозиторіями. Це означає, що в майбутньому до тієї самої бізнес-логіки можна додати REST-клієнт, мобільний застосунок або інтеграцію з зовнішньою WMS без повного переписування серверної частини.

2.5 Порівняння алгоритмічних підходів

Під час вибору алгоритмічного підходу розглядалися прості правила розміщення, евристичне багатофакторне оцінювання та складніші оптимізаційні методи. Для поточної роботи обрано компромісний підхід:

система використовує чіткі обмеження, пріоритет об'єднання, відстань і дані попиту.

Повна математична оптимізація всього складу потребує великої кількості точних даних: маршрутів, часу операцій, черг завдань, правил безпеки, обмежень проходів і плану відвантажень. У межах бакалаврської роботи доцільніше реалізувати контрольований алгоритм, який легко пояснити, протестувати та розширити.

Алгоритм Дейкстри обрано для маршрутизації, оскільки граф складу має невід'ємні ваги ребер. Для попиту використано модель машинного навчання, але результат прогнозування не замінює бізнес-правила, а лише впливає на пріоритетність місць.

Порівняння розглянутих алгоритмічних підходів до розміщення товарів наведено в таблиці 2.3.

Таблиця 2.3 – Порівняння алгоритмічних підходів до розміщення

Підхід	Переваги	Обмеження	Доцільність для роботи
Перша вільна комірка	Простота реалізації	Не враховує відстань і попит	Недостатня
Правила + відстань	Зрозуміла логіка, легко тестувати	Обмежене врахування майбутнього попиту	Базово придатна
Багатофакторна рекомендація	Поеднує сумісність, відстань, об'єднання та попит	Залежить від якості даних	Обрана в роботі
Глобальна оптимізація	Може давати найкращий загальний план	Складна, потребує більше даних	Перспектива розвитку

2.6 Обґрунтування прийнятих технологічних та алгоритмічних рішень

Обраний технологічний стек відповідає задачі створення інтелектуальної складської вебсистеми. Java та Spring Boot забезпечують

надійну основу для бізнес-логіки, PostgreSQL і Flyway підтримують контрольоване зберігання даних, а Tribuo дозволяє реалізувати машинне навчання в межах Java-екосистеми.

Інфраструктурні компоненти Kafka, Redis та Elasticsearch не є обов'язковими для найпростішої роботи системи, однак вони демонструють можливість подальшого розвитку проєкту в напрямі промислової експлуатації, аудиту, масштабування та моніторингу.

Вибір методів також є поступовим: спочатку система перевіряє жорсткі обмеження, потім застосовує рекомендаційну логіку, а вже після цього використовує прогнозування та оптимізаційні плани. Такий підхід робить систему зрозумілою для користувача й придатною для тестування.

Важливою перевагою обраних рішень є їхня узгодженість між собою. Серверна частина, база даних, модулі прогнозування, маршрутизації та аудиту працюють у межах єдиної архітектури, тому результати одних компонентів можуть безпосередньо використовуватися іншими. Наприклад, дані про попит впливають на рекомендації щодо розміщення, маршрутні відстані використовуються під час оцінювання місць зберігання, а події та журнал операцій дозволяють відстежувати наслідки виконаних рішень. Такий підхід забезпечує цілісність системи та створює основу для її подальшого розвитку.

3 ПРОЄКТУВАННЯ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ ОПТИМІЗАЦІЇ РОЗМІЩЕННЯ ТОВАРІВ

3.1 Вимоги до системи та користувацькі сценарії

Розроблювана система призначена для підтримки складських процесів, пов'язаних із прийманням, розміщенням, переміщенням і оптимізацією зберігання палетизованих товарів. Основною задачею є формування обґрунтованих рекомендацій щодо вибору місця зберігання.

Основні функціональні вимоги до системи наведено в таблиці 3.1.

Таблиця 3.1 – Функціональні вимоги до розроблюваної системи

Вимога	Опис
Керування складом	Створення структури складу, рядів, рівнів і місць зберігання
Керування товарами	Створення та редагування артикулів, габаритів, ваги й одиниць виміру
Приймання палет	Реєстрація палет із зазначенням товару, кількості, ваги та висоти
Рекомендації розміщення	Пошук доцільного місця або варіанта об'єднання
Прогнозування попиту	Аналіз історії замовлень і визначення популярності товарів
Маршрутизація	Розрахунок відстаней до місць зберігання
Аудит операцій	Збереження історії переміщень і подій системи

3.2 Архітектура системи та структура бази даних

Архітектура системи побудована за багаторівневим принципом. На рівні представлення користувач працює з вебінтерфейсом адміністратора або оператора. Запити надходять до контролерів, які передають їх до

сервісного рівня. Сервіси реалізують бізнес-логіку та взаємодіють з базою даних і алгоритмічними модулями.

База даних повинна зберігати інформацію про користувачів, склади, місця зберігання, товари, палети, рекомендації, переміщення, історію попиту, прогнозні моделі та оптимізаційні плани. Оскільки ці дані мають чіткі зв'язки, для їх зберігання використано реляційну модель.

Основні сутності бази даних наведено в таблиці 3.2.

Таблиця 3.2 – Основні сутності бази даних

Сутність	Призначення
User	зберігання облікових записів і ролей користувачів
Warehouse	опис складу та його загальних параметрів
StoragePlace	опис конкретного місця зберігання
Article	зберігання інформації про товарну позицію
Container	зберігання інформації про фізичну одиницю товару
PlacementRecommendation	збереження рекомендацій щодо розміщення
ContainerMovement	фіксація історії переміщень палет
OrderDemand	зберігання історії попиту за замовленнями
WarehouseOptimizationPlan	зберігання планів оптимізації та переміщень

3.3 Модель предметної галузі

Модель складу описує фізичну структуру зберігання. Склад складається з проходів, рядів, секцій, рівнів і місць зберігання. Кожне місце має унікальний код, параметри допустимої ваги та висоти, відстань від точки входу і поточний стан.

Товар описується артикулом, назвою, одиницею виміру, габаритами, вагою та максимальною кількістю на палеті. Палета є фізичною одиницею зберігання товару та містить інформацію про артикул, кількість, вагу, висоту, стан і місце розміщення.

3.4 Алгоритмічні модулі оптимізації

Алгоритм формування рекомендацій визначає найдоцільнішу дію для палети, яка очікує розміщення. Система спочатку перевіряє можливість об'єднання з уже наявною палетою такого самого товару. Якщо відповідного варіанта немає, аналізуються доступні місця зберігання.

Для кожного місця перевіряється стан, вантажопідйомність, допустима висота й інші обмеження. Місця, які не відповідають вимогам, відкидаються. Для решти варіантів розраховується оцінка, що враховує відстань і орієнтовний час переміщення.

3.4.1 Прогнозування попиту на товари

Прогнозування попиту використовується для підвищення якості рішень щодо розміщення товарів. Джерелом даних є історія замовлень, яка містить дату, артикул товару та кількість. На основі цих даних формується навчальний набір для моделі машинного навчання.

3.4.2 Маршрутизація та планування переміщень

Для оцінювання зручності місць зберігання складську топологію подано у вигляді графа. Вершинами графа є логічні точки складу або місця зберігання, а ребра описують можливі переходи між ними. Планування переміщень використовується тоді, коли система визначає можливість покращення поточного стану складу.

3.5 Деталізація архітектури та сценарії користувачів

Загальну архітектуру системи доцільно розглядати як набір пов'язаних рівнів. На верхньому рівні знаходиться користувач, який працює

через вебінтерфейс. Вебінтерфейс передає запити до Spring MVC-контролерів, а контролери делегують виконання бізнес-операцій сервісам.

Сервісний рівень є центральним у системі. Саме тут перевіряються права користувача, валідність даних, доступність складу, сумісність палети та місця зберігання, можливість об'єднання і правила оптимізаційних планів.

Окремі алгоритмічні модулі не ізольовані від предметної області. Модуль маршрутизації використовує структуру складу, модуль прогнозування спирається на історію замовлень, а модуль оптимізації аналізує розміщення палет і попит. Завдяки цьому система не просто зберігає дані, а формує прикладні рекомендації.

Узагальнену архітектуру вебсистеми оптимізації складського зберігання наведено на рисунку 3.1.

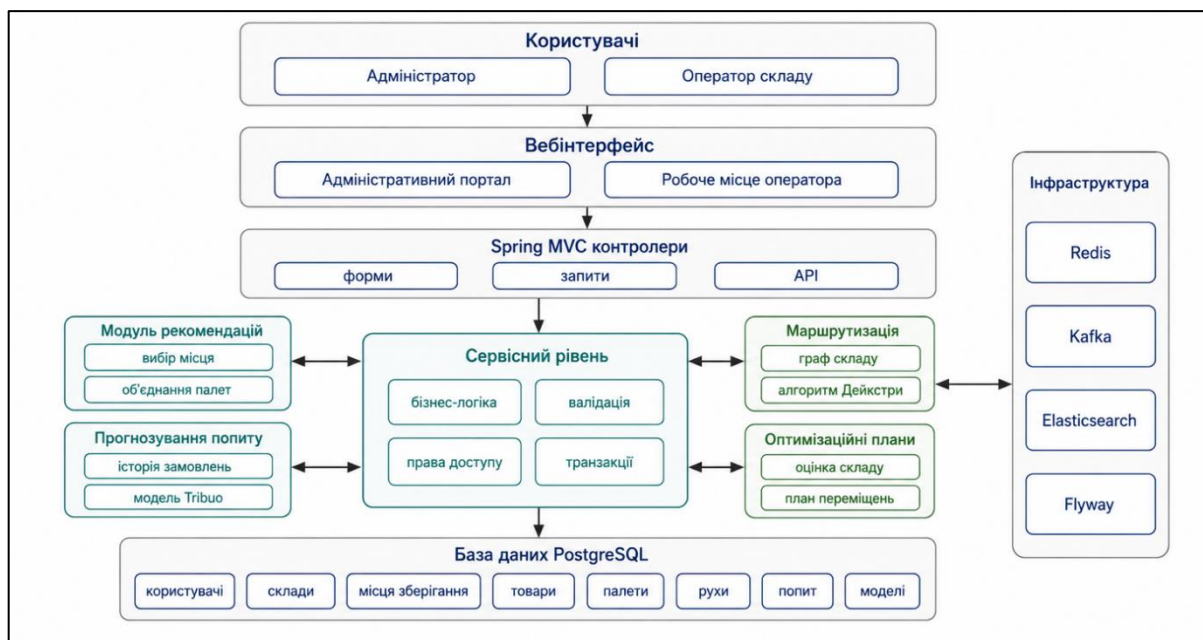


Рисунок 3.1 – Узагальнена архітектура вебсистеми оптимізації складського зберігання

Для системи важливо відокремити адміністративні та операторські сценарії. Адміністратор працює з довідниками, структурою складу,

користувачами, моделями попиту та планами оптимізації. Оператор виконує фізично наближені операції: приймання палети, розміщення, підтвердження переміщення та перевірку запасів.

Сценарій приймання починається з введення номера палети, артикула, кількості, ваги та висоти. Після збереження палета отримує статус очікування розміщення. Далі оператор переходить до сценарію розміщення, де система надає рекомендацію.

Сценарій виконання оптимізаційного плану відрізняється тим, що оператор не обирає довільну дію. Він відкриває затверджений план і виконує поточний крок. У реальному середовищі підтвердження має виконуватися скануванням номера палети та коду цільового місця.

Основні сценарії користувачів системи наведено в таблиці 3.3.

Таблиця 3.3 – Основні сценарії користувачів системи

Роль	Сценарій	Очікуваний результат
Адміністратор	Створення складу та місць зберігання	Формалізована топологія складу
Адміністратор	Ведення довідника товарів	Коректні габарити, вага та одиниці обліку
Адміністратор	Навчання моделі попиту	Активна або відхилена версія прогнозової моделі
Оператор	Приймання палети	Палета зі статусом очікування розміщення
Оператор	Підтвердження місця	Оновлений стан палети, місця та історії руху
Оператор	Виконання переміщення	Послідовно виконаний крок оптимізаційного плану

3.6 Концептуальна модель даних і алгоритм рекомендації

Концептуальна модель даних відображає основні зв'язки предметної області. Центальною сутністю є склад, оскільки більшість інших об'єктів

належить конкретному складу. До складу прив'язані місця зберігання, палети, історія попиту, рекомендації та плани оптимізації.

Товар і палета розділені як різні сутності. Товар описує довідкову інформацію: артикул, назву, габарити, вагу одиниці та допустиму кількість на палеті. Палета описує фактичну складську одиницю: номер, кількість, вагу, висоту, стан і поточне місце.

Рекомендація не змінює стан складу сама по собі. Вона фіксує запропоновану дію, а фактична зміна відбувається після підтвердження оператором. Це важливо, оскільки між рекомендацією та виконанням може змінитися стан складу.

Концептуальну модель даних системи наведено на рисунку 3.2.



Рисунок 3.2 – Концептуальна модель даних системи

Алгоритм рекомендації повинен бути пояснюваним для користувача. Якщо система пропонує об'єднання або конкретне місце, оператор повинен

розуміти, чому така дія є доцільною. Тому алгоритм побудовано як послідовність перевірок і оцінок, а не як непрозору модель.

На першому кроці система отримує палету, яка очікує розміщення, і перевіряє її параметри. Далі перевіряється можливість об'єднання з палетою такого самого товару. Якщо об'єднання можливе й не порушує обмеження місця зберігання, така рекомендація має пріоритет.

Якщо об'єднання неможливе, система переходить до пошуку місця. Для кожного кандидата перевіряються доступність, резервування, вантажопідйомність, допустима висота та відповідність складу. Після цього система обирає найкращий варіант з урахуванням відстані та інших факторів.

Схему алгоритму формування рекомендації наведено на рисунку 3.3.



Рисунок 3.3 – Схема алгоритму формування рекомендації щодо розміщення палети

3.7 Валідація, журнал операцій та аудит

Складські операції потребують суворої перевірки вхідних даних. Помилка в номері палети, артикулі, кількості або висоті може призвести до некоректної рекомендації. Тому система повинна перевіряти дані до виконання бізнес-операції.

Валідацію доцільно розділяти на синтаксичну та предметну. Синтаксична перевіряє наявність обов'язкових полів, діапазони числових значень і формат рядків. Предметна перевіряє існування складу, товару, місця зберігання, допустимість статусу та права користувача.

Обробка помилок має бути зрозумілою для користувача. Якщо палета не знайдена або місце не підходить за висотою, система повинна повідомити причину, а не просто відхилити операцію. Це особливо важливо для операторського інтерфейсу, де помилки виникають під час фізичної роботи.

Основні групи перевірок, які використовуються в системі, наведено в таблиці 3.4.

Таблиця 3.4 – Основні групи перевірок у системі

Група перевірок	Приклад	Результат порушення
Обов'язкові поля	відсутній номер палети або артикул	операція не виконується
Предметні зв'язки	товар або склад не існує	повідомлення про некоректні дані
Фізичні обмеження	вага або висота перевищує ліміт місця	місце відкидається з кандидатів
Статуси	палета вже розміщена або видалена	заборона повторної операції
Права доступу	оператор намагається керувати користувачами	доступ заборонено

Для складської системи важливо мати можливість відновити послідовність операцій. Журнал переміщень показує, як змінювалося місце палети, хто виконав дію, яку кількість було переміщено та чи була операція пов'язана з оптимізаційним планом.

Аудит подій має ширше призначення. Він може містити не лише переміщення, а й оцінювання оптимізації, створення планів, доставку подій, помилки та інші системні дії. Саме тому в проєкті передбачено outbox-підхід і можливість індексації подій в Elasticsearch.

Відокремлення історії рухів і аудиту є доцільним. Історія рухів потрібна користувачу для операційного контролю, а аудит подій – для технічного аналізу, пошуку інцидентів і інтеграції з іншими системами.

Журнал операцій також виконує важливу роль у контролі відповідності між цифровою моделлю складу та фактичним станом товарів. Якщо палета була переміщена, об'єднана або розміщена в новій комірці, система повинна зберегти не лише кінцевий результат, а й сам факт виконаної дії. Це дає змогу простежити історію переміщення конкретної палети, перевірити правильність дій оператора та виявити можливі розбіжності між даними системи й реальним складом.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ СИСТЕМИ

4.1 Структура проєкту та серверна частина

Програмну реалізацію системи виконано у вигляді вебзастосунку на основі Java та Spring Boot. Проєкт має модульну структуру, у якій окремі пакети відповідають за користувачів, автентифікацію, структуру складу, товари, палети, рекомендації розміщення, прогнозування попиту, маршрутизацію, оптимізаційні плани, аудит подій і тестування. Локальні інфраструктурні компоненти проєкту можуть запускатися через Docker Compose, а збирання Java-застосунку виконується засобами Maven [24], [25].

Основні програмні модулі системи наведено в таблиці 4.1.

Таблиця 4.1 – Основні програмні модулі системи

Модуль	Призначення
account, auth	користувачі, ролі, автентифікація та права доступу
warehouse	структура складу, ряди, рівні, місця зберігання
putaway.article	довідник товарів
putaway.container	приймання, розміщення, об'єднання та видалення палет
putaway.placement	формування рекомендацій щодо розміщення
demand.forecast	історія попиту, аналітика та прогнозна модель
warehouse.routing	графова модель складу та маршрутизація
optimization	оцінювання ефективності складу та плани переміщень
movement, eventing, search	журнал рухів, outbox-події, аудит та пошук

Серверна частина застосунку реалізована за багаторівневим підходом. Контролери приймають HTTP-запити, сервіси виконують бізнес-логіку, а репозиторії працюють із базою даних через Spring Data JPA. Для захисту

системи використано Spring Security, де адміністратор і оператор мають різні права доступу.

Адміністративний інтерфейс керування користувачами показано на рисунку 4.1.

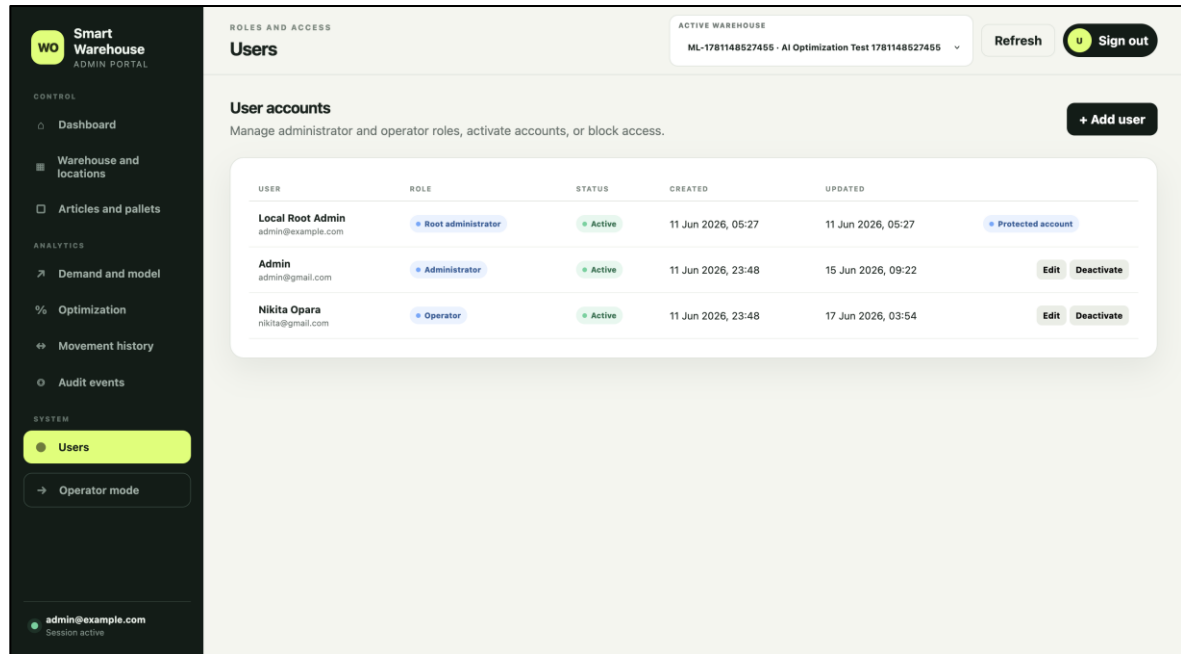


Рисунок 4.1 – Керування користувачами та ролями в адміністративному порталі

4.2 Складські модулі та облік палет

Модуль управління складом забезпечує створення структури складу та перегляд місць зберігання. Для кожного місця зберігаються код, рівень, допустима вага, допустима висота, відстань від точки входу та поточний стан. Ці параметри є основою для подальшої перевірки сумісності палет із місцями зберігання, оскільки система використовує їх під час формування рекомендацій, розрахунку маршрутів і планування оптимізаційних переміщень.

Сторінку перегляду структури складу наведено на рисунку 4.2.

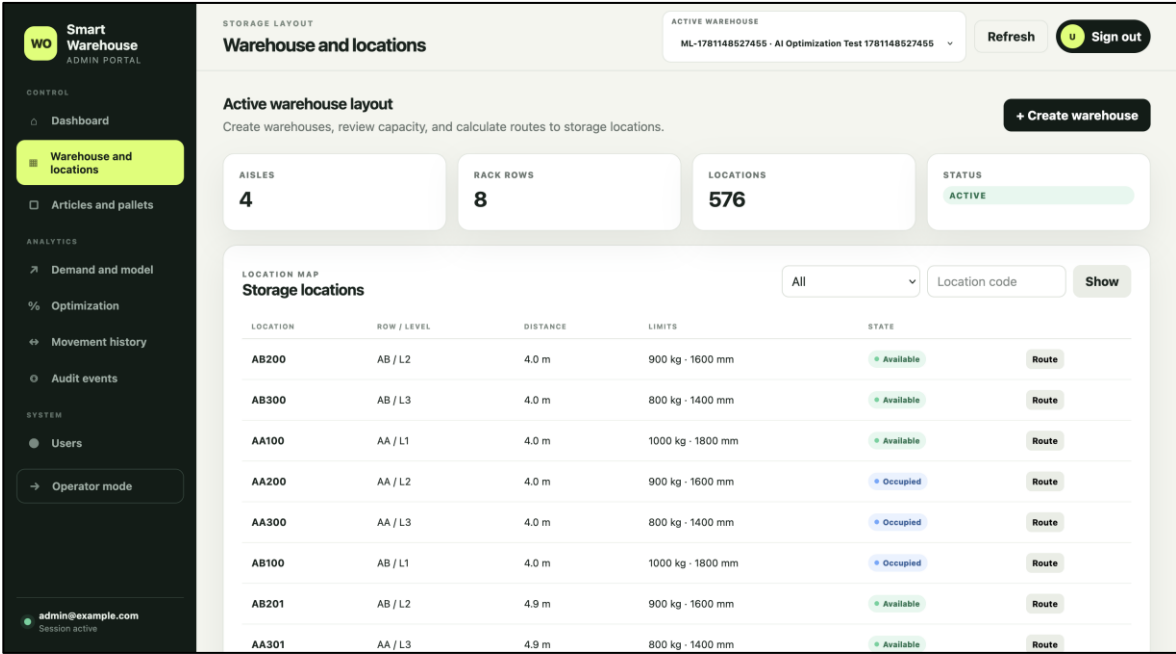


Рисунок 4.2 – Перегляд структури складу та місць зберігання

Довідник товарів містить артикули, назви, одиниці виміру, габарити, вагу та максимальну кількість на палеті.

Приклад довідника товарів наведено на рисунку 4.3.

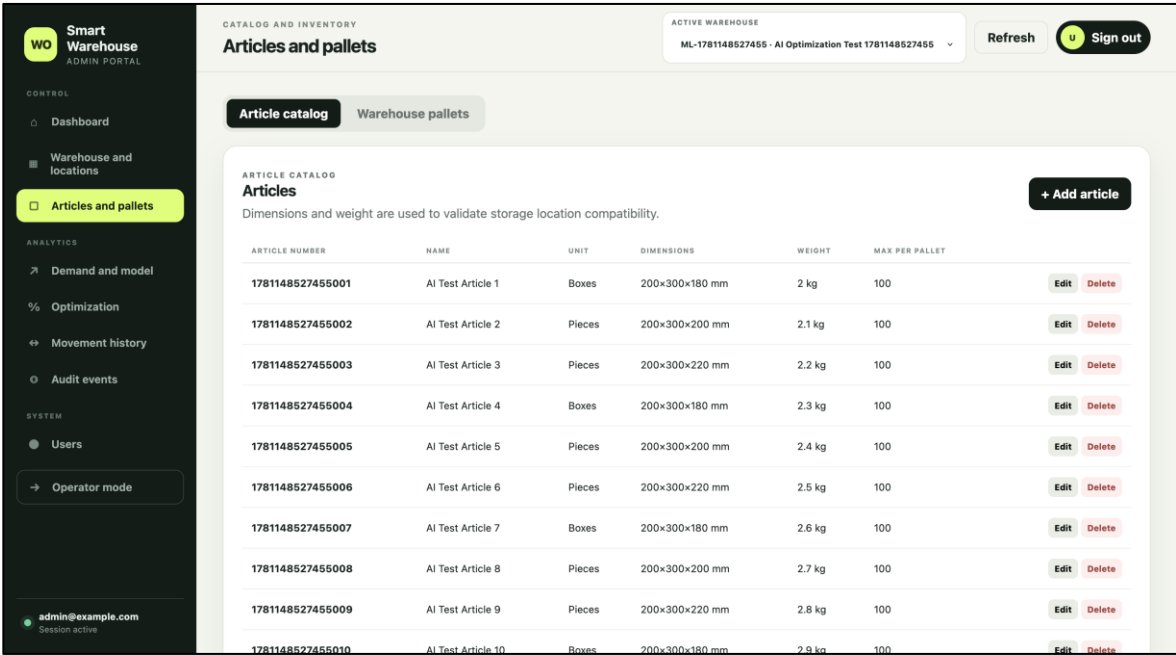


Рисунок 4.3 – Довідник товарів у вебінтерфейсі адміністратора

Палета пов’язується з конкретним товаром і має власний номер, кількість, вагу, висоту, статус і поточне місце зберігання.

Перелік палет і їх поточних статусів наведено на рисунку 4.4.

PALLET	ARTICLE	QUANTITY	WEIGHT / HEIGHT	LOCATION	STATE
MLC-1781148527455-001	1781148527455001 AI Test Article 1	40	80 kg · 1000 mm	—	Removed
MLC-1781148527455-002	1781148527455002 AI Test Article 2	40	84 kg · 1000 mm	AA300	Stored
MLC-1781148527455-003	1781148527455003 AI Test Article 3	40	88 kg · 1000 mm	AA200	Stored
MLC-1781148527455-004	1781148527455004 AI Test Article 4	40	92 kg · 1000 mm	AA101	Stored
MLC-1781148527455-005	1781148527455005 AI Test Article 5	20	48 kg · 1000 mm	AB100	Stored
MLC-1781148527455-006	1781148527455005 AI Test Article 5	25	60 kg · 1000 mm	—	Waiting for placement
MLC-1781148527455-007	1781148527455006 AI Test Article 6	40	100 kg · 1000 mm	—	Waiting for placement
MLC-1781148527455-008	1781148527455007 AI Test Article 7	40	104 kg · 1000 mm	—	Waiting for placement
MLC-1781148527455-009	1781148527455008 AI Test Article 8	40	108 kg · 1000 mm	—	Waiting for placement

Рисунок 4.4 – Перелік палет і їх поточних статусів

Фрагмент коду приймання палети наведено в лістингу 4.1.

Лістинг 4.1 – Реєстрація нової палети в системі

```
@Transactional
public ContainerResponse
receiveContainer(ReceiveContainerRequest request) {
    User actor =
    authenticatedUserService.getCurrentUser();

    containerValidationService.validateReceiveContainerRequest
    (actor, request);

    Warehouse warehouse = containerDataService
    .getWarehouseByIdOrThrow(request.warehouseId());
```

Продовження лістингу 4.1

```

        Article article = articleDataService

        .getByArticleNumberOrThrow(request.articleNumber().trim())
        ;

        Container container = Container.builder()
            .warehouse(warehouse)

        .containerNumber(request.containerNumber().trim())
            .article(article)
            .quantity(request.quantity())
            .weightKg(request.weightKg())
            .heightMm(request.heightMm())
            .status(ContainerStatus.WAITING_FOR_PLACEMENT)
            .build();

        return
        ContainerResponse.from(containerDataService.save(container
        ));
    }

```

У цьому фрагменті показано, що під час приймання палети система перевіряє користувача, валідність запиту, існування складу й товару, після чого створює палету зі статусом очікування розміщення.

4.3 Рекомендації щодо розміщення

Механізм рекомендацій визначає найдоцільнішу дію для палети, яка очікує розміщення. Система спочатку перевіряє можливість об'єднання палети з уже наявною палетою такого самого товару. Якщо відповідного варіанта немає, система шукає доступне місце зберігання.

Приклад отримання рекомендації щодо розміщення палети наведено на рисунку 4.5.

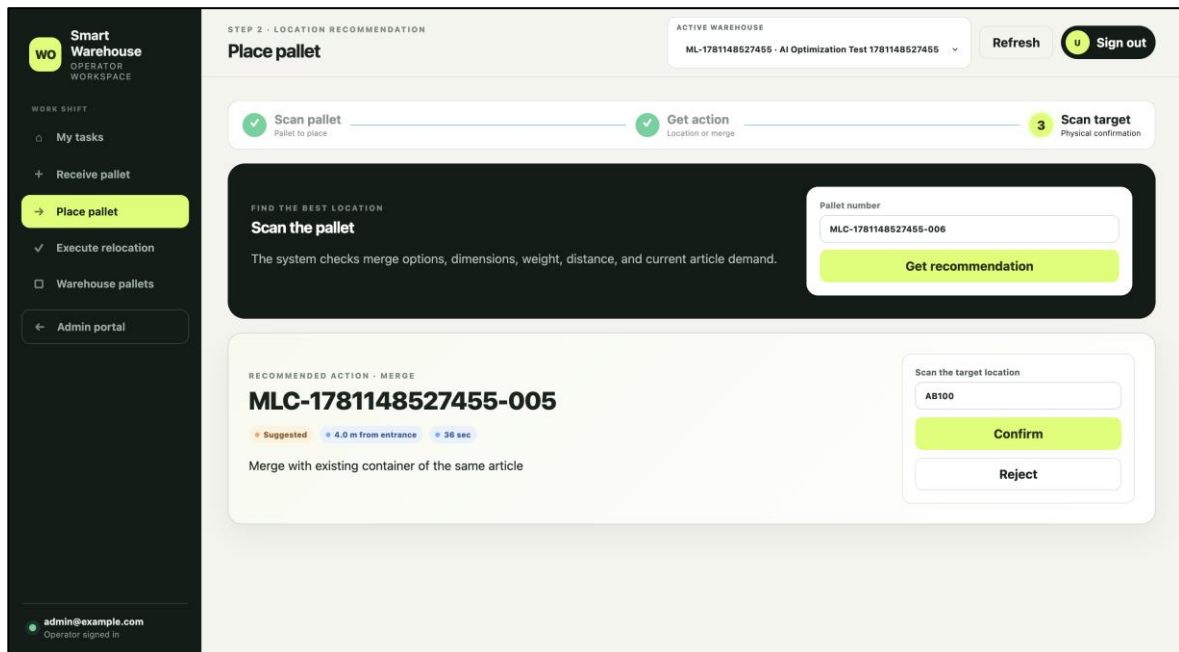


Рисунок 4.5 – Отримання рекомендації щодо розміщення палети

Фрагмент коду пошуку найкращого місця для розміщення наведено в лістингу 4.2.

Лістинг 4.2 – Пошук найкращого місця для розміщення палети

```
return storagePlaceRepository

    .findByWarehouseIdAndStatusOrderByDistanceFromEntryMmAsc (
        warehouseId,
        StoragePlaceStatus.AVAILABLE
    )
    .stream()
    .filter(place ->
        !blockedStoragePlaceIds.contains(place.getId())
    )
    .filter(place ->
        !place.isReservedForOptimization()
    )
```

Продовження лістингу 4.2

```

        .filter(place ->
            canStoragePlaceFitContainer(place, sourceContainer))
        .min(Comparator

.comparingInt(this::getStoragePlaceDistanceFromEntry)
        .thenComparing(StoragePlace::getCode));

```

Цей код демонструє логіку відбору кандидатів: система бере лише доступні місця, відкидає зарезервовані або несумісні комірки, а потім обирає найближче місце за відстанню.

4.4 Прогнозування попиту, маршрутизація та оптимізаційні плани

Для врахування майбутнього попиту в системі реалізовано модуль аналізу історії замовлень і навчання прогнозної моделі. Історичні дані використовуються для формування навчального набору, після чого модель оцінює очікуваний попит на товари.

Сторінку аналізу попиту та навчання моделі наведено на рисунку 4.6.

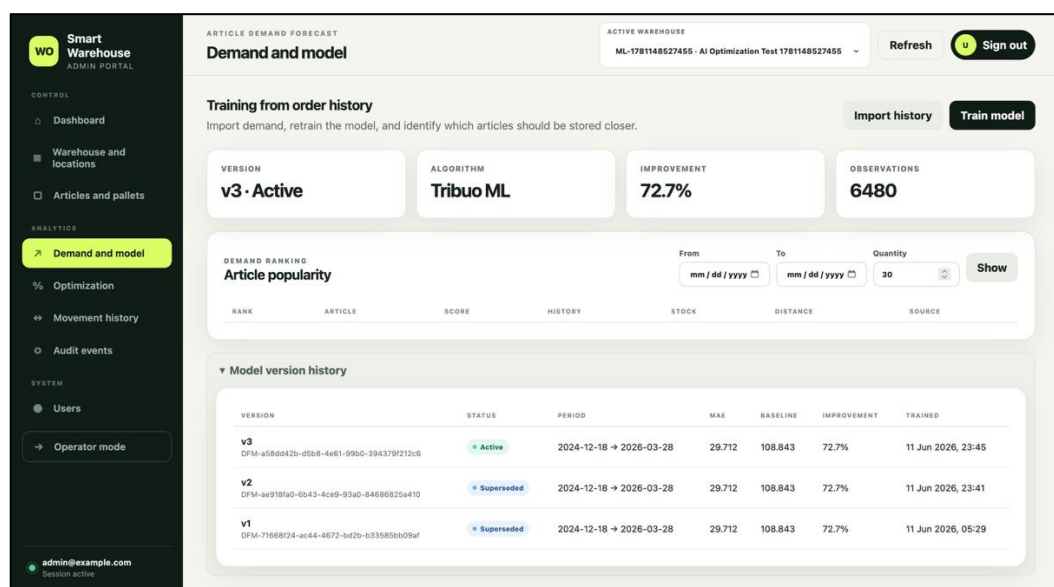


Рисунок 4.6 – Сторінка аналізу попиту та навчання моделі

Фрагмент коду активації прогнозової моделі наведено в лістингу 4.3.

Лістинг 4.3 – Активація моделі прогнозування попиту

```
TrainedDemandForecast trained = trainer.train(dataset);
applyMetrics(modelVersion, trained.metrics());

if (shouldActivate(trained.metrics())) {

    modelRepository.findFirstByWarehouseIdAndStatusOrderByTrainedAtDesc(
        warehouseId,
        DemandForecastModelStatus.ACTIVE
    ).ifPresent(active -> {
        active.supersede();
        modelRepository.saveAndFlush(active);
    });
    modelVersion.activate(trained.artifact(),
        completedAt);
} else {
    modelVersion.reject("Candidate did not improve
        baseline", completedAt);
}
```

Нова модель активується лише тоді, коли її якість є кращою за базовий підхід. Якщо модель не демонструє достатнього покращення, вона зберігається як відхилена і не використовується під час формування рекомендацій.

Після активації прогнозової моделі її результати можуть застосовуватися не лише для окремих рекомендацій, а й для загального оцінювання стану складу. Система враховує очікувану популярність товарів, відстані до місць зберігання та можливість подальших переміщень, щоб визначити палети, переміщення яких може підвищити ефективність складської топології.

Результат оцінювання ефективності складу та створення оптимізаційного плану наведено на рисунку 4.7.

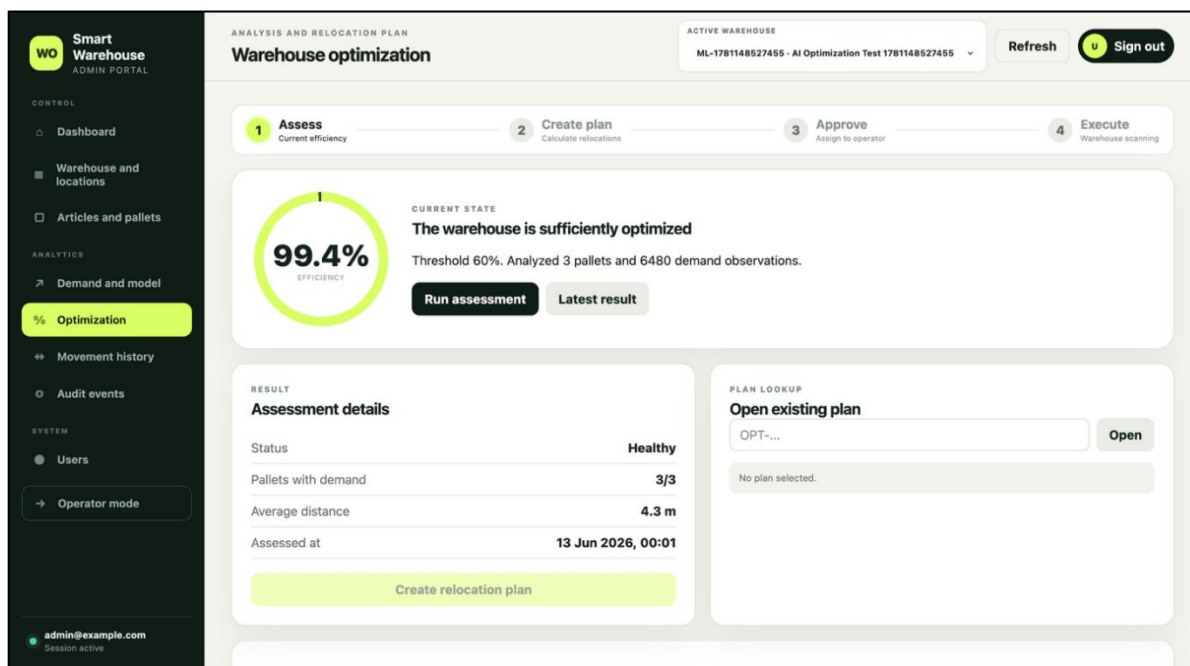


Рисунок 4.7 – Оцінювання ефективності складу та створення плану оптимізації

Для маршрутизації склад подано у вигляді графа. Вершинами є логічні точки складу або місця зберігання, а ребра відображають можливі переходи між ними. Для пошуку найкоротшого маршруту використано алгоритм Дейкстри.

Фрагмент коду реалізації пошуку найкоротших відстаней у графі складу наведено в лістингу 4.4.

Лістинг 4.4 – Пошук найкоротших відстаней у графі складу

```
while (!queue.isEmpty()) {
    NodeDistance current = queue.poll();
    if (current.distanceMm() !=
        distances.get(current.nodeId())) {
        continue;
    }
```

Продовження лістингу 4.4

```

    }

    for (WarehouseGraph.RouteEdge edge :
graph.getEdges(current.nodeId())) {
        int candidateDistance = Math.addExact(
            current.distanceMm(),
            edge.distanceMm()

        );

        if (candidateDistance <
distances.get(edge.targetNodeId())) {
            distances.put(edge.targetNodeId(),
candidateDistance);
            queue.add(new
NodeDistance(edge.targetNodeId(), candidateDistance));
        }
    }
}

```

Сторінку виконання переміщення наведено на рисунку 4.8.

The screenshot shows the 'Smart Warehouse Operator Workspace' interface. The sidebar on the left contains navigation links: 'My tasks', 'Receive pallet', 'Place pallet', 'Execute relocation' (highlighted), 'Warehouse pallets', and 'Admin portal'. The main content area is titled 'APPROVED OPTIMIZATION PLAN' and 'Execute relocation'. It features a warning banner: 'Execute only the current step. The pallet number and destination location must be confirmed by physical scans.' Below this, there are two steps: 'STEP 1: Open plan' with a search bar and a 'Show task' button, and 'STEP 2: Confirm relocation' with input fields for 'Plan code', 'Scanned pallet', 'New location', and 'Target pallet for merging', followed by a 'Confirm and get next step' button. The top right corner shows 'ACTIVE WAREHOUSE' information and a 'Sign out' button.

Рисунок 4.8 – Виконання кроку переміщення оператором

Наведений фрагмент демонструє базову логіку пошуку найкоротших відстаней у графі складу. Черга використовується для послідовної обробки вершин із найменшою поточною відстанню, а для кожного ребра перевіряється, чи можна покращити вже знайдений маршрут. Якщо нова відстань є меншою, значення оновлюється, а відповідна вершина повторно додається до черги.

На основі попиту та маршрутних відстаней система може оцінювати поточну ефективність складу. Якщо популярні товари розміщені надто далеко, система формує план переміщень, який оператор виконує послідовно.

4.5 Вебінтерфейс адміністратора й оператора

Веб-інтерфейс системи розділено на адміністративний портал і робочий простір оператора. Адміністратор керує складом, товарами, користувачами, моделями попиту, оптимізацією та аудитом. Оператор працює з практичними складськими операціями: прийманням палет, отриманням рекомендацій, підтвердженням розміщення та виконанням переміщень.

Адміністративний портал орієнтований на підготовку та супровід даних, без яких неможлива коректна робота рекомендаційного механізму. Через нього адміністратор створює структуру складу, веде довідник товарів, керує користувачами, імпортує історію попиту та переглядає результати оптимізації. Такий поділ дозволяє відокремити налаштування системи від щоденних операцій із палетами.

Робочий простір оператора має бути простішим і більш прикладним, оскільки він використовується під час виконання фізичних складських дій. Основна увага в ньому приділяється швидкому введенню номера палети, отриманню рекомендації, підтвердженню місця зберігання та виконанню кроків оптимізаційного плану.

Робочий простір оператора наведено на рисунку 4.9.

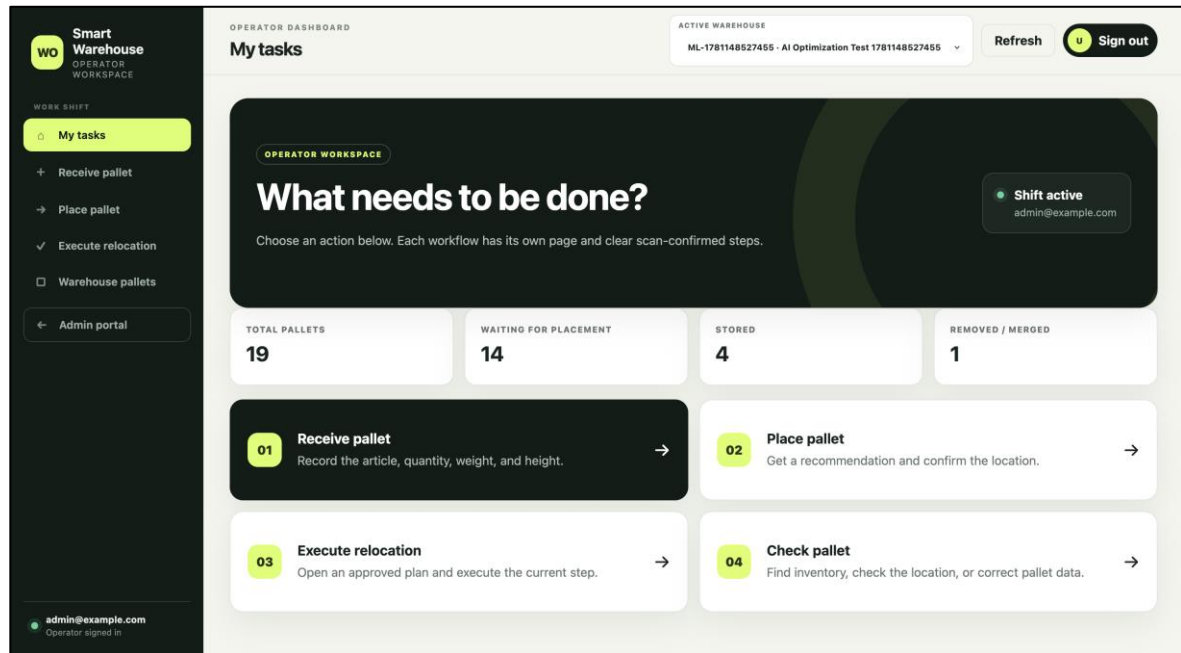


Рисунок 4.9 – Робочий простір оператора складу

Поточна реалізація системи орієнтована на перевірку бізнес-логіки складської оптимізації без використання фізичного складського обладнання. Через відсутність апаратного сканера сценарій сканування імітується ручним введенням номера палети або коду місця зберігання у відповідне поле. Водночас така умовність не змінює логіку процесу: користувач так само ідентифікує палету, отримує рекомендацію та підтверджує фактичне місце розміщення. У майбутньому ручне введення може бути замінено сканером, який працює як пристрій введення.

У реальному сценарії кожна палета та кожне місце зберігання можуть мати штрихкод або QR-код. Оператор сканує палету, система отримує інформацію про товар, кількість, вагу та висоту, після чого формує рекомендацію. Після фізичного переміщення оператор сканує код місця зберігання, а система перевіряє, чи відповідає фактична локація рекомендованій.

Перспективами розвитку є підключення сканерів штрихкодів, створення мобільного інтерфейсу для операторів, інтеграція з ERP/WMS-системами, підтримка декількох складів, урахування заблокованих проходів, а також використання складніших моделей прогнозування попиту.

4.6 Тестування системи та аналіз результатів

Для перевірки системи використано модульні та інтеграційні тести. Тестування охоплює рекомендації розміщення, маршрутизацію, прогнозування попиту, оптимізаційні плани, безпеку, аудит подій і роботу інфраструктурних компонентів.

Основні напрями тестування наведено в таблиці 4.2.

Таблиця 4.2 – Основні напрями тестування системи

Напрямок тестування	Приклади перевірок
Рекомендації розміщення	Вибір доступного місця, перевірка обмежень, повторне використання кандидатів
Маршрутизація	Вибір найкоротшого маршруту, розрахунок відстаней
Прогнозування попиту	Імпорт даних, навчання моделі, оцінювання якості
Оптимізація	Оцінка ефективності складу, формування плану переміщень
Безпека	Перевірка доступу для ролей адміністратора й оператора
Аудит	Запис переміщень, outbox-події, індексація подій

Журнал переміщень палет наведено на рисунку 4.10.

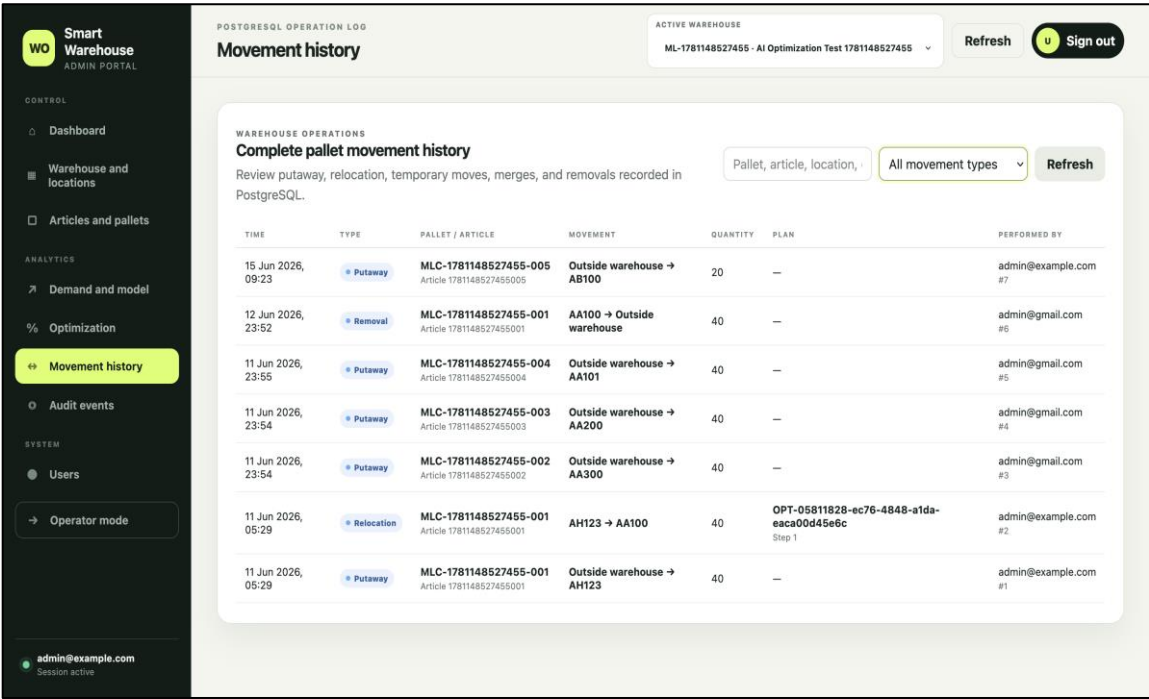


Рисунок 4.10 – Журнал переміщень палет

Події системи додатково можуть індексуватися для пошуку та аудиту. Приклад перегляду подій наведено на рисунку 4.11.

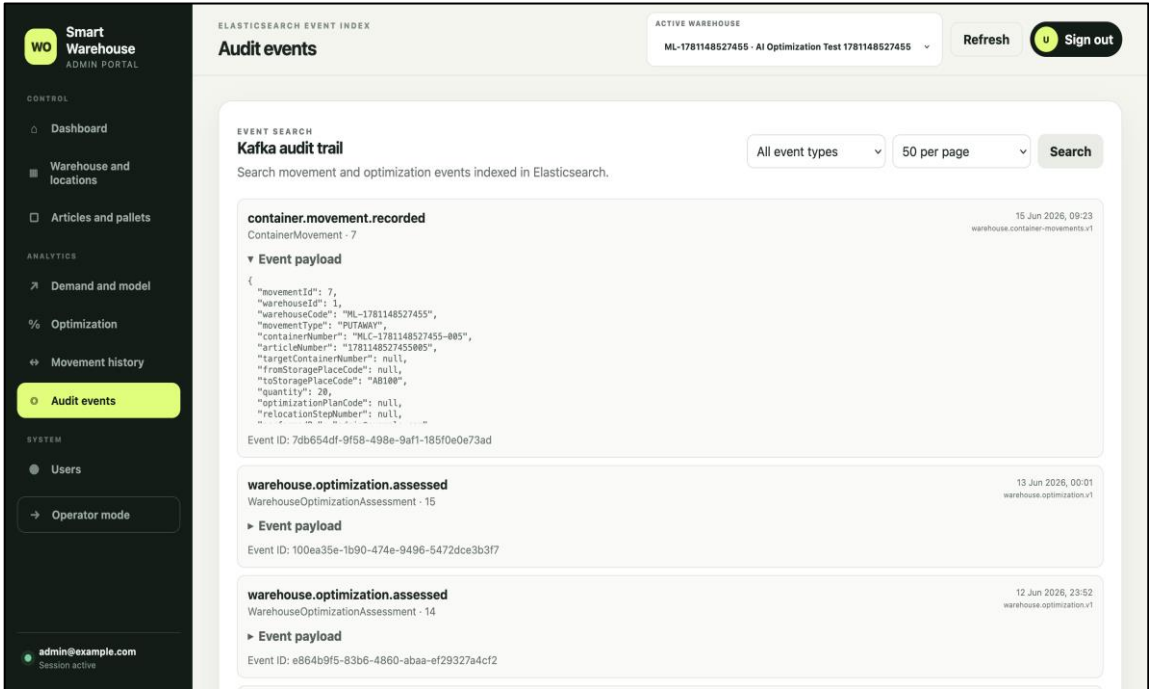


Рисунок 4.11 – Аудит подій системи

Результати тестування підтверджують працездатність основних модулів системи та коректність реалізації ключових сценаріїв: приймання палет, формування рекомендацій, розміщення, об'єднання, маршрутизації, аналізу попиту й планування переміщень.

4.7 Додаткові реалізовані сценарії

Створення структури складу реалізовано як адміністративний сценарій. Користувач задає код і назву складу, кількість проходів, рядів, секцій, рівнів, місць на рівні, ширину проходу та обмеження за вантажопідйомністю. На основі цих параметрів система генерує набір місць зберігання.

Під час генерації враховуються рівні зберігання, вантажні обмеження та відстані від входу. Це дає змогу одразу використовувати склад не лише як довідник місць, а й як основу для маршрутизації та рекомендацій.

Форму створення складу з основними параметрами наведено на рисунку 4.12.

The image shows a web application interface for creating a new warehouse. The form is titled 'NEW WAREHOUSE' and has a 'Main parameters' section. It includes input fields for 'Code', 'Name', and 'Aisles' (set to 4). Below these are 'Rack rows' (set to 8), 'Bays per rack row' (set to 8), and 'Pallet places per level' (set to 3). There is also a 'Rack levels' section with 'Rack levels' (set to 3) and 'Aisle width, mm' (set to 3500). A 'Bay load limit, kg' is set to 9000. A note indicates 'Vertical levels. Level cards are generated below.' and a 'Calculated bay load: 8,100 kg' is shown. The 'Rack levels' section displays three levels: Level 1 (Height: 1800 mm, Weight: 1000 kg), Level 2 (Height: 1600 mm, Weight: 900 kg), and Level 3 (Height: 1400 mm, Weight: 800 kg). A 'Create warehouse' button is at the bottom.

Parameter	Value
Code	
Name	
Aisles	4
Rack rows	8
Bays per rack row	8
Pallet places per level	3
Rack levels	3
Aisle width, mm	3500
Bay load limit, kg	9000
Calculated bay load	8,100 kg
Rack levels	
Level	Height, mm
Level 1	1800
Level 2	1600
Level 3	1400
Weight, kg	
Level 1	1000
Level 2	900
Level 3	800

Рисунок 4.12 – Форма створення структури складу

Приклад параметрів, які передаються під час створення структури складу, наведено в лістингу 4.5.

Лістинг 4.5 – Приклад параметрів створення складу

```
CreateWarehouseRequest(  
    code = "ML-1781148527455",  
    name = "AI Optimization Test",  
    aisles = 4,  
    rackRows = 8,  
    baysPerRackRow = 8,  
    rackLevels = 3,  
    palletPlacesPerLevel = 3,  
    aisleWidthMm = 3500  
)
```

Наведений приклад показує, що склад описується не вручну створеними окремими місцями, а набором параметрів, з яких система формує регулярну топологію. Це спрощує початкове налаштування та зменшує кількість помилок.

4.7.1 Імпорт історії попиту

Історія попиту використовується як джерело даних для аналітики та навчання прогнозної моделі. У поточній реалізації імпорт виконується через JSON-масив замовлень. Кожне замовлення містить номер, дату та перелік позицій із артикулами й кількістю.

Перед збереженням система перевіряє, що артикул товару вже існує в довіднику. У реальному впровадженні цей імпорт може бути замінений інтеграцією з ERP або системою продажів.

Після імпорту замовлень дані стають доступними для аналітики популярності товарів, побудови навчального набору та оцінювання

прогнозної моделі. Саме ці дані потім впливають на рекомендації щодо того, які товари варто зберігати ближче.

Форму імпорту історії замовлень наведено на рисунку 4.13.

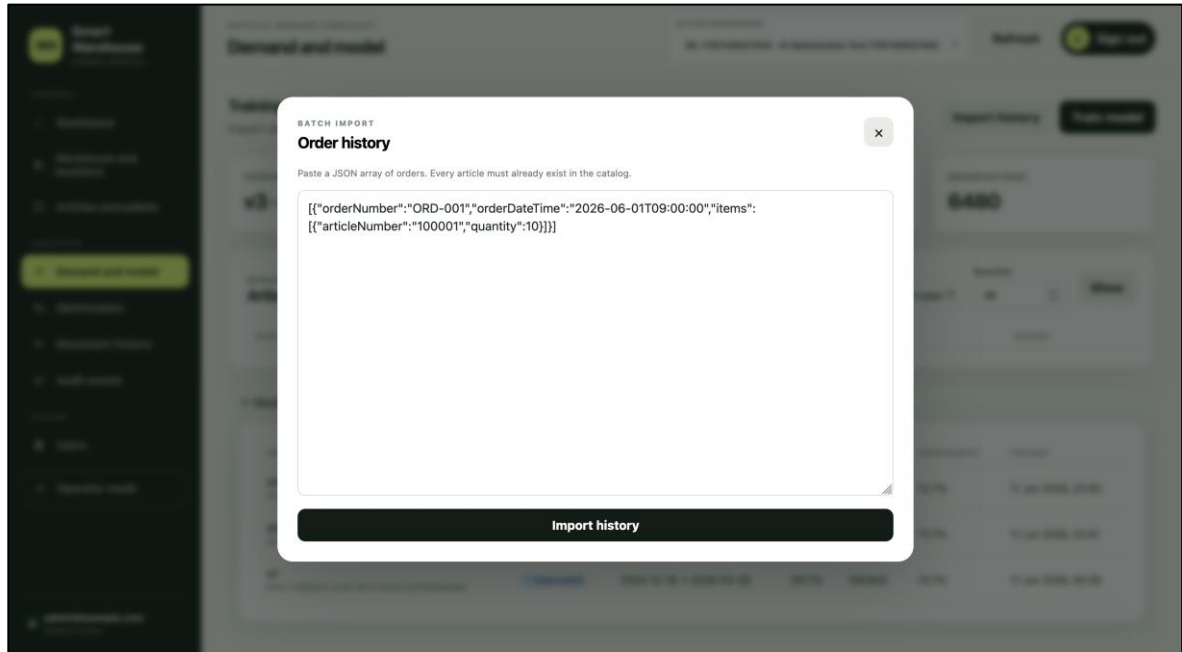


Рисунок 4.13 – Форма імпорту історії замовлень

Приклад структури імпортованого замовлення, що використовується для завантаження історії попиту, наведено в лістингу 4.6.

Лістинг 4.6 – Приклад структури імпортованого замовлення

```
[
  {
    "orderNumber": "ORD-001",
    "orderDateTime": "2026-06-01T09:00:00",
    "items": [
      { "articleNumber": "100001", "quantity": 10 }
    ]
  }
]
```

Такий формат достатній для навчальної реалізації, оскільки він містить головні ознаки попиту: час, артикул і кількість. Надалі формат може бути розширений полями клієнта, типу замовлення, зони відвантаження або пріоритету.

4.7.2 Адміністрування користувачів

Модуль користувачів забезпечує створення, редагування, активацію та деактивацію облікових записів. Для кожного користувача зберігаються електронна пошта, повне ім'я, пароль, роль, статус та службові часові позначки.

З точки зору складської системи користувач не є лише записом для входу. Саме користувач пов'язується з виконаними діями, переміщеннями та подіями аудиту.

У системі передбачено захищений root-адміністратор, який не може бути випадково деактивованій через інтерфейс. Це зменшує ризик втрати адміністративного доступу під час тестування або експлуатації.

Форму створення нового користувача наведено на рисунку 4.14.

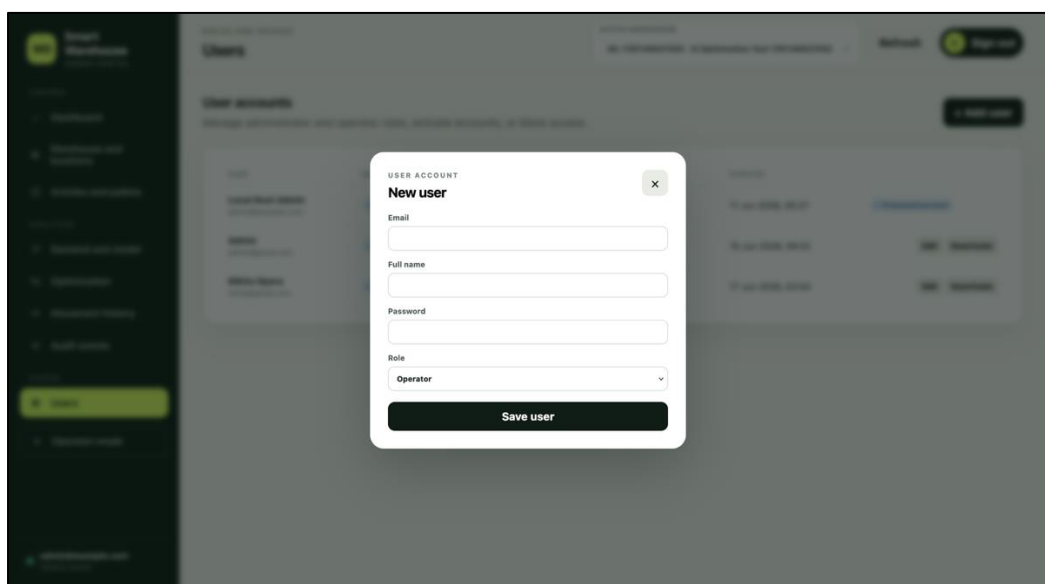
The image shows a web application interface with a dark sidebar on the left. The main content area is titled 'USER ACCOUNT' and 'New user'. It contains a form with the following fields: 'Email' (text input), 'Full name' (text input), 'Password' (text input), and 'Role' (a dropdown menu currently showing 'Operator'). At the bottom of the form is a 'Save user' button. The background is slightly blurred, showing a list of users.

Рисунок 4.14 – Форма створення нового користувача

Розподіл ролей користувачів і доступних для них дій наведено в таблиці 4.3.

Таблиця 4.3 – Ролі користувачів і доступні дії

Роль	Основні дії	Обмеження
ROOT_ADMIN	Повне адміністрування системи, керування користувачами	Обліковий запис захищений від деактивації
ADMIN	Склад, товари, палети, попит, оптимізація, аудит	Не повинен змінювати захищений root-акаунт
OPERATOR	Приймання, розміщення, виконання переміщень, перегляд палет	Немає доступу до адміністрування користувачів

4.7.3 Довідник товарів і коригування даних

Довідник товарів є одним з основних джерел даних для рекомендацій. Якщо габарити або вага товару задані неправильно, система може помилково вважати місце сумісним або несумісним. Тому в інтерфейсі передбачено створення та редагування товарів.

Редагування товару включає артикул, назву, одиницю виміру, ширину, довжину, висоту, вагу та максимальну кількість на палеті. Ці параметри використовуються під час приймання палет і перевірки фізичних обмежень місць зберігання.

Окремо реалізовано коригування параметрів палети. Це потрібно для ситуацій, коли під час приймання було помилково введено кількість, вагу або висоту. Адміністратор може виправити дані, після чого рекомендації надалі формуються на основі актуальних параметрів (рисунок 4.15). Крім того, внесені зміни автоматично оновлюють історію операцій, що гарантує цілісність даних аудиту. Можливість оперативного виправлення критично важлива для підтримання відповідності між цифровою моделлю та фактичним станом товарів на складі.

The screenshot displays the 'Articles and pallets' management interface. A modal window titled 'Edit 1781148527455001' is open, allowing for the editing of article parameters. The form includes the following fields:

- Article number:** 1781148527455001
- Name:** AI Test Article 1
- Unit:** Boxes
- Width, mm:** 200
- Length, mm:** 300
- Height, mm:** 180
- Weight, kg:** 2
- Max per pallet:** 100

A 'Save article' button is located at the bottom of the modal. The background shows a table of articles with columns for article number, name, unit, weight, and height.

Рисунок 4.15 – Форма редагування параметрів товару

Форму коригування параметрів палети наведено на рисунку 4.16.

The screenshot displays the 'Warehouse pallets' management interface. A modal window titled 'Correction Pallet MLC-1781148527455-001' is open, allowing for the correction of pallet parameters. The form includes the following fields:

- Quantity:** 40
- Weight, kg:** 80
- Height, mm:** 1000

A 'Save changes' button is located at the bottom of the modal. The background shows a table of warehouse pallets with columns for pallet number, article, unit, weight, and height.

Рисунок 4.16 – Форма коригування параметрів палети

4.7.4 Операторський пошук палет

Операторський інтерфейс містить сторінку пошуку палет і товарів. Вона потрібна для перевірки поточного стану запасів, уточнення місця зберігання, перегляду кількості, ваги, висоти та статусу палети. Такий сценарій є важливим для підтримки фактичної роботи складу.

Пошук може виконуватися за номером палети або артикулом. У майбутньому це поле може приймати значення від сканера штрихкоду. У поточній версії користувач вводить значення вручну, але логіка сценарію відповідає майбутньому скануванню.

Наявність сторінки перевірки палет також корисна для виправлення розбіжностей між фізичним складом і цифровою моделлю. Якщо оператор бачить, що фактична палета має інші параметри, адміністратор може скоригувати дані через відповідний інтерфейс.

Сторінку операторського пошуку палет наведено на рисунку 4.17.

Smart Warehouse OPERATOR WORKSPACE

WORK SHIFT

- My tasks
- Receive pallet
- Place pallet
- Execute relocation
- Warehouse pallets**
- Admin portal

admin@example.com
Operator signed in

SEARCH AND DATA CORRECTION
Warehouse pallets

ACTIVE WAREHOUSE
ML-1781148527455 - AI Optimization Test 1781148527455

Refresh Sign out

CURRENT INVENTORY
Find pallet or article
Check quantity, dimensions, status, and physical location.

Pallet number or article All statuses Refresh

PALLET	ARTICLE	QUANTITY	WEIGHT / HEIGHT	LOCATION	STATE	ACTIONS
MLC-1781148527455-001	1781148527455001 AI Test Article 1	40	80 kg - 1000 mm	—	Removed	Edit Remove
MLC-1781148527455-002	1781148527455002 AI Test Article 2	40	84 kg - 1000 mm	AA300	Stored	Edit Remove
MLC-1781148527455-003	1781148527455003 AI Test Article 3	40	88 kg - 1000 mm	AA200	Stored	Edit Remove
MLC-1781148527455-004	1781148527455004 AI Test Article 4	40	92 kg - 1000 mm	AA101	Stored	Edit Remove
MLC-1781148527455-005	1781148527455005 AI Test Article 5	20	48 kg - 1000 mm	AB100	Stored	Edit Remove
MLC-1781148527455-006	1781148527455006 AI Test Article 6	25	60 kg - 1000 mm	—	Waiting for placement	Edit Remove
MLC-1781148527455-007	1781148527455007 AI Test Article 7	40	100 kg - 1000 mm	—	Waiting for placement	Edit Remove
MLC-1781148527455-008	1781148527455008 AI Test Article 8	40	104 kg - 1000 mm	—	Waiting for placement	Edit Remove
MLC-1781148527455-009	1781148527455009 AI Test Article 9	40	108 kg - 1000 mm	—	Waiting for placement	Edit Remove
MLC-1781148527455-010	1781148527455010 AI Test Article 10	40	112 kg - 1000 mm	—	Waiting for placement	Edit Remove

Рисунок 4.17 – Операторський перегляд і пошук палет

4.8 Події, аудит та API системи

Після важливих складських операцій система формує доменні події. Наприклад, переміщення палети створює подію з ідентифікатором руху, кодом складу, типом операції, номером палети, початковим і кінцевим місцем, кількістю та користувачем.

Щоб подія не втрачалася у разі помилки зовнішньої інфраструктури, використано outbox-підхід. Подія спочатку зберігається в PostgreSQL разом з основною операцією. Після цього окремий сервіс може доставити її до Kafka, а інший компонент – проіндексувати в Elasticsearch.

Такий підхід відділяє основну складську операцію від зовнішніх інтеграцій. Якщо Kafka або Elasticsearch тимчасово недоступні, основна операція не повинна втрачати свої дані. Подію можна повторно доставити після відновлення інфраструктури.

Потік подій, аудиту та спостережуваності наведено на рисунку 4.18.



Рисунок 4.18 – Потік подій, аудиту та спостережуваності

Узагальнену структуру події переміщення палети наведено в лістингу 4.7.

Лістинг 4.7 – Узагальнена структура події переміщення

```
ContainerMovementEventPayload(
    movementId = movement.getId(),
    warehouseCode = warehouse.getCode(),
    movementType = movement.getMovementType(),
    containerNumber = movement.getContainerNumber(),
    fromStoragePlaceCode =
movement.getFromStoragePlaceCode(),
    toStoragePlaceCode = movement.getToStoragePlaceCode(),
    quantity = movement.getQuantity(),
    performedBy = movement.getPerformedBy()
)
```

4.8.1 API та серверні контролери

Серверні контролери виконують роль межі між HTTP-запитами та бізнес-логікою. Вони приймають параметри форм або JSON-запити, передають їх до сервісів, а потім повертають сторінку, перенаправлення або відповідь API.

У системі є контролери для складів, товарів, палет, рекомендацій розміщення, історії попиту, моделей прогнозування, оптимізації, користувачів, маршрутизації та аудиту. Така структура відповідає модульному поділу проєкту.

Документація API може формуватися через OpenAPI/Swagger. Це важливо для подальшої інтеграції із зовнішніми клієнтами, мобільним застосунком або тестовими інструментами.

Для кращого розуміння структури серверної частини доцільно окремо виділити основні модулі, які відповідають за обробку запитів, виконання бізнес-логіки та взаємодію з даними.

Приклади серверних модулів системи та їх відповідальності наведено в таблиці 4.4.

Таблиця 4.4 – Приклади серверних модулів і відповідальності

Модуль	Приклад відповідальності	Типові дані
WarehouseController	Створення складу, перегляд місць, маршрути	Склад, місце, route response
Article / Container	Товари, палети, приймання, коригування	Артикули, кількість, вага, висота
PlacementRecommendationController	Формування та підтвердження рекомендацій	Тип рекомендації, місце, статус
DemandHistory / Forecast	Імпорт попиту, навчання та версії моделей	Замовлення, метрики, модель
Optimization	Оцінювання ефективності та плани переміщень	План, крок, статус виконання
AuditSearchController	Пошук подій аудиту	Тип події, payload, час

4.8.2 Перевірка якості сервісів

Для перевірки системи використано модульні та інтеграційні тести. Вони охоплюють не лише окремі сервіси, а й важливі сценарії взаємодії модулів: рекомендації, маршрутизацію, оптимізаційні резервування, події, аудит, блокування та безпеку.

Тестування рекомендацій перевіряє, що система відкидає несумісні місця, враховує резервування, надає пріоритет об'єднанню та коректно працює з доступними місцями. Це важливо, оскільки рекомендація є центральним результатом роботи системи.

Тестування маршрутизації перевіряє побудову графа складу й пошук найкоротших шляхів. Тестування прогнозування попиту перевіряє формування набору даних, навчання моделі, метрики та правило активації нової версії.

Окремо перевіряються інфраструктурні сценарії: outbox-події, Kafka-диспетчер, індексація аудиту, Redis-блокування, health indicator і correlation id. Це показує, що система спроектована не лише як набір сторінок, а як застосунок із елементами промислової архітектури.

Приклади тестів, реалізованих у проєкті, наведено в таблиці 4.5.

Таблиця 4.5 – Приклади тестів, реалізованих у проєкті

Група тестів	Приклади класів	Що перевіряється
Рекомендації	Placementcandidateservicetest, placementrecommendationfactorytest	Відбір місць, сумісність, тип рекомендації
Маршрутизація	Dijkstrawarehouseroutertest, warehouseoutecalculatorstest	Найкоротші відстані та маршрути
Попит і ML	Demandforecasttrainingservicetest, tribudemandforecasttrainertest	Навчання, метрики, активація моделі
Оптимізація	Warehouse relocationplannertest, warehouseoptimizationassessmentservicetest	Плани переміщень, ефективність
Події та аудит	Domaineventpublishertest, warehouseauditeventindexertest	Outbox, індексація, payload подій
Безпека та UI	Securityconfigtest, warehouseuicontrollertest	Доступ за ролями та маршрути інтерфейсу

Приклад перевірки кандидата на розміщення в автоматизованому тесті наведено в лістингу 4.8.

Лістинг 4.8 – Приклад перевірки кандидата на розміщення

```
assertThat(candidate.storagePlaceCode()).isEqualTo("AA100")
);
assertThat(candidate.distanceFromEntryMm()).isPositive();
assertThat(candidate.canFit()).isTrue();
assertThat(candidate.reason()).contains("available");
```

Наведений фрагмент є спрощеним прикладом того, як у тестах перевіряються очікувані властивості кандидата на розміщення. Реальні тести проєкту додатково перевіряють граничні випадки, помилки та взаємодію з репозиторіями.

4.9 Експлуатаційні обмеження та підготовка до розвитку

Поточна версія системи є прототипом із достатньо повною бізнес-логікою, але має низку експлуатаційних обмежень. Найважливішим є відсутність фізичної інтеграції зі сканером штрихкодів. Через це сценарії сканування виконуються через ручне введення номерів палет і кодів місць.

Другим обмеженням є спрощена модель складу. Вона добре підходить для регулярної топології з проходами, рядами та рівнями, але не враховує складні зони, тимчасово закриті проходи, різні типи стелажів, температурні режими або спеціальні правила сумісності товарів.

Третім обмеженням є характер прогнозової моделі. Вона використовує історію попиту, але не враховує зовнішні фактори, маркетингові акції, календарні події, постачальницькі обмеження або майбутні замовлення. Це є напрямом подальшого розвитку.

Незважаючи на ці обмеження, архітектура системи залишає простір для розширення. Поля введення можуть бути замінені сканером, серверна логіка може отримати мобільний клієнт, а модуль прогнозування може бути доповнений складнішими моделями або інтеграцією з ERP.

Оскільки система має модульну структуру, кожен серверний компонент виконує окрему частину загальної логіки. Подальший розвиток системи доцільно розглядати як поетапне розширення поточної реалізації з урахуванням виявлених обмежень.

Основні напрями подальшого розвитку системи наведено в таблиці 4.6.

Таблиця 4.6 – Напрями подальшого розвитку системи

Напрямок	Поточний стан	Можливий розвиток
Сканування	Ручне введення номерів і кодів	Підключення сканера штрихкодів або QR-кодів
Мобільність	Браузерний інтерфейс	Окремий мобільний інтерфейс для оператора
Інтеграція	Локальний імпорт історії попиту	Обмін даними з ERP або WMS
Топологія	Регулярна структура складу	Підтримка зон, блокувань і різних типів місць
Прогнозування	Модель на основі історії замовлень	Урахування сезонності, акцій і зовнішніх факторів

4.9.1 Міграції бази даних

Структура бази даних у проєкті змінюється через Flyway-міграції. Такий підхід є важливим для відтворюваності: кожен запуск застосунку може привести схему бази до потрібного стану без ручного виконання SQL-команд.

Міграції розділено за етапами розвитку системи. Спочатку створюються користувачі та базова структура складу, потім таблиці товарів і палет, рекомендації розміщення, історія попиту, оптимізаційні оцінки, плани переміщень, історія рухів і outbox-події.

У межах кваліфікаційної роботи це має практичне значення, оскільки база даних не є випадковим набором таблиць. Її структура відображає еволюцію функціональності: від базового складського обліку до подій, аудиту та прогнозування.

Flyway також спрощує перевірку роботи на іншому комп'ютері. Якщо база створюється з нуля, достатньо підняти PostgreSQL і запустити застосунок, після чого всі міграції виконуються в заданому порядку. Такий

поділ дозволяє простежити, які частини схеми бази даних відповідають окремим функціональним модулям системи.

Основні групи Flyway-міграцій, використаних у проєкті, наведено в таблиці 4.7.

Таблиця 4.7 – Основні групи Flyway-міграцій

Група міграцій	Приклад файлів	Призначення
Користувачі та безпека	V1__create_users_table.sql	Створення таблиць користувачів, ролей і статусів
Складська структура	V2__create_warehouse_layout_tables.sql	Склади, проходи, ряди, рівні та місця
Палети й товари	V3__create_putaway_tables.sql	Товари, палети, статуси й зв'язки
Рекомендації	V4__create_placement_recommendations_table.sql	Збереження запропонованих дій
Попит і моделі	V5, V10	Історія замовлень і версії прогнозних моделей
Оптимізація	V7, V8, V11, V12	Оцінки, плани, резервування ресурсів
Події	V13__create_outbox_events.sql	Outbox для доменних подій

Приклад логіки міграції для збереження рекомендацій щодо розміщення наведено в лістингу 4.9.

Лістинг 4.9 – Приклад логіки міграції для рекомендацій

```
CREATE TABLE placement_recommendations (
    id BIGSERIAL PRIMARY KEY,
    warehouse_id BIGINT NOT NULL,
    container_id BIGINT NOT NULL,
    recommendation_type VARCHAR(32) NOT NULL,
    status VARCHAR(32) NOT NULL,
    target_storage_place_id BIGINT,
    target_container_id BIGINT,
    created_at TIMESTAMP NOT NULL
);
```

Цей фрагмент демонструє, що рекомендація зберігається як окрема сутність. Вона містить тип дії, статус і посилання на цільове місце або цільову палету, тому її можна переглянути, підтвердити або відхилити.

4.9.2 Ручні сценарії перевірки вебінтерфейсу

Окрім автоматизованих тестів, для веб-застосунку важливими є ручні сценарії перевірки. Вони дозволяють оцінити не лише коректність сервісного рівня, а й зручність послідовності дій у браузері.

Першим сценарієм є створення складу, товарів і палет. Він перевіряє, чи може адміністратор підготувати початкові дані без прямого доступу до бази даних. Другим сценарієм є отримання рекомендації та підтвердження місця. Він перевіряє основну цінність системи.

Третім сценарієм є імпорт історії попиту та навчання моделі. Він показує, чи може система перейти від звичайного складського обліку до використання аналітики. Четвертим сценарієм є виконання оптимізаційного плану оператором.

П'ятий сценарій пов'язаний з аудитом: після виконання операцій адміністратор повинен бачити історію переміщень і події. Це підтверджує, що система не втрачає операційну інформацію.

Ручні сценарії перевірки системи наведено в таблиці 4.8.

Таблиця 4.8 – Ручні сценарії перевірки системи

Сценарій	Дії користувача	Очікуваний результат
Підготовка складу	Створити склад, додати товари, прийняти палети	У системі є місця, товари й палети
Розміщення	Отримати рекомендацію, підтвердити місце	Палета переходить у стан stored
Об'єднання	Прийняти палету того самого товару	Система пропонує merge, якщо це можливо
Прогнозування	Імпортувати історію, навчити модель	З'являється активна або відхилена модель
Оптимізація	Запустити assessment, створити план	Система формує кроки переміщення
Аудит	Відкрити історію та події	Операції відображаються в журналі

Форму створення нового товару, яка використовується в ручному сценарії підготовки довідника, наведено на рисунку 4.19.

Рисунок 4.19 – Форма створення нового товару

4.9.3 Підготовка до інтеграції зі сканером штрихкодів

Хоча фізичний сканер у межах роботи не використовувався, система вже має логіку, яку можна адаптувати до сканування. У більшості простих складських сценаріїв сканер штрихкодів працює як HID-пристрій і передає відсканований код у фокусоване поле введення так само, як клавіатура.

Це означає, що на першому етапі інтеграції не обов'язково змінювати серверну частину. Достатньо забезпечити, щоб поле номера палети або коду місця було активним, а після сканування виконувалася та сама дія, що й після ручного введення. Такий підхід зменшує складність впровадження.

У майбутньому можна перейти до більш складного варіанта: мобільного або термінального інтерфейсу, який буде явно обробляти події сканера, показувати оператору наступний крок і блокувати підтвердження, якщо відсканований код не відповідає очікуваному місцю.

Найважливішим при інтеграції сканера є підтвердження фізичної дії. Система може рекомендувати місце, але повинна зафіксувати саме фактичне сканування палети та місця. Це дозволяє зменшити ризик розбіжності між цифровою моделлю і реальним складом.

Отже, поточне ручне введення є прийнятною умовністю для демонстраційної версії, але логіка процесу вже наближена до майбутнього сканування: спочатку ідентифікується палета, потім обирається або підтверджується місце, після чого створюється запис руху та подія аудиту.

4.9.4 Підсумок програмної реалізації

Програмна реалізація підтверджує, що розроблена система охоплює не один окремий алгоритм, а повний набір складських процесів. У ній реалізовано довідники, структуру складу, палети, рекомендації, переміщення, попит, прогностні моделі, оптимізаційні плани, аудит, ролі користувачів і допоміжну інфраструктуру.

Найбільш важливою особливістю реалізації є зв'язок між бізнес-логікою та станом складу. Рекомендація не існує окремо від палети, місця зберігання, обмежень ваги й висоти, історії попиту та маршруту. Саме ця зв'язність робить систему прикладною, а не лише демонстраційною.

Реалізований вебінтерфейс дозволяє показати роботу системи без спеціального обладнання. Однак послідовність дій побудована так, щоб у майбутньому її можна було перенести на сценарій зі сканером: оператор ідентифікує палету, отримує рекомендацію, підтверджує фізичну локацію та залишає цифровий слід операції.

Таким чином, програмна частина роботи створює основу для подальшого розвитку: підключення сканерів, мобільного інтерфейсу, зовнішніх систем обліку, складніших моделей прогнозування та більш гнучкої оптимізації складської топології.

4.9.5 Документаційна сторінка проєкту та API

Окрім основного адміністративного й операторського інтерфейсу, для проєкту підготовлено документаційну сторінку. Вона виконує роль короткого огляду системи: пояснює призначення застосунку, основні можливості, технологічний стек і сценарії роботи.

Наявність такої сторінки є корисною для захисту роботи та подальшого розвитку проєкту. Вона дозволяє швидко показати, що застосунок не обмежується окремим алгоритмом, а має цілісний набір модулів: склад, товари, палети, попит, оптимізацію, маршрутизацію, події та API.

API-документація важлива для майбутніх інтеграцій. Якщо система буде підключатися до мобільного клієнта, ERP або зовнішньої WMS, саме опис API стане основою для узгодження форматів запитів, відповідей і помилок.

Документаційну сторінку проєкту наведено на рисунку 4.20.

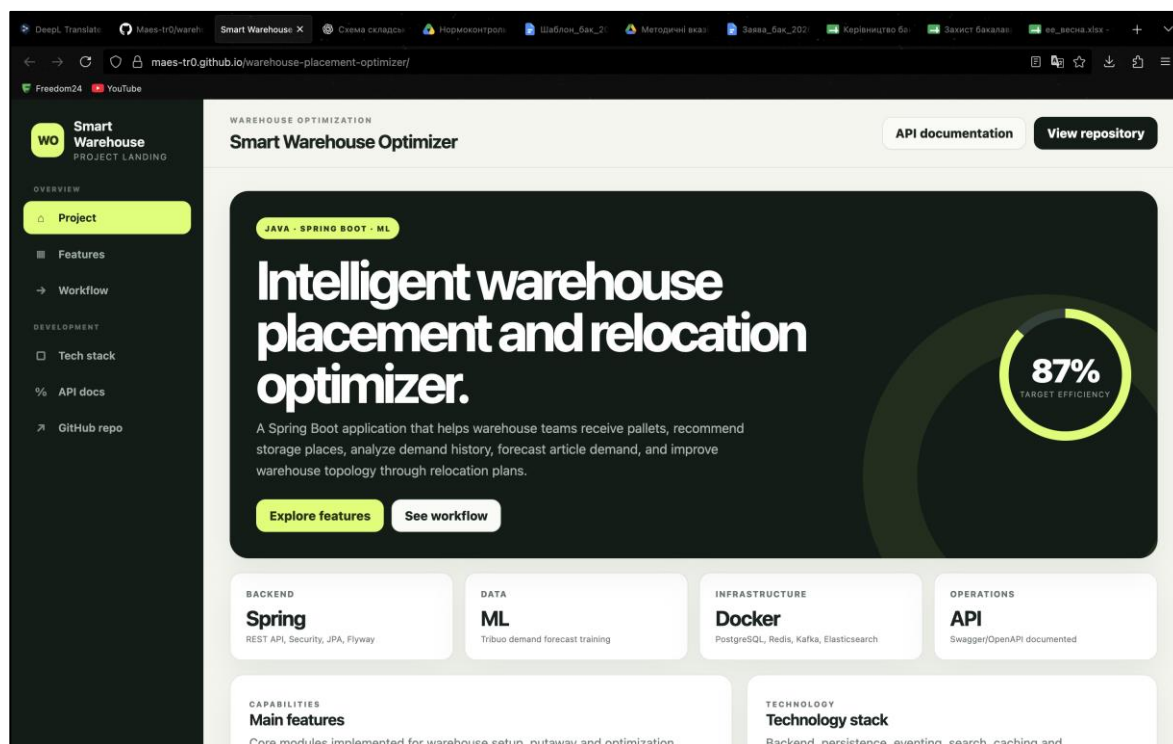


Рисунок 4.20 – Документаційна сторінка проєкту Smart Warehouse Optimizer

Окрема документаційна сторінка також допомагає відокремити демонстраційний опис від робочого інтерфейсу. Адміністративний портал призначений для виконання операцій, а документаційна сторінка – для пояснення архітектури, можливостей і меж реалізації.

4.9.6 Ризики впровадження та способи їх зменшення

Під час впровадження складської інформаційної системи виникають не лише технічні, а й організаційні ризики. Навіть якісний алгоритм не дасть очікуваного ефекту, якщо дані про склад, товари та палети будуть неповними або некоректними.

Першим ризиком є неточність початкового опису складу. Якщо коди місць, висота рівнів або вантажні обмеження не відповідають фактичному складу, рекомендації можуть бути неправильними. Зменшити цей ризик

можна через поетапне введення складу, перевірку місць і тестову інвентаризацію.

Другим ризиком є помилки операторів під час ручного введення. До підключення сканера номери палет і коди місць вводяться вручну, тому можливі помилки в символах. Для зменшення ризику потрібні маски введення, перевірка існування коду та підтвердження операції.

Третім ризиком є недостатність історії попиту. Якщо даних мало або вони нерівномірні, модель прогнозування може не покращити базовий підхід. У такому випадку система повинна використовувати прогноз лише як допоміжний фактор і не відкидати класичні обмеження.

Основні ризики впровадження системи та способи їх зменшення наведено в таблиці 4.9.

Таблиця 4.9 – Ризики впровадження та способи їх зменшення

Ризик	Можливий наслідок	Спосіб зменшення
Неточна топологія складу	Помилкові маршрути та рекомендації	Перевірка місць, контрольні маршрути, інвентаризація
Помилки ручного введення	Невірна палета або локація	Маски введення, довідники, майбутнє сканування
Недостатня історія попиту	Слабка якість прогнозу	Накопичення даних і baseline-захист моделі
Зміна процесів роботи	Опір користувачів або неправильне використання	Навчання операторів і поступове впровадження
Недоступність інфраструктури	Затримка аудиту або подій	Outbox, повторна доставка, health checks

Отже, визначені ризики не є критичними для демонстраційної версії системи, однак їх необхідно враховувати під час подальшого впровадження в реальному складському середовищі.

ВИСНОВКИ

У межах кваліфікаційної роботи було розроблено інтелектуальну вебсистему оптимізації розміщення товарів на складі. Розроблене програмне рішення спрямоване на автоматизацію складських процесів, підтримку прийняття рішень під час розміщення палет і підвищення ефективності використання складського простору.

Було проаналізовано предметну галузь складської логістики та визначено основні проблеми, пов'язані з ручним вибором місць зберігання, нераціональним розміщенням товарів, збільшенням часу переміщення операторів і недостатнім урахуванням історії попиту.

У роботі обґрунтовано вибір технологій, методів і моделей. Для серверної частини обрано Java та Spring Boot, для зберігання даних – PostgreSQL, для доступу до даних – Spring Data JPA, для керування міграціями – Flyway, для прогнозування попиту – Tribuo, а для маршрутизації – графову модель складу й алгоритм Дейкстри.

Реалізовано функціональність для створення структури складу, ведення довідника товарів, приймання палет, формування рекомендацій щодо розміщення, аналізу попиту, навчання прогнозної моделі, оцінювання ефективності складу, планування переміщень і ведення історії операцій.

Окрему увагу приділено умовностям реалізації. Через відсутність фізичного сканера штрихкодів у поточній версії системи сценарій сканування імітується ручним введенням номерів палет і кодів місць зберігання. Водночас бізнес-процес побудовано так, щоб у майбутньому ручне введення можна було замінити апаратним скануванням без суттєвої зміни логіки системи. Подальший розвиток системи може передбачати інтеграцію зі сканерами штрихкодів, мобільний інтерфейс для операторів, підтримку декількох складів, інтеграцію з ERP/WMS-системами, урахування тимчасово заблокованих проходів і використання складніших моделей прогнозування та оптимізації.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Network community detection using modified modularity criterion / V. Shergin et al. *Eastern-European journal of enterprise technologies*. 2024. Vol. 6, no. 4 (132). P. 6–13. URL: <https://doi.org/10.15587/1729-4061.2024.318452> (date of access: 17.06.2026).
2. Frazelle E. World-class warehousing and material handling. New York: McGraw-Hill, 2002. 242 p.
3. Mathematical models to optimize management of time and space in a warehouse. *Warehouse Science*. URL: <https://www.warehouse-science.com/> (date of access: 17.06.2026).
4. Warehouse design and control: framework and literature review / B. Rouwenhorst et al. *European journal of operational research*. 2000. Vol. 122, no. 3. P. 515–533.
5. Gu J., Goetschalckx M., McGinnis L. F. Research on warehouse operation: a comprehensive review. *European journal of operational research*. 2007. Vol. 177, No. 1. P. 1–21.
6. Gu J., Goetschalckx M., McGinnis L. F. Research on warehouse design and performance evaluation: a comprehensive review. *European journal of operational research*. 2010. Vol. 203, No. 3. P. 539–549.
7. De Koster R., Le-Duc T., Roodbergen K. J. Design and control of warehouse order picking: a literature review. *European journal of operational research*. 2007. Vol. 182, No. 2. P. 481–501.
8. Petersen C. G., Aase G. A comparison of picking, storage, and routing policies in manual order picking. *International journal of production economics*. 2004. Vol. 92, No. 1. P. 11–19.
9. Hyndman R. J., Athanasopoulos G. Forecasting: principles and practice. 3rd ed. Melbourne: OTexts, 2021. URL: <https://otexts.com/fpp3/> (date of access: 17.06.2026).

10. Breiman L., Friedman J. H., Olshen R. A., Stone C. J. Classification and regression trees. Boca Raton: Chapman and Hall/CRC, 1984. 368 p.
11. Java Platform, Standard Edition & Java Development Kit Version 26 API Specification. Oracle. URL: <https://docs.oracle.com/en/java/javase/26/docs/api/index.html> (date of access: 17.06.2026).
12. Spring Boot reference documentation. Spring. URL: <https://docs.spring.io/spring-boot/reference/index.html> (date of access: 17.06.2026).
13. PostgreSQL Documentation. *PostgreSQL global development group*. URL: <https://www.postgresql.org/docs/current/index.html> (date of access: 17.06.2026).
14. Spring Data JPA reference documentation. Spring. URL: <https://docs.spring.io/spring-data/jpa/reference/> (date of access: 17.06.2026).
15. Flyway Documentation. Redgate. URL: <https://documentation.red-gate.com/fd> (date of access: 17.06.2026).
16. Dijkstra E. W. A note on two problems in connexion with graphs. *Numerische Mathemata*. 1959. Vol. 1. P. 269–271.
17. Cormen T. H., Leiserson C. E., Rivest R. L., Stein C. Introduction to Algorithms. 4th ed. Cambridge: MIT Press, 2022. 1312 p.
18. Tribuo Documentation. Oracle Labs. URL: <https://tribuo.org/learn/4.3/docs/> (date of access: 17.06.2026).
19. Pocock A. Tribuo: machine learning with provenance in java. 2021. URL: <https://doi.org/10.48550/arXiv.2110.03022> (date of access: 17.06.2026).
20. Redis documentation. *Redis Ltd*. URL: <https://redis.io/docs/latest/> (date of access: 17.06.2026).
21. Apache software foundation. *Apache Kafka documentation*. URL: <https://kafka.apache.org/documentation/> (date of access: 17.06.2026).
22. Elasticsearch B.V. *Elastic docs*. URL: <https://www.elastic.co/docs> (date of access: 17.06.2026).

23. Spring security reference documentation. *Spring*. URL: <https://docs.spring.io/spring-security/reference/index.html> (date of access: 17.06.2026).

24. Docker compose documentation. *Docker Inc.* URL: <https://docs.docker.com/compose/> (date of access: 17.06.2026).

25. Maven documentation. *Apache software foundation*. URL: <https://maven.apache.org/guides/index.html> (date of access: 17.06.2026).