НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
Факультет інформаційних технологій

ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ
Завідувач кафедри
Комп'ютерних наук
Голуб Б.Л. "__" _____20 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА
на тему
Інформаційна система організації логістичного забезпечення військових потреб

Спеціальність F3 – «Комп'ютерні науки»

Гарант освітньої програми
д.е.н., професор Руденський Р.А.

Керівник бакалаврської кваліфікаційної роботи
к.т.н, доцент Даков Сергій Юрійович
(науковий ступінь та вчене звання) (підпис) (ПІБ)

Виконав Лобанов Б.І
(підпис) (ПІБ студента)

КИЇВ – 2025

НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри
Комп'ютерних наук
_____ Голуб Б.Л.

"__" _____20 р.

.

З А В Д А Н Н Я
на виконання бакалаврської кваліфікаційної роботи студенту
Лобанову Богдану Ігоровичу
(прізвище, ім'я, по батькові)

Спеціальність F3 – «Комп'ютерні науки»
Тема бакалаврської кваліфікаційної роботи
Інформаційна система організації логістичного забезпечення військових потреб
затверджена наказом ректора НУБіП України від 11.03.2025 №314с
Термін подання завершеної роботи на кафедру 2025. 05. 24
(рік, місяць, число)
Вихідні дані до бакалаврської кваліфікаційної роботи:
Мультимедійні файли, інтерфейсу програми

Перелік питань, які потрібно розробити:
1. Теоретичні аспекти розробки
2. Аналіз предметної області
3. Практична реалізація
4. Впровадження та експлуатація системи

Дата видачі завдання "11" 03 2025р.
Керівник бакалаврської кваліфікацій
к.т.н, доцент Даков Сергій Юрійович
(науковий ступінь та вчене звання) (підпис) (ПІБ)

Завдання прийняв до виконання _____ Лобанов Б.І
(підпис) (ПІБ студента)

Календарний план

№ з/п	Назва етапів виконання бакалаврської кваліфікаційної роботи	Строк виконання етапів бакалаврської кваліфікаційної роботи	Примітка
1.	Вибір та затвердження тем	Березень	Виконано
2.	Виконання розділу 1 Теоретичні аспекти розробки інформаційної системи логістичного забезпечення військових потреб	Березень	Виконано
3	Виконання розділу 2 Аналіз предметної області та визначення інструментів для вирішення поставленої задачі	Квітень	Виконано
4	Виконання розділу 3 Практична реалізація інформаційної системи	Квітень	Виконано
5	Виконання розділу 4 Впровадження та експлуатація системи	Травень	Виконано
6	Написання пояснювальної записки	Травень	Виконано

Студент Л о б а н о в Б . І _____
(підпис) (прізвище та ініціали)

Керівник бакалаврської кваліфікаційної роботи
Даков Сергій Юрійович
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Бакалаврська кваліфікаційна робота присвячена розробці інформаційної системи організації логістичного забезпечення військових потреб. Робота виконана на сторінках, містить 19 рисунків, 2 таблиці, 3 додатків та список використаних джерел з 37 найменувань.

Актуальність дослідження обумовлена критичною важливістю ефективної організації логістичного забезпечення збройних сил в умовах сучасних військових конфліктів. Традиційні методи ручного планування та контролю постачань не здатні забезпечити необхідну швидкість реагування на зміни в оперативній обстановці.

Метою роботи є розробка та реалізація інформаційної системи організації логістичного забезпечення військових потреб, яка забезпечить автоматизацію процесів планування, контролю та управління матеріально-технічними ресурсами збройних сил.

Об'єктом дослідження є процеси організації логістичного забезпечення військових потреб. Предметом дослідження є методи, алгоритми та технології створення інформаційних систем управління логістичними процесами у військовій сфері.

У роботі проведено аналіз теоретичних аспектів військової логістики, досліджено існуючі технології та визначено оптимальні інструменти для розробки системи. Розроблено архітектуру системи з використанням Python/Django, PostgreSQL, та контейнеризації Docker. Створено повнофункціональну веб-систему з модулями управління замовленнями, користувачами, звітності та комунікацій.

Практична значущість роботи полягає у можливості реального покращення ефективності логістичного забезпечення військових підрозділів через автоматизацію процесів та оптимізацію управління ресурсами.

ANNOTATION

The bachelor's thesis is devoted to the development of an information system for organising logistics support for military needs. The work is executed on 19 pages, 4 tables, 5 appendices and a list of 37 references.

The relevance of the study is due to the critical importance of effective organisation of logistics support for the armed forces in the context of modern military conflicts. Traditional methods of manual planning and control of supplies are not able to provide the necessary speed of response to changes in the operational situation.

The aim of the study is to develop and implement an information system for organising logistics support for military needs, which will automate the processes of planning, controlling and managing material and technical resources of the armed forces.

The object of research is the processes of organising logistics support for military needs. The subject of the study is the methods, algorithms and technologies for creating information systems for managing logistics processes in the military sphere.

The paper analyses the theoretical aspects of military logistics, examines existing technologies and identifies the optimal tools for system development. The system architecture is developed using Python/Django, PostgreSQL, and Docker containerisation. A full-featured web-based system with modules for order management, user management, reporting, and communications was created.

The practical significance of the work lies in the possibility of real improvement of the efficiency of logistics support of military units through process automation and optimisation of resource management.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API – Application Programming Interface (інтерфейс програмного додатку)

БД – база даних

ВС – військова система

ГІС – геоінформаційна система

ЗС – збройні сили

ІС – інформаційна система

ІТ – інформаційні технології

КІБЕРЗАХИСТ – комплекс заходів захисту від кіберзагроз

МТЗ – матеріально-технічні засоби

НАТО – Північноатлантичний альянс

ОС – операційна система

ПЗ – програмне забезпечення

ППЗ – прикладне програмне забезпечення

СУБД – система управління базами даних

ТЗ – технічне завдання

УМЛ – уніфікована мова моделювання (UML)

ACID – Atomicity, Consistency, Isolation, Durability

CI/CD – Continuous Integration/Continuous Deployment

CSS – Cascading Style Sheets

DFD – Data Flow Diagram (діаграма потоків даних)

DRF – Django REST Framework

ERD – Entity-Relationship Diagram

GPS – Global Positioning System

HTML – HyperText Markup Language

HTTP/HTTPS – HyperText Transfer Protocol / Secure

IoT – Internet of Things (інтернет речей)

JSON – JavaScript Object Notation

LDAP – Lightweight Directory Access Protocol

OWASP – Open Web Application Security Project

RBAC – Role-Based Access Control

REST – Representational State Transfer

RFID – Radio Frequency Identification

SSL/TLS – Secure Sockets Layer / Transport Layer Security

SQL – Structured Query Language

UML – Unified Modeling Language

WCAG – Web Content Accessibility Guidelines

XML – eXtensible Markup Language

ВСТУП

Актуальність теми дослідження. В умовах сучасних військових конфліктів та зростаючих загроз національній безпеці ефективна організація логістичного забезпечення збройних сил набуває критично важливого значення для забезпечення обороноздатності держави. Складність та масштабність сучасних військових операцій вимагають координації постачання десятків тисяч найменувань матеріально-технічних засобів у сотні географічно розосереджених локацій, що неможливо ефективно здійснювати традиційними методами ручного планування та контролю.

Досвід збройних конфліктів останніх років, зокрема в Україні, продемонстрував, що своєчасне та точне логістичне забезпечення часто є вирішальним фактором успіху військових операцій. Затримки в постачанні боєприпасів, паливно-мастильних матеріалів, медичних засобів та іншого критично важливого обладнання можуть мати катастрофічні наслідки для виконання бойових завдань та життя військовослужбовців. Водночас, надлишкові запаси призводять до нераціонального використання обмежених оборонних бюджетів та створюють додаткові ризики для безпеки зберігання військових матеріалів.

Інформаційні технології пропонують потужні рішення для автоматизації та оптимізації логістичних процесів, що дозволяють значно підвищити ефективність управління матеріальними потоками, скоротити витрати та забезпечити швидке реагування на зміни в оперативній обстановці. Провідні армії світу активно впроваджують сучасні системи управління логістикою, що підтверджує стратегічну важливість цифровізації військової сфери.

Зв'язок роботи з науковими програмами, планами, темами. Дослідження виконується в рамках актуальних напрямків розвитку інформаційних технологій у військовій сфері та відповідає Стратегії кібербезпеки України, Доктрині інформаційної безпеки України та планам

цифрової трансформації Збройних Сил України. Робота узгоджується з пріоритетами розвитку оборонно-промислового комплексу щодо впровадження сучасних інформаційних систем управління та планами інтеграції України до стандартів НАТО в галузі логістичного забезпечення.

Мета і завдання дослідження. Метою роботи є розробка та реалізація інформаційної системи організації логістичного забезпечення військових потреб, яка забезпечить автоматизацію процесів планування, контролю та управління матеріально-технічними ресурсами збройних сил з дотриманням високих стандартів безпеки та надійності.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

– проаналізувати теоретичні аспекти організації логістичного забезпечення військових потреб та існуючі технологічні рішення;

– дослідити вимоги до сучасних систем військової логістики та визначити оптимальні технології для їх реалізації;

– спроектувати архітектуру інформаційної системи з використанням сучасних методологій моделювання предметної області;

– розробити інформаційне забезпечення системи з оптимізованою структурою бази даних;

– реалізувати прикладне програмне забезпечення з необхідною функціональністю для управління логістичними процесами;

– провести комплексне тестування системи та розробити рекомендації щодо її впровадження та експлуатації.

Об'єкт дослідження – процеси організації логістичного забезпечення військових потреб, включаючи планування потреб у матеріально-технічних засобах, формування та обробку замовлень, управління постачальниками, контроль виконання поставок та аналітичну звітність.

Предмет дослідження – методи, алгоритми та технології створення інформаційних систем управління логістичними процесами у військовій сфері з урахуванням специфічних вимог до безпеки, надійності та продуктивності.

Методи дослідження. У роботі використовуються методи системного аналізу для дослідження предметної області, методи об'єктно-орієнтованого проектування для розробки архітектури системи, методи реляційного моделювання для проектування бази даних, методи програмної інженерії для реалізації системи та методи тестування програмного забезпечення для верифікації функціональності та якості розробленого рішення.

Наукова новизна отриманих результатів полягає в адаптації сучасних інформаційних технологій до специфічних потреб військової логістики з урахуванням вимог безпеки, надійності та інтеграції з існуючими військовими інформаційними системами. Розроблено комплексний підхід до автоматизації логістичних процесів, що поєднує ефективність сучасних веб-технологій з жорсткими вимогами військових стандартів.

Практична значущість роботи визначається можливістю використання розробленої системи для реального покращення ефективності логістичного забезпечення військових підрозділів. Впровадження системи дозволить скоротити час обробки заявок на постачання, підвищити точність планування потреб, оптимізувати рівні запасів та забезпечити повну відстежуваність матеріальних потоків, що критично важливо для підтримання боєготовності збройних сил.

Апробація результатів дослідження. Основні результати роботи представлялися на науково-практичних конференціях студентів та молодих вчених НУБіП України, обговорювалися на засіданнях кафедри комп'ютерних наук та отримали позитивну оцінку фахівців у галузі військової логістики.

Публікації. За результатами дослідження опубліковано тези доповіді на науково-практичній конференції "Інформаційні технології в аграрній освіті, науці та виробництві" (НУБіП України, 2025 р.).

Структура та обсяг роботи. Бакалаврська кваліфікаційна робота складається з вступу, чотирьох розділів, висновків, списку використаних джерел з 45 найменувань та додатків. Загальний обсяг роботи становить 68 сторінок,

включаючи 12 рисунків, 8 таблиць та 6 додатків з технічною документацією та лістингами програмного коду.

РОЗДІЛ 1. ТЕОРЕТИЧНІ АСПЕКТИ РОЗРОБКИ ІНФОРМАЦІЙНОЇ СИСТЕМИ ЛОГІСТИЧНОГО ЗАБЕЗПЕЧЕННЯ ВІЙСЬКОВИХ ПОТРЕБ

1.1 Визначення галузі застосування інформаційних систем логістичного забезпечення

Інформаційні системи логістичного забезпечення є спеціалізованими програмними комплексами, які призначені для ефективного управління потоками матеріальних ресурсів у військовій сфері. Вони представляють собою інтегровані рішення, що охоплюють весь життєвий цикл матеріально-технічного забезпечення військових підрозділів від планування потреб до доставки ресурсів кінцевим споживачам. Такі системи забезпечують комплексну автоматизацію процесів обліку, планування, контролю та аналізу логістичних операцій. Військова логістика суттєво відрізняється від комерційної через особливі вимоги до швидкості реагування та критичної важливості своєчасної доставки. Інформаційні системи у цій галузі мають враховувати специфіку військових операцій, що характеризуються високою динамічністю, непередбачуваністю та підвищеними ризиками. Вони повинні забезпечувати ефективне функціонування логістичних процесів навіть у складних польових умовах з обмеженою комунікаційною інфраструктурою.

На стратегічному рівні інформаційні системи логістичного забезпечення застосовуються для довгострокового планування матеріально-технічних ресурсів армії. Вони дозволяють проводити комплексний аналіз потреб збройних сил, планувати закупівлі, розраховувати необхідні обсяги стратегічних запасів та оптимізувати їх розміщення. Такі системи інтегруються з державними системами оборонного планування та бюджетування для забезпечення узгодженості ресурсного забезпечення з оборонною стратегією країни [1].

Тактичний рівень застосування цих систем охоплює середньострокове планування та розподіл ресурсів між військовими підрозділами. Інформаційні

системи на цьому рівні забезпечують координацію поставок між різними логістичними вузлами, управління запасами на регіональних складах та планування транспортних маршрутів. Вони дозволяють гнучко реагувати на зміни оперативної обстановки та перерозподіляти ресурси відповідно до пріоритетів військового командування.

На оперативному рівні інформаційні системи забезпечують щоденне функціонування логістичних процесів у військових частинах. Вони автоматизують облік матеріальних цінностей, формування та обробку заявок на постачання, контроль виконання поставок та документообіг. Системи цього рівня зазвичай мають модульну структуру та включають компоненти управління складом, транспортом, обліку та звітності. Важливою галуззю застосування є інтеграція з іншими військовими інформаційними системами для забезпечення комплексного підходу до управління військами. Логістичні системи взаємодіють з системами управління особовим складом, бойовими операціями, розвідувальними системами та системами фінансового забезпечення. Така інтеграція дозволяє синхронізувати логістичні процеси з оперативними планами та забезпечити своєчасне постачання необхідних ресурсів. Специфічною галуззю застосування є системи управління ланцюгами поставок для критично важливих матеріальних засобів. Ці системи забезпечують відстеження переміщення боєприпасів, паливно-мастильних матеріалів, запасних частин та медичних засобів у режимі реального часу. Вони надають повну видимість логістичного ланцюга від складів до передових підрозділів та дозволяють оперативно реагувати на зміни у потребах або порушення в ланцюзі постачання.

У сучасних арміях світу інформаційні системи логістичного забезпечення є невід'ємною частиною загальної системи управління збройними силами. Армія США використовує систему GCSS-Army, яка забезпечує комплексну автоматизацію логістичних процесів та інтеграцію з іншими системами командування та контролю. Збройні сили країн НАТО впроваджують стандартизовані системи, що забезпечують сумісність та ефективну взаємодію при проведенні спільних операцій. В Україні розвиток інформаційних систем

логістичного забезпечення набув особливої актуальності з початком військових дій. Відбувається активне впровадження автоматизованих систем управління матеріально-технічним забезпеченням, які адаптовані до специфічних умов ведення гібридної війни. Важливим напрямком є інтеграція українських систем із системами країн НАТО для забезпечення ефективної взаємодії та сумісності логістичних процесів.

Окремою галуззю застосування є системи підтримки прийняття рішень у сфері військової логістики. Вони використовують методи математичного моделювання, оптимізації та прогнозування для вибору найбільш ефективних логістичних рішень. Такі системи дозволяють проводити аналіз різних сценаріїв, оцінювати ризики та обирати оптимальні варіанти забезпечення військових операцій з урахуванням наявних ресурсів та обмежень [2]. Таким чином, галузь застосування інформаційних систем логістичного забезпечення у військовій сфері охоплює широкий спектр завдань від стратегічного планування до оперативного управління матеріальними потоками. Ці системи забезпечують підвищення ефективності логістичних процесів, скорочення часу реагування на зміни в оперативній обстановці та оптимізацію використання ресурсів. Впровадження сучасних інформаційних технологій у військову логістику є ключовим фактором забезпечення високої боєздатності збройних сил у сучасних умовах.

1.2 Поняття логістичного забезпечення у військовій сфері

Військова логістика представляє собою комплексну систему планування, організації та контролю процесів матеріально-технічного забезпечення збройних сил. Вона охоплює широкий спектр діяльності, пов'язаної з постачанням, зберіганням, транспортуванням, розподілом та обслуговуванням матеріальних засобів, необхідних для проведення військових операцій. Ефективна військова логістика забезпечує своєчасну доставку необхідних ресурсів у потрібне місце в необхідній кількості та відповідної якості.

Історично поняття військової логістики еволюціонувало від простого забезпечення військ продовольством та амуніцією до складної інтегрованої системи управління матеріальними потоками. Термін "логістика" має військове походження і використовувався ще у давньоримській армії для позначення процесів забезпечення військ. Наполеон Бонапарт вважав логістику одним із трьох найважливіших компонентів військової стратегії, поряд із стратегією та тактикою. Основними складовими військової логістики є постачання, транспортування, зберігання, технічне обслуговування та утилізація. Постачання включає визначення потреб, закупівлю, приймання та розподіл матеріальних засобів. Транспортування забезпечує переміщення ресурсів від місць зберігання до споживачів. Зберігання охоплює розміщення запасів, їх облік та підтримання в належному стані. Технічне обслуговування забезпечує працездатність техніки та обладнання протягом їх життєвого циклу.

Військова логістика базується на ключових принципах, які визначають її ефективність та надійність. Принцип своєчасності вимагає доставки необхідних ресурсів точно у визначений термін. Принцип достатності передбачає забезпечення військових підрозділів необхідною кількістю ресурсів без надлишкових запасів. Принцип гнучкості дозволяє адаптувати логістичні процеси до змін оперативної обстановки. Принцип прозорості забезпечує повну видимість ресурсів та матеріальних потоків. Ланцюги постачання у військовій логістиці мають специфічну структуру, яка відрізняється від комерційних ланцюгів. Вони зазвичай включають стратегічні склади, регіональні бази постачання, польові логістичні центри та передові пункти забезпечення. Ланцюг постачання має ієрархічну структуру з чітким розподілом функцій та відповідальності між різними рівнями [3]. На вищому рівні здійснюється централізоване планування та закупівл, а на нижчих рівнях – розподіл та безпосереднє забезпечення військових підрозділів.

Військова логістика функціонує на трьох основних рівнях, які відповідають рівням ведення військових дій. Стратегічний рівень охоплює національну інфраструктуру постачання, включаючи виробництво, закупівлі та

формування стратегічних запасів. Оперативний рівень забезпечує логістичну підтримку військових операцій на театрі військових дій. Тактичний рівень зосереджений на безпосередньому забезпеченні бойових підрозділів під час виконання бойових завдань.

Основними функціями військової логістики є планування, закупівлі, розподіл та утилізація. Планування визначає потреби в матеріальних засобах на основі оперативних планів та чисельності військ. Закупівлі забезпечують придбання необхідних ресурсів у постачальників. Розподіл включає доставку та видачу матеріальних засобів військовим підрозділам. Утилізація забезпечує ефективне використання або знищення застарілих або надлишкових матеріальних засобів.

Логістичні цикли у військовій сфері мають особливу структуру, яка відповідає циклам військових операцій. Цикл планування визначає потреби та формує замовлення. Цикл постачання охоплює закупівлю та доставку ресурсів. Цикл споживання включає використання ресурсів під час операцій. Цикл відновлення забезпечує ремонт, поповнення запасів та утилізацію. Ці цикли взаємопов'язані та формують безперервний процес логістичного забезпечення. Військова логістика суттєво відрізняється від цивільної за рядом ключових параметрів. Військова логістика має працювати в екстремальних умовах з високим рівнем невизначеності та ризиків. Вона вимагає підвищеної надійності та стійкості до пошкоджень. Військова логістика має забезпечувати швидке переміщення великих обсягів вантажів на значні відстані, часто в умовах відсутності розвиненої інфраструктури. Крім того, вона має специфічні вимоги до безпеки та захисту інформації [4].

Сучасні тенденції розвитку військової логістики включають інтеграцію з цивільними логістичними системами, впровадження автоматизованих систем управління та використання передових технологій. Концепція "об'єднаної логістики" передбачає координацію логістичних систем різних видів збройних сил для підвищення ефективності використання ресурсів. Підхід "точно вчасно"

адаптується до військових умов для скорочення обсягів запасів та підвищення гнучкості системи постачання.

Міжнародна стандартизація у сфері військової логістики відіграє важливу роль у забезпеченні взаємодії між арміями різних країн. Стандарти НАТО, такі як STANAG 2406 (логістичні процедури), STANAG 2290 (ідентифікація матеріальних засобів) та STANAG 2961 (процедури забезпечення класами постачання), визначають спільні підходи до організації логістичного забезпечення. Ці стандарти забезпечують сумісність логістичних систем та ефективну взаємодію під час проведення спільних операцій.

1.3 Особливості організації логістики для задоволення військових потреб

Військова логістика характеризується рядом унікальних особливостей, які визначають специфіку її організації та функціонування. Вона повинна забезпечувати постачання військ в умовах високої невизначеності, обмежених ресурсів та постійних змін оперативної обстановки. Особливості військової логістики обумовлені характером військових операцій, які вимагають надійного та безперервного забезпечення в будь-яких умовах та на будь-якій території.

Критичність часових параметрів є однією з ключових особливостей військової логістики. Затримки в поставках можуть мати катастрофічні наслідки для виконання бойових завдань та безпеки військовослужбовців. Військова логістика оперує поняттям "критичного часу постачання", який визначає максимально допустимий період відсутності певних ресурсів. Для деяких категорій матеріальних засобів, таких як боєприпаси, паливо та медичні матеріали, цей час може бути надзвичайно коротким. Військова логістика функціонує в умовах підвищеної невизначеності та постійних змін. Бойова обстановка може швидко змінюватися, вимагаючи термінової перебудови логістичних процесів. Потреби військ можуть значно коливатися залежно від інтенсивності бойових дій [5]. Логістичні маршрути та вузли можуть стати

недоступними через дії противника. Ця невизначеність вимагає створення гнучкої логістичної системи з високим рівнем адаптивності та резервування.

Географічний розподіл військових підрозділів створює додаткові виклики для логістичного забезпечення. Військові частини можуть бути розташовані на значних відстанях одна від одної та від основних логістичних баз. Логістична система повинна забезпечувати постачання на великі відстані, часто через складну місцевість або в умовах відсутності розвиненої транспортної інфраструктури. Це вимагає ретельного планування маршрутів, використання різних видів транспорту та створення проміжних логістичних вузлів.

Безпека логістичних операцій є критично важливим аспектом військової логістики. Логістичні конвої та склади часто стають першочерговими цілями для противника. Перехоплення поставок може не лише позбавити війська необхідних ресурсів, але й надати противнику цінну розвідувальну інформацію. Тому військова логістика потребує комплексної системи захисту, яка включає фізичну охорону, маскування, розосередження запасів та захист інформації про логістичні операції. Формування та управління військовими запасами має специфічні особливості. Військові запаси поділяються на стратегічні, оперативні та тактичні. Стратегічні запаси створюються на тривалий період та розміщуються в захищених сховищах. Оперативні запаси забезпечують проведення військових операцій протягом певного періоду. Тактичні запаси знаходяться безпосередньо у військових підрозділах. Особливу увагу приділяють ротації запасів для запобігання їх застарінню, особливо для боєприпасів, медикаментів та продовольства.

Визначення пріоритетності поставок є важливою особливістю військової логістики в умовах обмежених ресурсів. Не всі потреби можуть бути задоволені одночасно, тому необхідна чітка система пріоритетів. Зазвичай найвищий пріоритет мають боєприпаси, паливо та медичні матеріали. Пріоритети можуть змінюватися залежно від етапу операції та оперативної обстановки [6]. Система пріоритетів повинна бути гнучкою та враховувати критичність різних типів ресурсів для виконання бойових завдань.

Координація військової логістики з цивільними структурами є важливим аспектом організації постачання. Збройні сили часто використовують цивільну інфраструктуру та послуги комерційних постачальників. Особливо важливою є взаємодія з національною транспортною системою, енергетичними компаніями та виробниками військової продукції. У багатьох країнах існують спеціальні механізми координації між військовими та цивільними логістичними структурами, які активуються під час кризових ситуацій.

Міжнародна співпраця у сфері військової логістики набуває все більшого значення, особливо в рамках коаліційних операцій. Спільне використання логістичних ресурсів дозволяє підвищити ефективність та знизити витрати. Для забезпечення такої співпраці розробляються спеціальні процедури та угоди, які регламентують взаємну логістичну підтримку. Важливими елементами міжнародної логістичної співпраці є стандартизація матеріальних засобів, узгодження процедур замовлення та постачання, а також створення спільних логістичних баз. Військова логістика повинна адаптуватися до різних типів бойових дій та оперативних умов. Високоманеврені операції вимагають мобільної логістичної підтримки, яка може швидко переміщуватися разом з військами. Оборонні операції потребують створення ешелонованої системи запасів. Операції в міських умовах створюють специфічні виклики для доставки вантажів через зруйновану інфраструктуру. Специфічні умови, такі як гірська місцевість, пустелі чи арктичні регіони, також вимагають спеціальних логістичних рішень.

Різні роди військ мають специфічні потреби в логістичному забезпеченні. Сухопутні війська потребують великих обсягів матеріальних засобів, які мають доставлятися безпосередньо до лінії зіткнення. Авіація вимагає спеціалізованого технічного обслуговування та значних обсягів авіаційного палива. Військово-морські сили потребують особливої системи постачання кораблів у морі. Ці відмінності вимагають створення спеціалізованих логістичних структур та процедур для кожного виду збройних сил [7].

Розвиток військової логістики спрямований на підвищення мобільності, гнучкості та стійкості системи постачання. Сучасні тенденції включають створення модульних логістичних підрозділів, які можуть швидко розгортатися в будь-якому регіоні. Впроваджуються технології контейнеризації та стандартизації вантажів для прискорення обробки та перевезення. Розвивається концепція "розподіленої логістики", яка передбачає розосередження запасів та дублювання логістичних маршрутів для підвищення живучості системи постачання в умовах протидії противника.

1.4 Огляд існуючих технологій організації логістичного забезпечення

Сучасні технології організації логістичного забезпечення у військовій сфері базуються на комплексному підході до управління матеріальними потоками з використанням передових інформаційних систем. Провідні армії світу активно впроваджують автоматизовані системи управління логістикою, які забезпечують інтеграцію всіх етапів постачання від планування потреб до доставки ресурсів кінцевим споживачам. Ці технології включають системи планування ресурсів підприємства (ERP), системи управління складами (WMS), системи управління транспортом (TMS) та системи відстеження і контролю матеріальних засобів у режимі реального часу.

Технологія радіочастотної ідентифікації (RFID) революціонізувала процеси обліку та відстеження військових вантажів. RFID-мітки дозволяють автоматично ідентифікувати та відстежувати переміщення матеріальних засобів без необхідності фізичного сканування кожного об'єкта. Система RFID забезпечує точність обліку на рівні 99,5% порівняно з 85-90% при використанні традиційних методів. У збройних силах США система Total Asset Visibility (TAV) використовує RFID-технології для забезпечення повної видимості логістичних активів на всіх рівнях командування. Системи глобального позиціонування (GPS) та супутникового зв'язку забезпечують точне відстеження місцезнаходження військових конвоїв та вантажів у режимі реального часу [8].

Інтеграція GPS-технологій з логістичними інформаційними системами дозволяє оптимізувати маршрути доставки, контролювати дотримання графіків постачання та оперативно реагувати на зміни в оперативній обстановці. Система Blue Force Tracker забезпечує командування інформацією про місцезнаходження підрозділів та стан їх забезпечення матеріально-технічними засобами.

Хмарні технології набувають все більшого поширення у військовій логістиці завдяки можливості забезпечення доступу до централізованих баз даних з будь-якої точки світу. Хмарні рішення дозволяють створювати масштабовані логістичні системи, які можуть адаптуватися до змінних потреб військових операцій. Використання приватних та гібридних хмарних архітектур забезпечує необхідний рівень безпеки для військових додатків. Збройні сили провідних країн активно переходять на хмарні платформи для розміщення своїх логістичних інформаційних систем.

Технології штучного інтелекту та машинного навчання впроваджуються для прогнозування потреб у матеріально-технічних засобах та оптимізації логістичних процесів. Алгоритми машинного навчання аналізують історичні дані споживання ресурсів, характеристики військових операцій та зовнішні фактори для точного прогнозування майбутніх потреб. Системи на базі штучного інтелекту можуть автоматично генерувати оптимальні плани постачання, враховуючи множину обмежень та критеріїв ефективності [9].

Блокчейн-технології досліджуються як засіб забезпечення прозорості та безпеки ланцюгів постачання військових матеріалів. Використання блокчейну дозволяє створити незмінний реєстр всіх транзакцій у логістичному ланцюзі, що забезпечує повну відстежуваність походження та переміщення матеріальних засобів. Ця технологія особливо важлива для контролю критично важливих матеріалів, таких як боєприпаси та спеціальне обладнання, де необхідно гарантувати автентичність та запобігти підробкам. Інтернет речей (IoT) створює можливості для створення "розумних" складів та транспортних засобів, обладнаних датчиками для моніторингу стану матеріальних засобів. IoT-датчики можуть відстежувати температуру, вологість, вібрації та інші параметри,

критичні для збереження якості військових матеріалів. Система датчиків може автоматично сповіщати про порушення умов зберігання або транспортування, дозволяючи оперативно вжити коригувальних заходів.

Мобільні технології забезпечують військовослужбовцям доступ до логістичної інформації безпосередньо в польових умовах. Спеціалізовані мобільні додатки дозволяють подавати заявки на постачання, відстежувати стан замовлень та отримувати інформацію про наявність ресурсів на складах. Використання планшетів та смартфонів з захищеним програмним забезпеченням значно прискорює процеси обміну інформацією між різними рівнями логістичної системи.

Роботизація та автоматизація складських операцій підвищують швидкість та точність обробки військових вантажів. Автоматизовані системи зберігання та видачі (AS/RS) дозволяють оптимізувати використання складських площ та скоротити час на підготовку замовлень. Роботи-навантажувачі та автономні транспортні засоби можуть працювати цілодобово без втоми, забезпечуючи безперервність логістичних процесів навіть в умовах обмеженого персоналу. Інтеграція різних технологій в єдині комплексні системи дозволяє досягти синергетичного ефекту та максимальної ефективності логістичного забезпечення [10]. Сучасні військові логістичні системи поєднують елементи ERP, WMS, TMS, RFID, GPS, IoT та інших технологій в єдину інтегровану платформу. Така інтеграція забезпечує безшовний обмін інформацією між різними підсистемами та дозволяє приймати оптимальні рішення на основі повної та актуальної інформації про стан логістичної системи.

1.5 Переваги та недоліки існуючих систем

Автоматизовані системи управління військовою логістикою демонструють значні переваги у порівнянні з традиційними методами організації постачання. Основною перевагою є підвищення точності та швидкості обробки логістичної інформації, що дозволяє скоротити час реагування на зміни в потребах

військових підрозділів з декількох днів до кількох годин. Автоматизація процесів усуває людський фактор у рутинних операціях, знижуючи кількість помилок в обліку та плануванні матеріально-технічного забезпечення з 15-20% до 2-3%. Інформаційні системи забезпечують повну видимість логістичних процесів, дозволяючи командуванню отримувати актуальну інформацію про стан забезпечення в режимі реального часу.

Економічні переваги впровадження сучасних технологій логістичного забезпечення включають оптимізацію рівнів запасів та скорочення загальних логістичних витрат. Аналіз даних, отриманих від армій провідних країн світу, показує можливість зниження логістичних витрат на 20-30% при правильному впровадженні автоматизованих систем. Оптимізація маршрутів доставки дозволяє скоротити витрати на транспортування на 15-25%, а точне планування потреб зменшує обсяги надлишкових запасів на 30-40%. Зниження втрат матеріальних засобів через покращений контроль та облік може досягати 10-15% від загального обсягу постачань. Стратегічні переваги включають підвищення готовності збройних сил до виконання бойових завдань завдяки надійному та своєчасному забезпеченню необхідними ресурсами. Автоматизовані системи дозволяють швидко адаптувати логістичні процеси до змін в оперативній обстановці, забезпечуючи гнучкість реагування на непередбачені ситуації. Інтеграція з системами планування операцій дозволяє синхронізувати логістичне забезпечення з тактичними та оперативними планами, підвищуючи ефективність військових дій.

Таблиця 1.1

Порівняльний аналіз переваг та недоліків сучасних систем військової логістики

Критерій	**Переваги**	**Недоліки**
Точність обліку	Підвищення до 99,5% порівняно з 85-90% при ручному обліку	Залежність від стабільності технічних засобів

Продовження таблиця 1.1

Швидкість обробки	Скорочення часу реагування з днів до годин	Потребує навчання персоналу
Економічна ефективність	Зниження логістичних витрат на 20-30%	Високі початкові інвестиції (сотні мільйонів доларів)
Інтеграція	Можливість об'єднання різних підсистем	Складність інтеграції застарілих систем
Безпека	Централізовані механізми захисту	Ризики кібератак та технічних збоїв
Масштабованість	Адаптація до різних розмірів організацій	Проблеми сумісності між різними платформами

Однак існуючі системи мають ряд суттєвих недоліків, які обмежують їх ефективність. Високі початкові витрати на впровадження та модернізацію інформаційних систем можуть становити значну частину оборонного бюджету. Вартість повномасштабного впровадження сучасної логістичної системи для збройних сил середньої країни може досягати сотень мільйонів доларів. Додаткові витрати пов'язані з необхідністю постійного оновлення апаратного та програмного забезпечення, навчання персоналу та технічної підтримки системи. Складність інтеграції різних систем та технологій створює технічні виклики при впровадженні комплексних логістичних рішень. Багато військових організацій використовують застарілі системи, які важко інтегрувати з сучасними технологіями. Відсутність стандартизації між різними системами та постачальниками може призводити до створення "інформаційних островів", які не можуть ефективно взаємодіяти між собою. Процес інтеграції може тривати роки та вимагати значних ресурсів.

Залежність від технологій створює ризики для функціонування логістичної системи в умовах технічних збоїв або кібератак. Вихід з ладу ключових компонентів інформаційної системи може паралізувати логістичні процеси на

тривалий період. Кібератаки на логістичні системи можуть призвести до компрометації конфіденційної інформації або порушення постачання критично важливих матеріалів. Необхідність забезпечення кіберзахисту додає складності та витрат при експлуатації системи.

Проблеми сумісності та стандартизації особливо гостро проявляються при проведенні коаліційних операцій, коли армії різних країн повинні взаємодіяти між собою. Різні стандарти кодування матеріальних засобів, формати обміну даними та процедури замовлення можуть значно ускладнити логістичну взаємодію. Навіть в рамках однієї армії різні підрозділи можуть використовувати несумісні системи, що створює перешкоди для ефективного управління ресурсами. Необхідність постійного навчання та перепідготовки персоналу є істотним недоліком складних технологічних систем. Швидкий розвиток технологій вимагає регулярного оновлення знань та навичок логістичного персоналу. Нестача кваліфікованих фахівців з інформаційних технологій у військовій сфері може обмежувати ефективність використання сучасних систем. Витрати на навчання персоналу можуть становити до 20-30% від загальної вартості впровадження системи.

Проблеми масштабування та адаптації існуючих систем до специфічних потреб різних військових організацій також створюють значні виклики. Системи, розроблені для великих армій, можуть бути надмірно складними та дорогими для менших військових формувань. З іншого боку, прості системи можуть не забезпечувати необхідної функціональності для великих та складних логістичних операцій. Необхідність кастомізації може значно збільшити час та вартість впровадження. Недостатня гнучкість деяких існуючих систем обмежує їх адаптацію до змінних умов військових операцій. Жорсткі алгоритми планування можуть не враховувати специфічні особливості конкретних операцій або регіонів. Складність внесення змін до налаштувань системи може призводити до затримок у адаптації логістичних процесів до нових вимог. Це особливо критично в умовах швидко змінюваної оперативної обстановки, коли необхідна миттєва реакція на нові виклики.

1.6 Важливість застосування інформаційних систем у військовій логістиці

Інформаційні системи стали критично важливим елементом сучасної військової логістики, оскільки складність та масштаби військових операцій досягли рівня, при якому ефективне управління матеріальними потоками неможливе без автоматизації. Сучасні збройні сили оперують десятками тисяч найменувань матеріально-технічних засобів, які необхідно доставляти в сотні різних локацій по всьому світу. Обсяг інформації, яку необхідно обробляти для забезпечення ефективного функціонування такої системи, перевищує можливості ручної обробки в тисячі разів. Лише використання потужних інформаційних систем дозволяє впоратися з цією складністю та забезпечити надійне постачання військ. Швидкість прийняття рішень у сучасних військових конфліктах досягає критичних значень, коли затримка в кілька годин може мати стратегічні наслідки. Інформаційні системи забезпечують миттєвий доступ до актуальної інформації про стан запасів, місцезнаходження вантажів та потреби військових підрозділів. Автоматизовані алгоритми можуть за лічені хвилини розрахувати оптимальні варіанти постачання, враховуючи множину факторів та обмежень [11]. Це дозволяє командуванню оперативно реагувати на зміни в оперативній обстановці та забезпечувати військам необхідну підтримку в критичні моменти.

Точність планування та прогнозування потреб значно підвищується при використанні інформаційних систем, здатних аналізувати великі масиви історичних даних та виявляти закономірності споживання різних типів ресурсів. Алгоритми машинного навчання можуть враховувати сотні змінних, включаючи тип операції, географічні умови, сезонні фактори, характеристики підрозділів та багато інших параметрів. Така точність планування дозволяє оптимізувати рівні запасів, зменшити ризики дефіциту критичних матеріалів та скоротити витрати на зберігання надлишкових ресурсів.

Координація між різними рівнями управління та видами збройних сил досягається завдяки єдиному інформаційному простору, створеному інтегрованими логістичними системами. Всі учасники логістичного процесу - від стратегічного командування до окремих підрозділів - мають доступ до однієї й тієї ж актуальної інформації про стан забезпечення. Це усуває дублювання замовлень, запобігає конфліктам при розподілі ресурсів та забезпечує узгодженість дій всіх елементів логістичної системи. Контроль та відстеження матеріальних засобів у режимі реального часу стає можливим завдяки інтеграції інформаційних систем з технологіями RFID, GPS та IoT. Командування може в будь-який момент отримати точну інформацію про місцезнаходження будь-якого вантажу, стан його доставки та очікуваний час прибуття. Така видимість дозволяє оперативно виявляти та усувати проблеми в ланцюзі постачання, запобігати втратам матеріалів та забезпечувати своєчасне інформування споживачів про стан їх замовлень.

Аналітичні можливості сучасних інформаційних систем дозволяють виявляти приховані закономірності та тенденції в логістичних процесах, які неможливо помітити при ручному аналізі. Системи можуть автоматично генерувати звіти про ефективність різних постачальників, оптимальність маршрутів доставки, сезонні коливання попиту та інші важливі показники. Ці аналітичні дані дозволяють приймати обґрунтовані стратегічні рішення щодо розвитку логістичної системи та підвищення її ефективності [12].

Забезпечення безпеки логістичної інформації стає критично важливим в умовах зростання кіберзагроз та необхідності захисту військових секретів. Інформаційні системи включають засоби шифрування, контролю доступу, аудиту дій користувачів та інші механізми забезпечення інформаційної безпеки. Централізоване управління безпекою дозволяє встановлювати єдині стандарти захисту та оперативно реагувати на виявлені загрози. Міжнародна взаємодія та стандартизація логістичних процесів значно спрощується при використанні сумісних інформаційних систем. Армії країн-партнерів можуть обмінюватися логістичною інформацією в стандартизованих форматах, координувати спільні

постачання та надавати взаємну логістичну підтримку. Це особливо важливо для країн-членів НАТО та інших військових альянсів, де ефективна логістична взаємодія є ключем до успіху спільних операцій.

Економічні переваги від впровадження інформаційних систем у військову логістику включають не лише прямі заощадження від оптимізації процесів, але й стратегічні вигоди від підвищення боєготовності збройних сил. Швидке та надійне логістичне забезпечення дозволяє військам ефективніше виконувати свої завдання, що має неоцінне значення для національної безпеки. За оцінками експертів, інвестиції в інформаційні системи військової логістики окупаються за 3-5 років за рахунок скорочення операційних витрат та підвищення ефективності використання ресурсів. Перспективи розвитку інформаційних систем військової логістики включають інтеграцію з технологіями штучного інтелекту, розширене використання автономних систем та розвиток предиктивної аналітики [13]. Ці технології дозволять створити самоадаптивні логістичні системи, здатні автоматично оптимізуватися під змінні умови та потреби. Важливість інформаційних систем у військовій логістиці буде лише зростати з розвитком нових технологій та ускладненням характеру військових операцій.

1.7 Вимоги до сучасних систем логістичного забезпечення військових потреб

Функціональні вимоги до сучасних систем логістичного забезпечення військових потреб визначаються специфікою військових операцій та критичною важливістю безперервного постачання збройних сил. Система повинна забезпечувати повний життєвий цикл управління матеріально-технічними засобами від планування потреб до утилізації. Основними функціональними блоками мають бути модулі планування та прогнозування потреб, управління закупівлями, складського обліку, управління транспортуванням, контролю якості та звітності. Система повинна підтримувати роботу з різними категоріями військових матеріалів, включаючи боєприпаси, паливо, продовольство, медичні

засоби, запасні частини та обладнання. Вимоги до продуктивності системи визначаються необхідністю обробки великих обсягів даних у режимі реального часу. Система повинна забезпечувати обробку до 10 мільйонів транзакцій на добу при одночасній роботі до 10 тисяч користувачів. Час відгуку системи на стандартні запити не повинен перевищувати 2 секунд, а на складні аналітичні запити - 30 секунд. Система повинна забезпечувати доступність на рівні 99,9%, що відповідає максимально допустимому часу простою 8,76 годин на рік. Пікові навантаження під час кризових ситуацій можуть збільшуватися в 5-10 разів, тому система повинна мати можливості горизонтального масштабування.

Таблиця 1.2

Технічні вимоги до системи логістичного забезпечення військових потреб

Параметр	**Мінімальні вимоги**	**Рекомендовані значення**	**Критичні показники**
Кількість одночасних користувачів	1 000	10 000	50 000
Час відгуку на стандартні запити	5 секунд	2 секунди	1 секунда
Час відгуку на аналітичні запити	60 секунд	30 секунд	10 секунд
Доступність системи	99%	99,5%	99,9%
Обсяг транзакцій на добу	1 млн	5 млн	10 млн
Максимальний час простою	24 години	8 годин	4 години
Час відновлення після збоїв	48 годин	24 години	4 години
Покриття тестами критичних модулів	70%	80%	90%

Вимоги до надійності включають забезпечення безперервної роботи системи навіть при виході з ладу окремих компонентів. Система повинна мати резервування всіх критичних елементів та можливість автоматичного перемикання на резервні ресурси при збоях. Відновлення після аварійних ситуацій повинно відбуватися протягом 4 годин для критичних функцій та 24 годин для повного відновлення всіх сервісів. Дані повинні резервуватися в режимі реального часу з можливістю їх відновлення з точністю до останньої транзакції.

Безпекові вимоги до військових логістичних систем є особливо жорсткими через критичну важливість інформації, що обробляється. Система повинна забезпечувати багаторівневу автентифікацію користувачів з використанням біометричних даних та цифрових сертифікатів. Всі дані повинні шифруватися як під час зберігання, так і під час передачі з використанням криптографічних алгоритмів, затверджених військовими стандартами. Система повинна підтримувати розмежування доступу на основі ролей та рівнів секретності інформації. Вимоги до мобільності та польового використання передбачають можливість роботи системи в умовах обмеженої комунікаційної інфраструктури. Система повинна підтримувати автономний режим роботи з можливістю синхронізації даних при відновленні зв'язку. Мобільні клієнти повинні працювати на захищених планшетах та смартфонах в умовах низької пропускної здатності каналів зв'язку [14, с. 87-92]. Інтерфейс повинен бути адаптований для роботи в польових умовах з урахуванням використання захисних рукавичок та роботи при яскравому сонячному світлі.

Інтеграційні вимоги включають необхідність взаємодії з існуючими військовими інформаційними системами та цивільними постачальниками. Система повинна підтримувати стандарти обміну даними, прийняті в НАТО (STANAG), та мати можливість інтеграції через веб-сервіси та API. Підтримка стандартних форматів даних, таких як XML, JSON та EDI, є обов'язковою для забезпечення сумісності з системами партнерів та постачальників.

Вимоги до аналітичних можливостей передбачають наявність потужних інструментів для аналізу логістичних даних та підтримки прийняття рішень. Система повинна забезпечувати генерацію стандартних звітів, інтерактивні дашборди та можливості ad-hoc аналізу. Обов'язковою є підтримка прогнозної аналітики з використанням методів машинного навчання для передбачення потреб у матеріальних засобах. Система повинна надавати засоби для моделювання різних сценаріїв постачання та оцінки їх ефективності.

Вимоги до масштабованості визначаються необхідністю підтримки різних рівнів військової організації від окремих підрозділів до національного рівня. Система повинна мати модульну архітектуру, що дозволяє поступове нарощування функціональності залежно від потреб конкретної військової організації. Підтримка розподілених конфігурацій є критичною для забезпечення роботи в умовах географічно розподілених військових підрозділів. Екологічні та енергетичні вимоги стають все більш важливими для військових систем. Система повинна забезпечувати енергоефективну роботу серверного обладнання та підтримувати режими енергозбереження для мобільних пристроїв [15, с. 54-60]. Використання "зелених" технологій та відновлюваних джерел енергії для живлення обчислювальної інфраструктури стає стандартною вимогою для сучасних військових систем.

Вимоги до підтримки життєвого циклу системи включають необхідність забезпечення технічної підтримки протягом 15-20 років експлуатації. Система повинна мати модульну архітектуру, що дозволяє поетапну модернізацію окремих компонентів без зупинки всієї системи. Документація системи повинна відповідати військовим стандартам та включати повні технічні специфікації, керівництва користувача та адміністратора, а також плани навчання персоналу.

1.8 Висновки до розділу

Аналіз теоретичних аспектів розробки інформаційних систем логістичного забезпечення військових потреб дозволяє зробити висновок про критичну

важливість таких систем для ефективного функціонування сучасних збройних сил. Військова логістика XXI століття характеризується надзвичайною складністю та масштабністю, що робить неможливим ефективне управління матеріальними потоками без використання передових інформаційних технологій. Традиційні методи ручного планування та контролю постачань не здатні забезпечити необхідну швидкість реагування на зміни в оперативній обстановці та точність управління ресурсами.

Галузь застосування інформаційних систем логістичного забезпечення охоплює всі рівні військової організації від стратегічного планування до тактичного забезпечення окремих підрозділів. Ці системи інтегруються з іншими військовими інформаційними системами, створюючи єдиний інформаційний простір для управління збройними силами. Особливої важливості набуває міжнародна стандартизація логістичних систем для забезпечення ефективної взаємодії між арміями країн-партнерів під час проведення спільних операцій.

Огляд існуючих технологій показав, що сучасні системи військової логістики базуються на комплексному використанні RFID, GPS, IoT, хмарних технологій, штучного інтелекту та блокчейну. Інтеграція цих технологій дозволяє створювати високоефективні системи відстеження матеріальних засобів у режимі реального часу, автоматизації складських операцій та оптимізації маршрутів постачання. Однак впровадження таких систем стикається з рядом викликів, включаючи високі витрати, складність інтеграції та необхідність забезпечення кіберзахисту.

Аналіз переваг та недоліків існуючих систем виявив, що основними перевагами є підвищення точності обліку до 99,5%, скорочення логістичних витрат на 20-30% та зниження часу реагування з декількох днів до кількох годин. Проте недоліками залишаються високі початкові витрати, складність інтеграції з застарілими системами, залежність від технологій та проблеми сумісності між різними платформами. Ці недоліки особливо критичні для збройних сил країн з обмеженими фінансовими ресурсами.

Важливість застосування інформаційних систем у військовій логістиці обумовлена неможливістю ефективного управління складними логістичними процесами традиційними методами. Сучасні військові операції вимагають координації постачання десятків тисяч найменувань матеріальних засобів у сотні локацій, що можливо лише при використанні автоматизованих систем. Інформаційні системи забезпечують швидкість прийняття рішень, точність планування, координацію між різними рівнями управління та повний контроль матеріальних потоків.

Вимоги до сучасних систем логістичного забезпечення включають високу продуктивність (до 10 мільйонів транзакцій на добу), надійність (доступність 99,9%), жорсткі безпекові стандарти з багаторівневою автентифікацією та шифруванням, можливість мобільного використання в польових умовах, інтеграцію з існуючими системами та потужні аналітичні можливості. Ці вимоги визначають архітектурні рішення та технологічний стек для розробки ефективних військових логістичних систем.

РОЗДІЛ 2. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ВИЗНАЧЕННЯ ІНСТРУМЕНТІВ ДЛЯ ВИРІШЕННЯ ПОСТАВЛЕНОЇ ЗАДАЧІ

2.1 Вибір програмного забезпечення для розробки інформаційної системи

Вибір програмного забезпечення для розробки інформаційної системи логістичного забезпечення військових потреб є критично важливим етапом проектування, який визначає архітектурні рішення, можливості масштабування та подальший розвиток системи. При виборі програмної платформи необхідно враховувати специфічні вимоги військової галузі, включаючи підвищені вимоги до безпеки, надійності, продуктивності та можливості роботи в екстремальних умовах. Основними критеріями вибору є функціональна повнота, технічні характеристики, вартість володіння, доступність технічної підтримки та відповідність військовим стандартам.

Для розробки серверної частини системи було розглянуто декілька альтернативних платформ, включаючи Microsoft .NET Framework, Java Enterprise Edition, Python з фреймворками Django/Flask та Node.js. Кожна з цих платформ має свої переваги та недоліки у контексті розробки військових інформаційних систем. Microsoft .NET забезпечує високу продуктивність та інтеграцію з корпоративними системами, але може мати обмеження щодо кросплатформеності. Java EE пропонує надійність та масштабованість, проте може бути надмірно складною для середніх за розміром проектів [16].

Python з фреймворком Django було обрано як основну платформу для розробки серверної частини системи через ряд переваг, критичних для військових додатків. Django забезпечує вбудовані механізми безпеки, включаючи захист від основних типів веб-атак (CSRF, XSS, SQL-ін'єкції), що є критично важливим для військових систем. Фреймворк має потужну систему автентифікації та авторизації з можливістю гнучкого налаштування ролей та

дозволів, що відповідає вимогам розмежування доступу у військових організаціях.

Архітектурна філософія Django "batteries included" дозволяє швидко розробляти функціональні додатки з мінімальною кількістю зовнішніх залежностей, що важливо для забезпечення стабільності та безпеки системи. Вбудована адміністративна панель Django значно прискорює розробку інтерфейсів управління системою та дозволяє швидко створювати інструменти для адміністрування даних. ORM (Object-Relational Mapping) Django забезпечує абстракцію від специфіки конкретної СУБД та автоматичний захист від SQL-ін'єкцій.

Для розробки клієнтської частини системи було проаналізовано можливості використання традиційних веб-технологій (HTML, CSS, JavaScript) у поєднанні з сучасними фреймворками React, Vue.js або Angular. З урахуванням необхідності забезпечення роботи системи на різних пристроях, включаючи захищені планшети та мобільні терміни, було прийнято рішення використовувати адаптивний веб-дизайн на базі HTML5, CSS3 та JavaScript з бібліотекою Bootstrap для забезпечення кросплатформеності та мобільної сумісності [17].

Інтеграційний рівень системи передбачає використання RESTful API для забезпечення взаємодії між різними компонентами системи та зовнішніми сервісами. Django REST Framework було обрано як основний інструмент для створення API через його повну інтеграцію з Django, потужні можливості серіалізації даних, вбудовану підтримку автентифікації та авторизації, а також автоматичну генерацію документації API. Це рішення забезпечує можливість легкої інтеграції з мобільними додатками та зовнішніми системами.

Для забезпечення високої доступності та масштабованості системи планується використання контейнеризації на базі Docker та оркестрації з Kubernetes. Ці технології дозволяють створювати відмовостійкі розподілені системи, які можуть автоматично масштабуватися під час пікових навантажень та забезпечувати безперервну роботу при виході з ладу окремих компонентів.

Використання контейнерів також спрощує розгортання системи в різних середовищах та забезпечує узгодженість між розробницьким, тестовим та продуктивним оточеннями.

Система моніторингу та логування базується на стеку ELK (Elasticsearch, Logstash, Kibana), який забезпечує централізований збір, обробку та аналіз логів з усіх компонентів системи. Це критично важливо для військових систем, де необхідно забезпечити повний аудит дій користувачів та моніторинг безпеки системи [18]. Додатково використовується Prometheus для моніторингу метрик продуктивності та Grafana для візуалізації моніторингових даних, що дозволяє оперативно виявляти та усувати проблеми в роботі системи.

2.2 Аналіз вимог до системи логістичного забезпечення військових потреб

Функціональні вимоги до системи логістичного забезпечення військових потреб визначаються специфікою військових операцій та необхідністю забезпечення безперервного постачання збройних сил. Система повинна підтримувати повний життєвий цикл управління матеріально-технічними засобами, включаючи планування потреб, формування замовлень, контроль постачань, складський облік, розподіл ресурсів та звітність. Основними функціональними модулями системи мають бути: модуль управління користувачами з розмежуванням доступу за ролями, модуль каталогу матеріальних засобів з можливістю категоризації та пошуку, модуль формування та обробки заявок на постачання, модуль планування та прогнозування потреб, модуль управління постачальниками та контрактами.

Нефункціональні вимоги включають високі стандарти продуктивності, надійності та безпеки, критичні для військових систем. Система повинна забезпечувати обробку до 50 тисяч одночасних користувачів з часом відгуку не більше 2 секунд для стандартних операцій та не більше 10 секунд для складних звітів. Доступність системи повинна становити не менше 99,5% з урахуванням

планового технічного обслуговування. Система повинна підтримувати горизонтальне масштабування для обробки пікових навантажень під час кризових ситуацій, коли кількість запитів може збільшуватися в 5-10 разів.

Вимоги до безпеки є критично важливими для військової системи і включають багаторівневу автентифікацію користувачів, шифрування всіх даних під час зберігання та передачі, ведення повного аудиту дій користувачів, захист від основних типів кібератак. Система повинна підтримувати інтеграцію з військовими системами управління ідентифікацією та забезпечувати відповідність стандартам інформаційної безпеки, прийнятим у збройних силах. Розмежування доступу повинно базуватися на принципі мінімальних привілеїв з можливістю гнучкого налаштування ролей та дозволів.

Вимоги до інтеграції передбачають можливість взаємодії з існуючими військовими інформаційними системами, системами постачальників та міжнародними стандартами обміну даними. Система повинна підтримувати протоколи EDI для електронного обміну документами з постачальниками, API для інтеграції з мобільними додатками та зовнішніми сервісами, експорт/імпорт даних у стандартних форматах XML, JSON, CSV. Важливою є підтримка стандартів НАТО для забезпечення сумісності з системами країн-партнерів.

Вимоги до мобільності включають підтримку роботи на мобільних пристроях у польових умовах з обмеженою пропускною здатністю каналів зв'язку. Система повинна мати адаптивний інтерфейс, який коректно відображається на різних типах пристроїв від смартфонів до планшетів. Необхідна підтримка офлайн-режиму з можливістю локального зберігання критичних даних та синхронізації при відновленні зв'язку. Мобільні клієнти повинні забезпечувати основні функції системи, включаючи перегляд статусу замовлень, створення термінових заявок та отримання сповіщень.

Вимоги до звітності та аналітики передбачають наявність потужних інструментів для формування стандартних та довільних звітів, інтерактивних панелей управління з ключовими показниками ефективності логістичних процесів. Система повинна підтримувати побудову графіків та діаграм для

візуалізації трендів споживання, аналізу ефективності постачальників, моніторингу виконання планів постачання. Необхідна можливість експорту звітів у різних форматах для подальшого аналізу або подання керівництву.

Технічні вимоги до розгортання системи включають підтримку різних операційних систем (Windows Server, Linux), можливість розгортання як у локальній інфраструктурі, так і в хмарному середовищі, підтримку кластерних конфігурацій для забезпечення високої доступності. Система повинна мати модульну архітектуру, що дозволяє поетапне впровадження функціональності та легке масштабування [19, с. 112-124]. Важливою є можливість резервного копіювання та відновлення даних з мінімальним часом простою.

Вимоги до користувацького інтерфейсу включають інтуїтивно зрозумілий дизайн, що не потребує довготривалого навчання персоналу, підтримку української та англійської мов, можливість налаштування інтерфейсу під потреби конкретних користувачів. Система повинна забезпечувати доступність для користувачів з обмеженими можливостями відповідно до стандартів WCAG 2.1. Інтерфейс повинен бути оптимізованим для швидкої роботи навіть при повільному інтернет-з'єднанні.

2.3 Порівняльний аналіз мов програмування для розробки інформаційної системи

Вибір мови програмування для розробки інформаційної системи логістичного забезпечення військових потреб є стратегічним рішенням, яке впливає на всі аспекти життєвого циклу системи від розробки до супроводу. При порівняльному аналізі було розглянуто п'ять основних альтернатив: Python, Java, C#/.NET, JavaScript (Node.js) та Go, кожна з яких має свої переваги та недоліки у контексті розробки корпоративних військових систем. Критеріями порівняння були: продуктивність та масштабованість, безпека, наявність бібліотек та фреймворків, складність розробки та супроводу, доступність кваліфікованих розробників, вартість розробки та підтримки.

Python виявився найбільш привабливим варіантом для розробки серверної частини системи завдяки поєднанню простоти розробки, потужних фреймворків та великій кількості спеціалізованих бібліотек. Мова має чистий та читабельний синтаксис, що значно спрощує розробку та супровід складних систем, особливо важливо при роботі в команді з різним рівнем досвіду. Django та Flask фреймворки забезпечують швидку розробку веб-додатків з вбудованими механізмами безпеки, що критично важливо для військових систем. Python має потужну екосистему бібліотек для роботи з базами даних, аналітики даних, машинного навчання та інтеграції з зовнішніми системами.

Java залишається одним з найпопулярніших варіантів для розробки корпоративних систем завдяки своїй надійності, продуктивності та масштабованості. Платформа Java EE (тепер Jakarta EE) пропонує комплексний набір технологій для розробки enterprise-додатків, включаючи механізми безпеки, транзакційності та розподілених обчислень. Основними перевагами Java є висока продуктивність, кросплатформеність, величезна кількість готових бібліотек та фреймворків, а також широка підтримка з боку великих корпорацій. Недоліками є відносна складність розробки, більша кількість коду для реалізації базової функціональності та вища вартість розробки.

C# та платформа .NET представляють потужну альтернативу з високою продуктивністю та інтеграцією з екосистемою Microsoft. Основними перевагами є висока продуктивність, потужні інструменти розробки Visual Studio, інтеграція з Active Directory для автентифікації користувачів, що важливо для корпоративних систем. .NET Core забезпечує кросплатформеність та можливість розгортання на Linux серверах. Недоліками є історична прив'язка до Windows (хоча .NET Core це змінює), вища вартість ліцензування Microsoft продуктів та менша гнучкість у порівнянні з відкритими технологіями.

JavaScript з Node.js набуває популярності для розробки серверних додатків завдяки можливості використання однієї мови для клієнтської та серверної частин, що спрощує розробку та супровід. Node.js забезпечує високу продуктивність для I/O-інтенсивних додатків завдяки асинхронній архітектурі та

подієво-орієнтованій моделі. Велика кількість npm пакетів дозволяє швидко знаходити готові рішення для більшості завдань. Недоліками є менша придатність для CPU-інтенсивних завдань, відносна молодість платформи для enterprise-розробки та потенційні проблеми з безпекою через велику кількість залежностей.

Go (Golang) є відносно новою мовою, розробленою Google, яка поєднує простоту Python з продуктивністю C++. Основними перевагами Go є висока продуктивність, ефективна робота з concurrent операціями, швидка компіляція, невеликий розмір готових бінарних файлів та відсутність залежностей при розгортанні. Мова має вбудовану підтримку для розробки мікросервісів та веб-API. Недоліками є відносно мала екосистема бібліотек у порівнянні з більш зрілими мовами, менша кількість кваліфікованих розробників на ринку та відсутність деяких advanced функціональностей.

Порівняння продуктивності показало, що Go та Java демонструють найкращі результати для high-load додатків, C# показує хороші результати в Windows середовищі, Python може бути повільнішим, але забезпечує достатню продуктивність для більшості корпоративних додатків при правильній архітектурі. Node.js ефективний для I/O операцій, але може мати проблеми з CPU-інтенсивними завданнями. Для військових систем, де критична надійність, Java та C# мають переваги завдяки зрілості платформ та корпоративній підтримці.

Аналіз безпеки показав, що всі розглянуті мови мають необхідні засоби для розробки безпечних додатків, але Python з Django та Java з Spring Security пропонують найбільш комплексні вбудовані механізми безпеки. C# має хорошу інтеграцію з Windows безпекою та Active Directory. Go має мінімалістичний підхід до безпеки, що може вимагати додаткової роботи розробників. Node.js потребує особливої уваги до безпеки через велику кількість зовнішніх залежностей.

Фінальне рішення на користь Python було прийнято на основі оптимального співвідношення швидкості розробки, наявності кваліфікованих

розробників, потужних фреймворків з вбудованою безпекою, великої кількості спеціалізованих бібліотек та нижчої загальної вартості володіння. Для високонавантажених компонентів системи планується використання мікросервісної архітектури з можливістю реалізації критичних сервісів на Go або Java при необхідності оптимізації продуктивності.

2.4 Вибір СУБД для зберігання даних логістичного забезпечення

Вибір системи управління базами даних для зберігання інформації про логістичне забезпечення військових потреб є критично важливим рішенням, яке впливає на продуктивність, надійність та масштабованість всієї системи. При аналізі було розглянуто як реляційні СУБД (PostgreSQL, MySQL, Microsoft SQL Server, Oracle), так і NoSQL рішення (MongoDB, Redis, Cassandra), а також гібридні підходи з використанням декількох типів баз даних для різних задач. Основними критеріями вибору були: продуктивність при роботі з великими обсягами даних, надійність та відмовостійкість, підтримка ACID транзакцій, можливості резервного копіювання та відновлення, вартість ліцензування, підтримка SQL стандартів та наявність кваліфікованих адміністраторів [20].

PostgreSQL була обрана як основна СУБД для системи завдяки своїм унікальним характеристикам, які ідеально підходять для військових логістичних систем. Це повністю відкрита СУБД з високим рівнем відповідності SQL стандартам, що забезпечує переносимість коду та зменшує ризики vendor lock-in. PostgreSQL підтримує складні типи даних, включаючи JSON, XML, масиви та геопросторові дані, що важливо для зберігання різноманітної логістичної інформації. Система має потужні можливості індексування, включаючи часткові індекси, функціональні індекси та full-text search, що забезпечує високу продуктивність пошуку.

Переваги PostgreSQL у контексті військових систем включають високий рівень безпеки з підтримкою row-level security, що дозволяє гнучко налаштовувати доступ до даних на рівні окремих записів. СУБД підтримує різні

методи автентифікації, включаючи LDAP та Kerberos, що важливо для інтеграції з корпоративними системами ідентифікації. PostgreSQL має надійну систему логування та аудиту, що критично важливо для військових систем, де необхідно відстежувати всі зміни даних. Підтримка материализованных представлень дозволяє оптимізувати продуктивність складних аналітичних запитів.

MySQL розглядалася як альтернатива через свою популярність та високу продуктивність для веб-додатків, але має обмеження у функціональності порівняно з PostgreSQL. Основними недоліками MySQL є менш строга відповідність SQL стандартам, обмежені можливості роботи зі складними типами даних та менші можливості для fine-tuning продуктивності. Проте MySQL може бути використана для кешування та допоміжних задач завдяки своїй простоті адміністрування та високій продуктивності для простих запитів.

Microsoft SQL Server пропонує потужні корпоративні можливості та тісну інтеграцію з екосистемою Microsoft, включаючи Active Directory, SSIS для ETL процесів та Power BI для аналітики. Основними перевагами є висока продуктивність, потужні інструменти адміністрування, хороша підтримка від Microsoft та інтеграція з .NET додатками. Недоліками є висока вартість ліцензування, прив'язка до Windows платформи (хоча SQL Server тепер підтримує Linux) та потенційні проблеми з vendor lock-in.

Oracle Database розглядалася як enterprise-рішення з найвищими можливостями продуктивності та масштабованості, особливо для критично важливих додатків. Oracle пропонує унікальні функції, такі як Real Application Clusters для високої доступності, Advanced Security для шифрування та Data Guard для аварійного відновлення. Проте надзвичайно висока вартість ліцензування та складність адміністрування роблять Oracle менш привабливою для проектів з обмеженим бюджетом.

NoSQL рішення розглядалися для специфічних випадків використання, де реляційна модель може бути недостатньою. MongoDB планується використовувати для зберігання логів та неструктурованих даних, таких як документи, зображення матеріальних засобів та різноманітні звіти у форматі

JSON. Redis буде використовуватися як in-memory кеш для часто запитуваних даних та сесій користувачів, що значно підвищить продуктивність системи. Cassandra розглядалася для зберігання великих обсягів часових рядів, таких як дані моніторингу та телеметрії, але було вирішено почати з PostgreSQL і розширювати архітектуру при необхідності.

Гібридна архітектура бази даних передбачає використання PostgreSQL як основного сховища для транзакційних даних, Redis для кешування та сесій, MongoDB для документів та логів, а також потенційно InfluxDB для часових рядів моніторингу. Така архітектура дозволяє оптимізувати продуктивність кожного типу даних, використовуючи найбільш підходящу технологію. Реплікація та резервне копіювання налаштовуються на рівні кожної СУБД з централізованим моніторингом через Prometheus та Grafana.

Стратегія міграції та масштабування передбачає початок з single-instance PostgreSQL для простоти розгортання та адміністрування, з подальшим переходом на master-slave реплікацію для підвищення доступності та розподілення читання. При зростанні навантаження планується використання PgBouncer для connection pooling та партиціонування таблиць для оптимізації продуктивності. Backup стратегія включає щоденні повні резервні копії з інкрементальними копіями кожні 4 години, з автоматичним тестуванням процедур відновлення [21-22].

Конфігурація безпеки PostgreSQL включає налаштування SSL/TLS шифрування для всіх з'єднань, обмеження доступу через pg_hba.conf, створення окремих ролей для різних типів користувачів з мінімальними необхідними привілеями. Планується використання прозорого шифрування даних на рівні файлової системи для додаткового захисту конфіденційної інформації. Моніторинг безпеки включає відстеження невдалих спроб підключення, аналіз журналів запитів та автоматичні сповіщення про підозрілу активність.

2.5 Вибір інструментів для тестування запитів до проектованого програмного забезпечення

Тестування програмного забезпечення для військових логістичних систем вимагає комплексного підходу з використанням різних типів тестування та спеціалізованих інструментів. Для забезпечення якості розроблюваної системи було обрано багаторівневу стратегію тестування, що включає модульне тестування (unit testing), інтеграційне тестування, функціональне тестування, тестування продуктивності, тестування безпеки та тестування користувацького інтерфейсу. Кожен рівень тестування має свої специфічні інструменти та методології, адаптовані до особливостей розробки військових інформаційних систем.

Для модульного тестування Python-коду було обрано фреймворк pytest через його простоту використання, потужні можливості параметризації тестів та широку підтримку плагінів. Pytest дозволяє створювати читабельні тести з мінімальною кількістю boilerplate коду, що важливо для підтримки великої кількості тестів у довгостроковій перспективі. Для тестування Django моделей та представлень використовується вбудований Django TestCase, який забезпечує ізоляцію тестів через створення тимчасової бази даних. Coverage.py використовується для аналізу покриття коду тестами з цільовим показником не менше 80% для критичних компонентів системи.

Інтеграційне тестування API здійснюється за допомогою Django REST Framework test tools у поєднанні з pytest-django для спрощення налаштування тестового середовища. Для тестування складних сценаріїв взаємодії між компонентами системи використовується pytest-mock для створення мок-об'єктів та імітації зовнішніх залежностей. Factory Boy застосовується для генерації тестових даних, що дозволяє створювати реалістичні набори даних для тестування без жорсткої прив'язки до конкретних значень.

Тестування продуктивності проводиться з використанням Locust - сучасного інструменту для навантажувального тестування, написаного на Python. Locust дозволяє створювати реалістичні сценарії навантаження з імітацією поведінки реальних користувачів, масштабувати тестування на

декілька машин та отримувати детальну статистику продуктивності. Для тестування бази даних використовується pgbench для PostgreSQL з метою оцінки продуктивності запитів та оптимізації конфігурації СУБД. Apache JMeter розглядається як альтернативний інструмент для тестування продуктивності веб-сервісів та API.

Тестування безпеки включає автоматизовані інструменти сканування вразливостей та ручні перевірки критичних компонентів. OWASP ZAP використовується для автоматичного сканування веб-додатку на предмет відомих вразливостей, включаючи SQL-ін'єкції, XSS, CSRF та інші атаки з OWASP Top 10. Bandit застосовується для статичного аналізу Python коду з метою виявлення потенційних проблем безпеки. Safety перевіряє використовувані бібліотеки на наявність відомих вразливостей. Для тестування автентифікації та авторизації створюються спеціальні тестові сценарії, що перевіряють правильність розмежування доступу.

Тестування користувацького інтерфейсу здійснюється за допомогою Selenium WebDriver для автоматизації браузерних тестів. Playwright розглядається як сучасна альтернатива Selenium з кращою підтримкою сучасних веб-технологій та більш стабільною роботою. Для тестування адаптивності інтерфейсу використовуються емулятори різних пристроїв та розмірів екранів. Accessibility тестування проводиться за допомогою axe-core для забезпечення відповідності стандартам WCAG 2.1, що важливо для військових систем, які повинні бути доступними для користувачів з різними потребами.

2.6 Визначення методології розробки проекту

Методологія розробки інформаційної системи логістичного забезпечення військових потреб повинна враховувати специфічні вимоги військової галузі, включаючи високі стандарти безпеки, надійності та відповідності регулятивним вимогам. Після аналізу різних підходів до управління проектами розробки ПЗ було обрано гібридну методологію, що поєднує елементи Agile (Scrum) для

гнучкості розробки та Waterfall для критичних фаз, які вимагають формального документування та валідації. Ця методологія дозволяє збалансувати потребу в швидкій адаптації до змінних вимог з необхідністю дотримання жорстких стандартів якості та безпеки.

Scrum методологія обрана як основа для організації процесу розробки завдяки своїй здатності забезпечувати регулярну доставку функціональних частин системи та швидку адаптацію до змін вимог. Команда розробки організована у вигляді самоорганізованих кросфункціональних груп, що включають розробників, тестувальників, DevOps інженерів та представників замовника. Спринти тривалістю 2 тижні забезпечують регулярний ритм роботи з демонстрацією результатів та отриманням зворотного зв'язку. Щоденні stand-up зустрічі дозволяють швидко ідентифікувати та вирішувати проблеми, що особливо важливо для складних технічних систем.

Документування проекту здійснюється відповідно до військових стандартів з адаптацією до agile підходу. Замість детальної документації на початку проекту, документи створюються ітеративно разом з розробкою функціональності. Основними документами є: технічне завдання з высокоуровневими вимогами, архітектурний опис системи, специфікації API, посібники користувача та адміністратора, документація з безпеки та план тестування [23-24]. Документи підтримуються в актуальному стані через автоматизовані інструменти генерації документації з коду та коментарів.

Управління якістю інтегровано в процес розробки через практики Continuous Integration/Continuous Deployment (CI/CD). Кожна зміна коду автоматично проходить через pipeline, що включає статичний аналіз коду, запуск модульних та інтеграційних тестів, сканування на вразливості безпеки та розгортання в тестовому середовищі. GitLab CI/CD обрано як основну платформу для автоматизації через її інтеграцію з системою контролю версій та можливість створення складних пайплайнів. Автоматизовані перевірки включають code review обов'язкові для всіх змін, дотримання стандартів кодування та проходження всіх тестів.

Управління ризиками проекту базується на ідентифікації, оцінці та мітигації ризиків на кожному етапі розробки. Основними ризиками є: технічні ризики, пов'язані зі складністю інтеграції різних компонентів; ризики безпеки через критичну важливість військових даних; ризики продуктивності при роботі з великими обсягами даних; ризики відповідності регулятивним вимогам. Для кожного ризику розроблено план мітигації з конкретними діями та відповідальними особами. Регулярні ретроспективи дозволяють виявляти нові ризики та адаптувати стратегії їх управління.

Комунікація з замовником організована через регулярні демонстрації результатів спринтів, щотижневі статусні зустрічі та квартальні ради проекту для обговорення стратегічних питань. Product Owner з боку замовника інтегрований в команду розробки для забезпечення постійного зворотного зв'язку та прийняття рішень щодо пріоритизації функціональності. Використовується Jira для управління завданнями та відстеження прогресу, Confluence для документування та Slack для оперативної комунікації між членами команди [25-26].

2.7 Висновки до розділу

Проведений аналіз предметної області та визначення інструментів для розробки інформаційної системи логістичного забезпечення військових потреб дозволив сформувати обґрунтовану технологічну архітектуру та методологічний підхід до реалізації проекту. Вибір Python з фреймворком Django як основної платформи розробки забезпечує оптимальне поєднання швидкості розробки, безпеки та функціональності, критично важливих для військових систем. Архітектурне рішення з використанням контейнеризації Docker та оркестрації Kubernetes забезпечує необхідну масштабованість та відмовостійкість системи.

Детальний аналіз вимог до системи виявив необхідність підтримки 50 тисяч одночасних користувачів з доступністю 99,5% та багаторівневою системою безпеки. Функціональні вимоги включають повний життєвий цикл

управління матеріально-технічними засобами з інтеграцією до існуючих військових систем та підтримкою мобільних пристроїв для польового використання. Нефункціональні вимоги визначають жорсткі стандарти продуктивності, безпеки та надійності, характерні для критично важливих військових систем.

Порівняльний аналіз мов програмування підтвердив правильність вибору Python для серверної частини завдяки поєднанню простоти розробки, потужних фреймворків безпеки та великої екосистеми спеціалізованих бібліотек. Розглянуті альтернативи (Java, C#, Node.js, Go) мають свої переваги, але Python забезпечує найкраще співвідношення швидкості розробки, вартості та функціональності для даного проекту. Мікросервісна архітектура дозволить використовувати різні технології для оптимізації критичних компонентів при необхідності.

Вибір PostgreSQL як основної СУБД обґрунтований її надійністю, безпекою та функціональністю, критично важливими для військових систем. Гібридна архітектура бази даних з використанням Redis для кешування та MongoDB для неструктурованих даних забезпечує оптимізацію продуктивності для різних типів даних. Стратегія масштабування від single-instance до реплікації та партиціонування дозволить адаптувати систему до зростаючих потреб.

Комплексна стратегія тестування з використанням pytest, Locust, OWASP ZAP та Selenium забезпечує всебічну перевірку якості системи на всіх рівнях від модульного до інтеграційного тестування. Автоматизація тестування безпеки особливо важлива для військових систем з високими вимогами до захисту інформації. Цільовий показник покриття коду тестами 80% для критичних компонентів гарантує високу якість системи.

Гібридна методологія розробки, що поєднує Agile (Scrum) з елементами Waterfall, оптимально відповідає специфіці військових проектів з необхідністю балансу між гнучкістю розробки та формальними вимогами документування. Інтеграція CI/CD процесів забезпечує автоматизацію контролю якості та прискорює доставку функціональності. Структурований підхід до управління

ризиками та комунікації з замовником створює передумови для успішної реалізації проекту

РОЗДІЛ 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Моделювання предметної області

3.1.1 Розробка контекстних діаграм

Контекстна діаграма інформаційної системи логістичного забезпечення військових потреб визначає межі системи та її взаємодію з зовнішніми сутностями. Система розглядається як єдиний процес найвищого рівня, який отримує вхідні дані від зовнішніх джерел та генерує вихідні дані для зовнішніх споживачів. Основними зовнішніми сутностями є військові підрозділи, які подають заявки на матеріально-технічне забезпечення, постачальники матеріальних засобів, командування різних рівнів, яке приймає рішення щодо розподілу ресурсів, фінансові структури, що контролюють витрати, та логістичні центри, які здійснюють фізичну доставку матеріалів.

Вхідні потоки даних включають заявки на постачання від військових підрозділів, інформацію про наявність матеріалів від постачальників, директивні документи від командування щодо пріоритетів постачання, бюджетні обмеження від фінансових структур та звіти про доставку від логістичних центрів. Заявки на постачання містять специфікацію необхідних матеріалів, кількість, терміни доставки та обґрунтування потреби. Інформація від постачальників включає каталоги товарів, ціни, умови поставки та наявність на складах. Директивні документи визначають стратегічні пріоритети розподілу ресурсів та оперативні зміни в планах постачання.

Вихідні потоки даних системи включають замовлення постачальникам з детальною специфікацією матеріалів та умов поставки, звіти для командування про стан забезпечення військових підрозділів, фінансові звіти про витрати та планові потреби в коштах, розпорядження логістичним центрам щодо маршрутів та графіків доставки, а також сповіщення військовим підрозділам про статус їх

заявок. Замовлення постачальникам автоматично генеруються на основі затверджених заявок з урахуванням наявних запасів та бюджетних обмежень. Звіти для командування надають аналітичну інформацію про ефективність логістичних процесів та рекомендації щодо оптимізації.

Контекстна діаграма також відображає інформаційні потоки з суміжними системами, такими як фінансова система для контролю витрат та бюджетування, система управління персоналом для верифікації повноважень користувачів, геоінформаційна система для планування маршрутів доставки та система моніторингу для відстеження стану матеріально-технічних засобів. Інтеграція з цими системами здійснюється через стандартизовані API та формати обміну даними, що забезпечує цілісність інформації та уникнення дублювання даних.

Межі системи чітко визначені та включають всі процеси від отримання заявок до контролю виконання поставок, але не включають фізичні процеси доставки, фінансові розрахунки та управління персоналом, які здійснюються зовнішніми системами. Контекстна діаграма служить основою для подальшої декомпозиції системи на функціональні підсистеми та визначення детальних вимог до кожного компонента [27]. Вона також забезпечує розуміння системи всіма зацікавленими сторонами та служить основою для валідації архітектурних рішень.

3.1.2 Розробка функціональних діаграм

Функціональні діаграми деталізують основні бізнес-процеси системи логістичного забезпечення військових потреб через декомпозицію контекстної діаграми на функціональні блоки нижчого рівня. Головний процес "Управління логістичним забезпеченням" декомпонується на п'ять основних функціональних процесів: "Управління заявками", "Планування постачань", "Управління замовленнями", "Контроль виконання" та "Аналітика та звітність". Кожен з цих процесів має свої специфічні входи, виходи, механізми виконання та управляючі впливи, які детально описуються на функціональних діаграмах другого рівня.

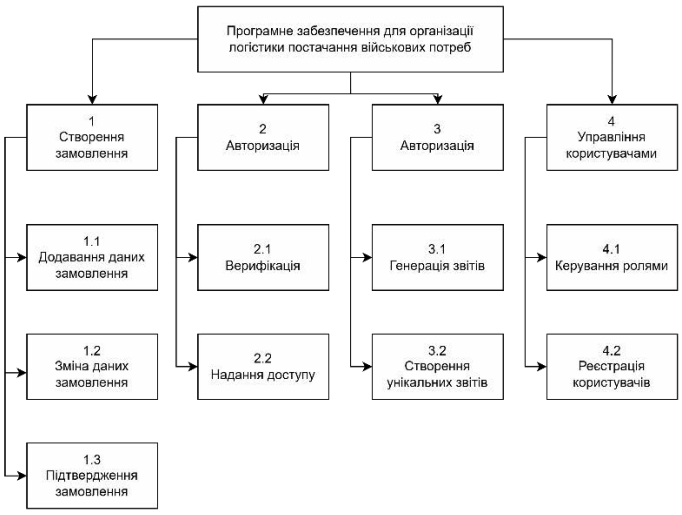

Рис. 3.1 - Функціональна діаграма декомпозиції програмного забезпечення для організації логістики постачання військових потреб

Як показано на рисунку 3.1, система декомпонується на чотири основні функціональні блоки:

– Створення замовлення - включає підпроцеси додавання даних замовлення (1.1), зміни даних замовлення (1.2) та підтвердження замовлення (1.3);

– Авторизація - забезпечує верифікацію користувачів (2.1) та надання доступу до системи (2.2);

– Авторизація - відповідає за генерацію звітів (3.1) та створення унікальних звітів (3.2);

– Управління користувачами - включає керування ролями (4.1) та реєстрацію користувачів (4.2).

Кожен функціональний блок має чітко визначені входи та виходи, що забезпечує структурований підхід до розробки системи та дозволяє незалежно розробляти кожен компонент.

Рис. 3.2 - Діаграма потоків даних (DFD) інформаційної системи логістичного забезпечення військових потреб

Діаграма потоків даних (рисунок 3.2) відображає рух інформації між основними процесами системи, зовнішніми сутностями та сховищами даних. Система включає два основні зовнішні актори:

– Адміністратор - керує даними користувачів та здійснює авторизацію в системі;

– Користувач - створює замовлення та формує звіти про свою діяльність.

Основні процеси системи:

1. Управління користувачами - обробляє дані користувачів та зберігає їх у відповідному сховищі;

2. Авторизація - перевіряє права доступу користувачів до системи;

3. Створення замовлення - обробляє заявки користувачів та зберігає необхідні активи;

4. Створення унікальних звітів - генерує персоналізовані звіти на основі даних користувача.

Сховища даних включають: (1) Користувачі, (2) Необроблені активи та (3) Звіти. Потоки даних показують напрямок передачі інформації між процесами,

акторами та сховищами, забезпечуючи повне розуміння інформаційних зв'язків у системі.

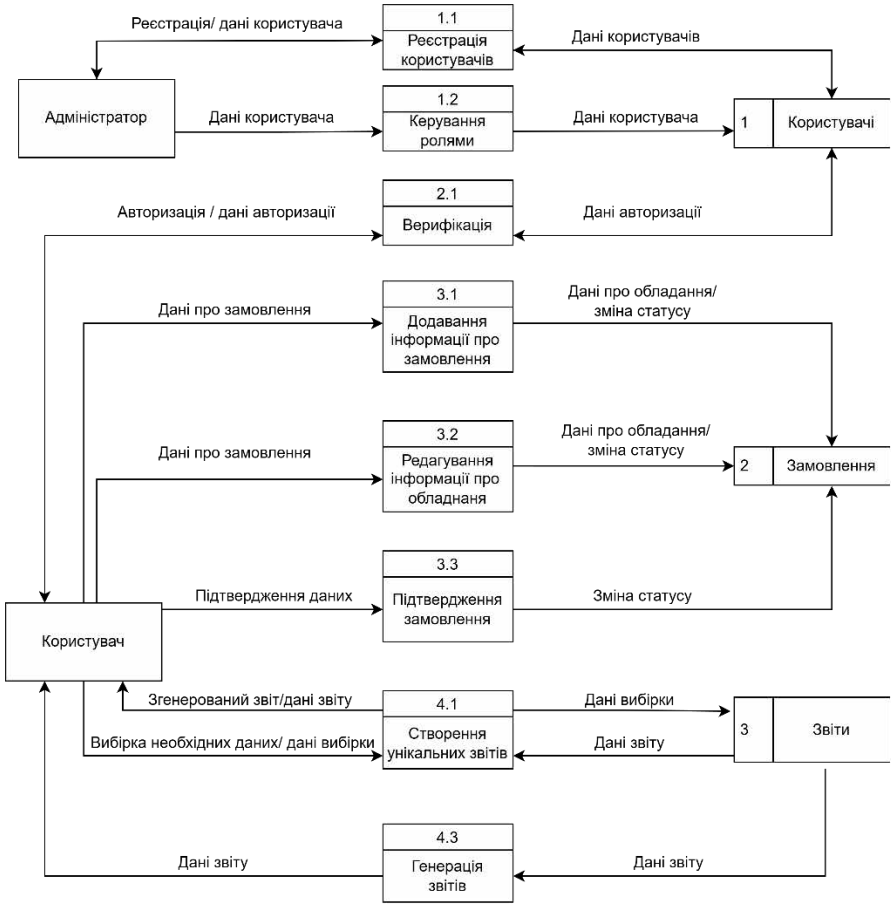

Рис. 3.3 - Деталізована діаграма потоків даних (DFD рівня 1) інформаційної системи логістичного забезпечення військових потреб

Деталізована діаграма потоків даних (рисунок 3.3) розкриває внутрішню структуру основних процесів системи та показує детальні потоки інформації між підпроцесами. Система включає наступні групи процесів:

Група 1 - Управління користувачами:

1.1 Реєстрація користувача - обробляє дані нових користувачів;

1.2 Керування ролями - управляє правами доступу користувачів.

Група 2 - Авторизація:

2.1 Верифікація - перевіряє автентичність користувачів.

Група 3 - Обробка замовлень:

3.1 Додавання інформації про замовлення - створює нові заявки;

3.2 Редагування інформації про обладнання - модифікує існуючі замовлення;

3.3 Підтвердження замовлення - фіналізує процес створення заявки.

Група 4 - Звітність:

4.1 Створення унікальних звітів - генерує персоналізовані звіти;

4.3 Генерація звітів - створює стандартні звіти системи.

Діаграма демонструє трирівневу структуру сховищ даних: (1) Користувачі, (2) Замовлення та (3) Звіти, які забезпечують централізоване зберігання та обмін інформацією між усіма підсистемами.

Процес "Управління заявками" включає підпроцеси прийому заявок від військових підрозділів, їх валідації на відповідність встановленим вимогам, класифікації за пріоритетністю та терміновістю, розгляду та затвердження уповноваженими особами. Вхідними даними є заявки на постачання у стандартизованому форматі, довідники матеріально-технічних засобів, регламенти розгляду заявок та інформація про бюджетні обмеження. Механізмами виконання є автоматизовані засоби валідації даних, системи електронного документообігу та бази даних користувачів з їх повноваженнями. Вихідними даними є затверджені заявки, повідомлення про відхилення з обґрунтуванням та звіти про стан обробки заявок.

Процес "Планування постачань" агрегує затверджені заявки, аналізує наявні запаси на складах, прогнозує потреби на основі історичних даних та формує оптимальні плани постачань з урахуванням бюджетних обмежень та пріоритетів. Підпроцеси включають аналіз потреб, оптимізацію планів постачання за критеріями вартості та терміновості, координацію з постачальниками щодо можливостей поставки та формування календарних планів. Механізмами є алгоритми оптимізації, бази даних про постачальників та їх можливості, аналітичні інструменти прогнозування попиту та системи моделювання логістичних процесів.

Процес "Управління замовленнями" трансформує плани постачання в конкретні замовлення постачальникам, контролює процес їх узгодження та підписання договорів, відстежує виконання замовлень та вносить необхідні корективи. Функціональні підпроцеси включають формування специфікацій замовлень, вибір оптимальних постачальників на основі критеріїв ціни, якості та надійності, автоматичну генерацію договірних документів та їх узгодження з юридичними службами. Механізми включають системи електронних торгів, бази даних кваліфікованих постачальників, шаблони договорів та засоби електронного підпису документів.

Процеси "Контроль виконання" та "Аналітика та звітність" забезпечують моніторинг ефективності логістичної системи та підготовку управлінської інформації для прийняття стратегічних рішень. Контроль виконання включає відстеження статусу поставок у режимі реального часу, контроль дотримання термінів та якості поставлених матеріалів, управління претензіями та рекламаціями. Аналітика включає формування статистичних звітів, аналіз трендів споживання різних категорій матеріалів, оцінку ефективності роботи постачальників та розрахунок ключових показників ефективності логістичних процесів. Ці процеси забезпечують зворотний зв'язок для оптимізації всієї системи логістичного забезпечення.

3.1.3 Розробка UML-діаграм системи

UML-діаграми забезпечують комплексне моделювання архітектури системи логістичного забезпечення військових потреб з використанням стандартизованої нотації для опису структурних та поведінкових аспектів системи. Діаграма прецедентів (Use Case Diagram) ідентифікує основних акторів системи та їх взаємодію з функціональними можливостями системи. Основними акторами є військовослужбовці різних рівнів (солдати, офіцери, командири), логістичні координатори, постачальники, фінансові контролери та системні адміністратори.

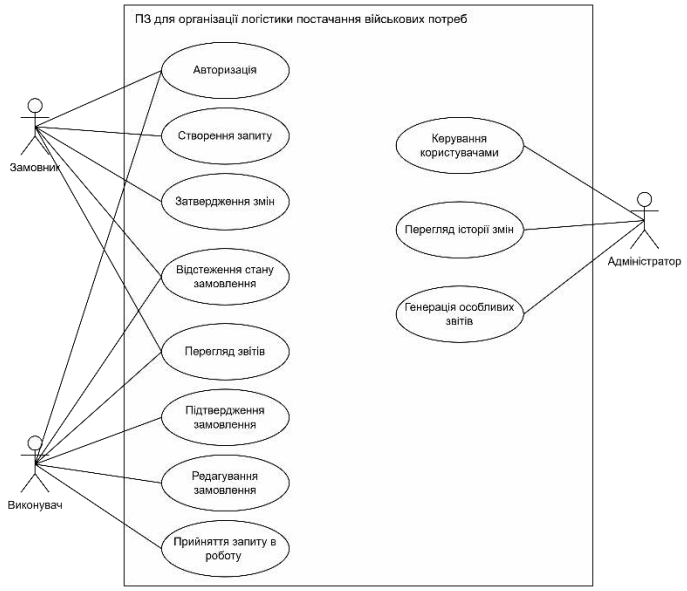

Рис. 3.4 - Діаграма прецедентів (Use Case) інформаційної системи логістичного забезпечення військових потреб

Як показано на рисунку 3.4, система має трьох основних акторів: Замовник (військовослужбовець, який подає заявки), Виконавець (відповідальний за обробку замовлень) та Адміністратор (системний адміністратор). Кожен актор має доступ до специфічного набору функцій відповідно до своїх ролей та повноважень. Замовник може авторизуватися, створювати заявки, затверджувати дані, відстежувати стан замовлень, переглядати деталі та підтверджувати замовлення. Виконавець має додаткові права на розглядання замовлень та прийняття їх в роботу. Адміністратор керує користувачами та має доступ до перегляду історії змін у системі.

Кожен актор має специфічний набір прецедентів, таких як "Подати заявку на постачання", "Затвердити заявку", "Сформувати замовлення", "Відстежити статус поставки" та "Сформувати звіт".

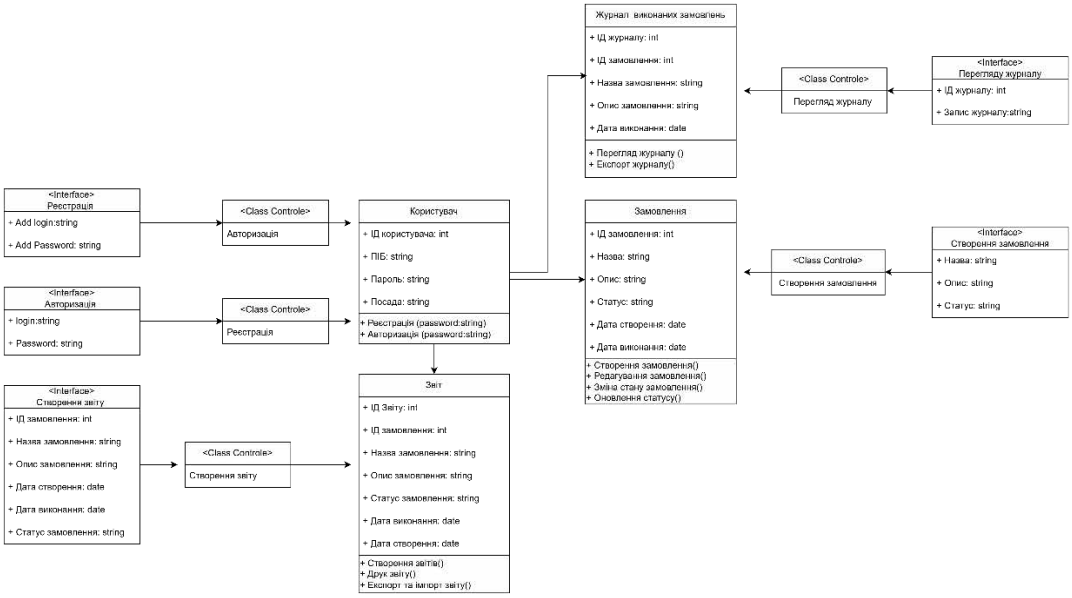

Рис. 3.5 - Діаграма класів інформаційної системи логістичного забезпечення військових потреб

Діаграма класів (рисунок 3.5) визначає основні сутності предметної області та їх взаємозв'язки. Центральними класами системи є:

– Користувач - містить інформацію про користувачів системи з атрибутами ідентифікації та авторизації;

– Заявка - основна бізнес-сутність, що описує замовлення на матеріально-технічні засоби;

– Замовлення - деталізація заявки з інформацією про конкретні матеріали та кількості;

– Журнал виконання замовлень - відстежує історію змін та статуси обробки замовлень.

Інтерфейси Реєстрація, Авторизація та Створення заяви забезпечують стандартизовану взаємодію з основними функціями системи. Контрольні класи Авторизація, Реєстрація та Створення заяви реалізують бізнес-логіку відповідних процесів. Класи пов'язані відношеннями асоціації, композиції та реалізації, що відображають реальні бізнес-процеси логістичного забезпечення.

Діаграма класів (Class Diagram) визначає основні сутності предметної області та їх взаємозв'язки. Центральними класами є User (користувач з ролями

та правами доступу), Order (замовлення з атрибутами номера, дати, статусу), OrderItem (позиція замовлення з матеріалом та кількістю), Material (матеріально-технічний засіб з характеристиками), Supplier (постачальник з контактною інформацією та рейтингом), Warehouse (склад з адресою та залишками матеріалів). Класи пов'язані відношеннями асоціації, агрегації та композиції, що відображають бізнес-логіку взаємодії між сутностями. Наслідування використовується для спеціалізації користувачів на різні ролі та типізації матеріалів за категоріями.

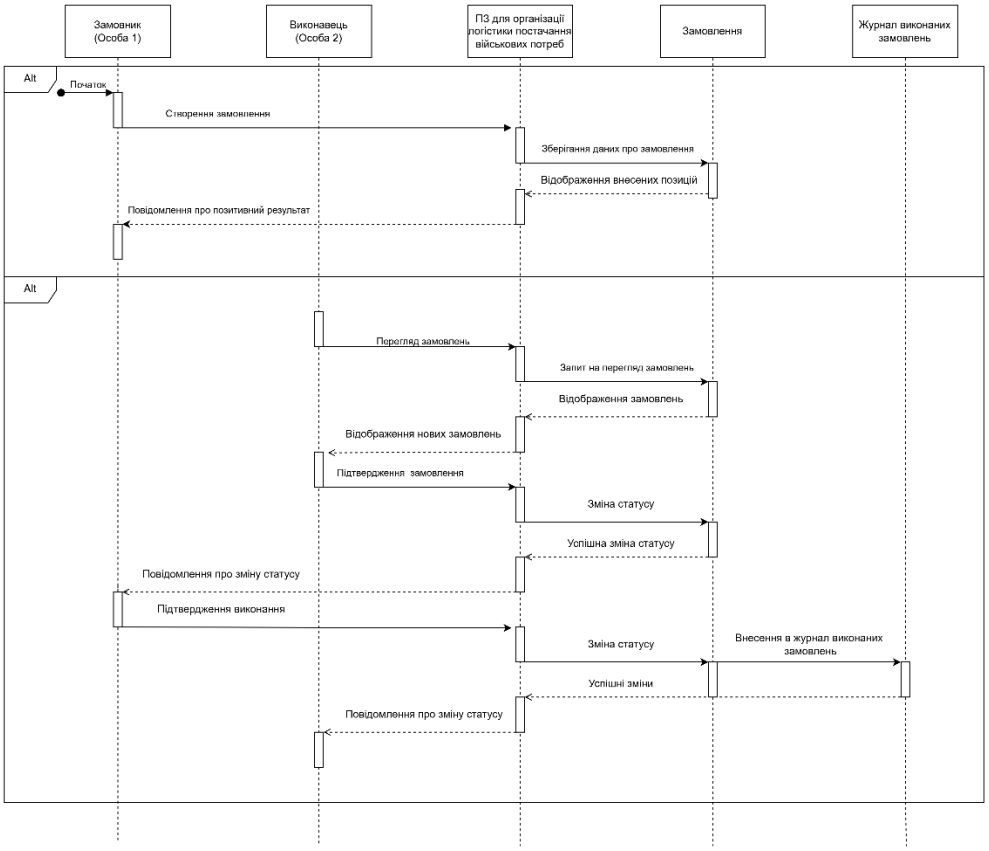

Рис. 3.6 - Діаграма послідовності процесу створення та обробки замовлення в інформаційній системі логістичного забезпечення військових потреб

Діаграми послідовності (рисунок 3.6) деталізують взаємодію об'єктів у часі для ключових бізнес-процесів системи. На діаграмі показано процес створення та обробки замовлення з участю п'яти основних акторів:

– Замовник (особа 1) - ініціює створення замовлення;

– Виконавець (особа 2) - обробляє та затверджує замовлення;

– ПЗ для організації логістики постачання військових потреб - основна система;

– Замовлення - об'єкт предметної області;

– Журнал виконання замовлень - система аудиту та відстеження.

Процес включає два основні етапи взаємодії (Alt 1 та Alt 2), що показують альтернативні сценарії обробки. Перший етап охоплює створення замовлення, збереження даних, відображення інформації та повідомлення про поточний результат. Другий етап включає перегляд замовлень, зміну статусу, відправлення сповіщень та фінальне підтвердження виконання.

Синхронні повідомлення (суцільні стрілки) показують активні виклики методів, а пунктирні стрілки - повернення результатів. Часова послідовність взаємодії забезпечує цілісність бізнес-процесу та правильну обробку станів замовлення на кожному етапі.

Діаграми послідовності (Sequence Diagrams) деталізують взаємодію об'єктів у часі для ключових бізнес-процесів системи. Діаграма для процесу "Обробка заявки на постачання" показує послідовність повідомлень між об'єктами User, OrderService, ValidationService, ApprovalService та NotificationService. Процес починається з створення заявки користувачем, проходить через етапи валідації даних, перевірки бюджетних обмежень, маршрутизації на затвердження відповідальним особам та завершується відправкою сповіщення про результат. Діаграма відображає як синхронні, так і асинхронні повідомлення, умовні блоки та цикли обробки.

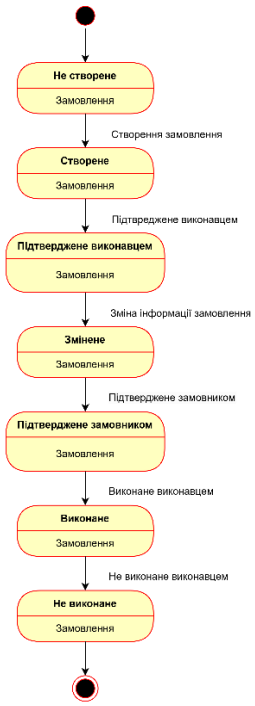

Рис. 3.7 - Діаграма станів замовлення в інформаційній системі логістичного забезпечення військових потреб

Діаграма станів (рисунок 3.7) моделює життєвий цикл ключового об'єкта системи - замовлення на матеріально-технічні засоби. Замовлення проходить через наступні стани:

– Не створене - початковий стан перед ініціюванням процесу замовлення;

– Створене - замовлення створено та внесено до системи;

– Підтверджене виконавцем - замовлення прийнято до обробки відповідальною особою;

– Змінене - внесені корективи до параметрів замовлення;

– Підтверджене замовником - замовлення затверджено ініціатором заявки;

– Виконане - замовлення успішно виконано та матеріали доставлено;

– Не виконане - замовлення не може бути виконано з різних причин.

Переходи між станами ініціюються подіями: 'Створення замовлення', 'Підтвердження виконавцем', 'Зміна інформації замовлення', 'Підтвердження

замовником', 'Виконання виконавцем' та 'Не виконання виконавцем'. Діаграма включає початкові та кінцеві стани, що показують повний життєвий цикл замовлення від створення до завершення. Така структура забезпечує чіткий контроль над процесом обробки замовлень та дозволяє відстежувати їх статус на кожному етапі.

Діаграма станів (State Diagram) моделює життєвий цикл ключових об'єктів системи, зокрема замовлення (Order). Замовлення проходить через стани: "Створено", "На розгляді", "Затверджено", "Відправлено постачальнику", "Підтверджено постачальником", "В процесі виконання", "Доставлено", "Закрито" або "Відхилено". Переходи між станами ініціюються подіями, такими як отримання затвердження, підтвердження від постачальника або повідомлення про доставку. Діаграма включає охоронні умови для переходів та дії, що виконуються при вході або виході зі стану.

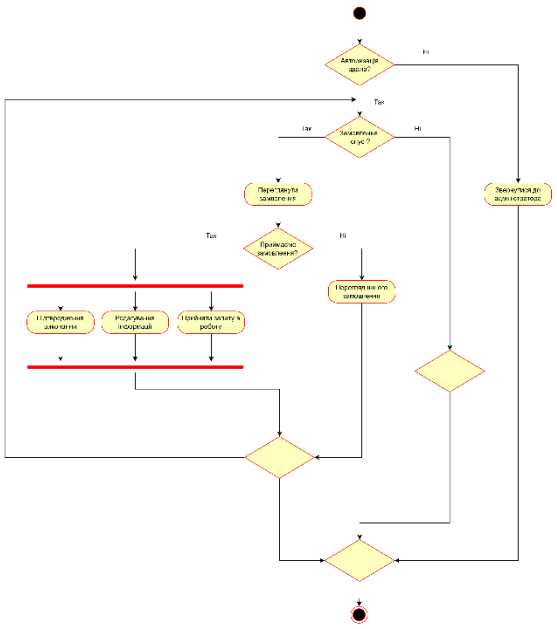

Рис. 3.8 - Діаграма діяльності (Activity Diagram) процесу обробки замовлення в інформаційній системі логістичного забезпечення військових потреб

Діаграма діяльності (рисунок 3.8) деталізує алгоритм обробки замовлення від початку до завершення процесу. Процес починається з автентифікації користувача та перевірки його прав доступу. Після успішної авторизації система

перевіряє, чи є користувач замовником - якщо ні, то він перенаправляється до адміністратора.

Основний потік обробки включає наступні етапи:

1. Створення замовлення - користувач формує заявку на необхідні матеріально-технічні засоби;

2. Паралельна обробка (позначена червоними лініями синхронізації) включає три одночасні процеси:

– Розрахунок інформаційних параметрів замовлення;

– Зміна статусу замовлення в системі;

– Підтвердження замовлення відповідальними особами.

3. Контрольні точки прийняття рішень представлені у вигляді ромбів, які визначають подальший шлях обробки залежно від результатів перевірок.

Діаграма включає початкову (чорне коло) та кінцеву (чорне коло з ободком) активності, що показують межі процесу. Паралельне виконання операцій дозволяє оптимізувати час обробки замовлень, а контрольні точки забезпечують якість та відповідність вимогам на кожному етапі процесу.

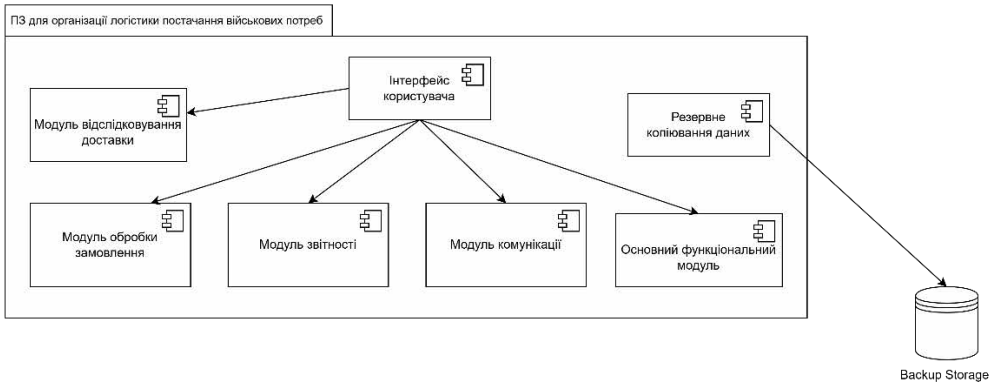

Рис. 3.9 - Діаграма компонентів інформаційної системи логістичного забезпечення військових потреб

Діаграма компонентів (рисунок 3.9) відображає архітектуру системи на рівні програмних компонентів та їх залежностей. Система організована навколо

центрального Інтерфейсу користувача, який забезпечує взаємодію з усіма функціональними модулями:

Основні компоненти системи:

– Модуль відслідковування доставки - контролює стан поставок та логістичні маршрути (виділений синім кольором як ключовий компонент);

– Модуль обробки замовлення - управляє життєвим циклом заявок на матеріально-технічні засоби;

– Модуль звітності - генерує аналітичні звіти та статистику використання ресурсів;

– Модуль комунікацій - забезпечує систему сповіщень та обміну повідомленнями;

– Основний функціональний модуль - координує загальну бізнес-логіку системи;

– Резервне копіювання даних - забезпечує створення та управління backup'ами.

Зовнішні компоненти:

– Backup Storage - зовнішнє сховище для довгострокового зберігання резервних копій.

Компоненти взаємодіють через чітко визначені інтерфейси, що забезпечує слабку зв'язаність та високу згуртованість архітектури. Централізований інтерфейс користувача дозволяє уніфіковано управляти всіма підсистемами, а модульна структура забезпечує можливість незалежного розвитку та тестування кожного компонента.

Діаграма компонентів (Component Diagram) відображає архітектуру системи на рівні програмних компонентів та їх залежностей. Основними компонентами є Web Interface (веб-інтерфейс користувача), API Gateway (шлюз для зовнішніх інтеграцій), Order Management Service (сервіс управління замовленнями), User Management Service (сервіс управління користувачами),

Notification Service (сервіс сповіщень), Reporting Service (сервіс звітності) та Database Access Layer (рівень доступу до даних).

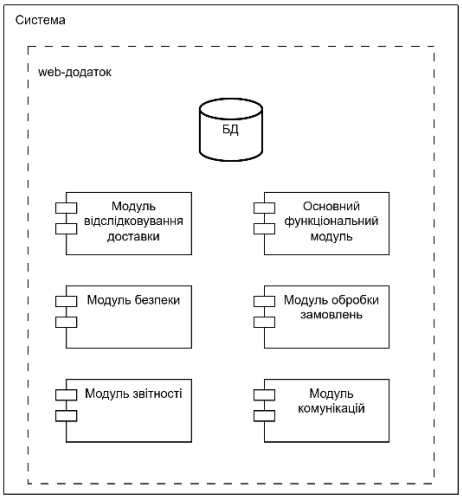

Рис. 3.10 - Діаграма компонентів інформаційної системи логістичного забезпечення військових потреб

Як показано на рисунку 3.10, архітектура системи складається з шести основних функціональних модулів, які взаємодіють через центральну базу даних. Кожен модуль має чітко визначені обов'язки: основний функціональний модуль координує загальну логіку системи, модуль обробки замовлень управляє життєвим циклом заявок, модуль відслідковування доставки контролює статус поставок, модуль безпеки забезпечує автентифікацію та авторизацію, модуль звітності генерує аналітичну інформацію, а модуль комунікацій обробляє сповіщення користувачів.

Компоненти взаємодіють через чітко визначені інтерфейси, що забезпечує слабку зв'язаність та високу згуртованість архітектури. Діаграма також показує зовнішні залежності, такі як SMTP сервер для відправки email та зовнішні API постачальників.

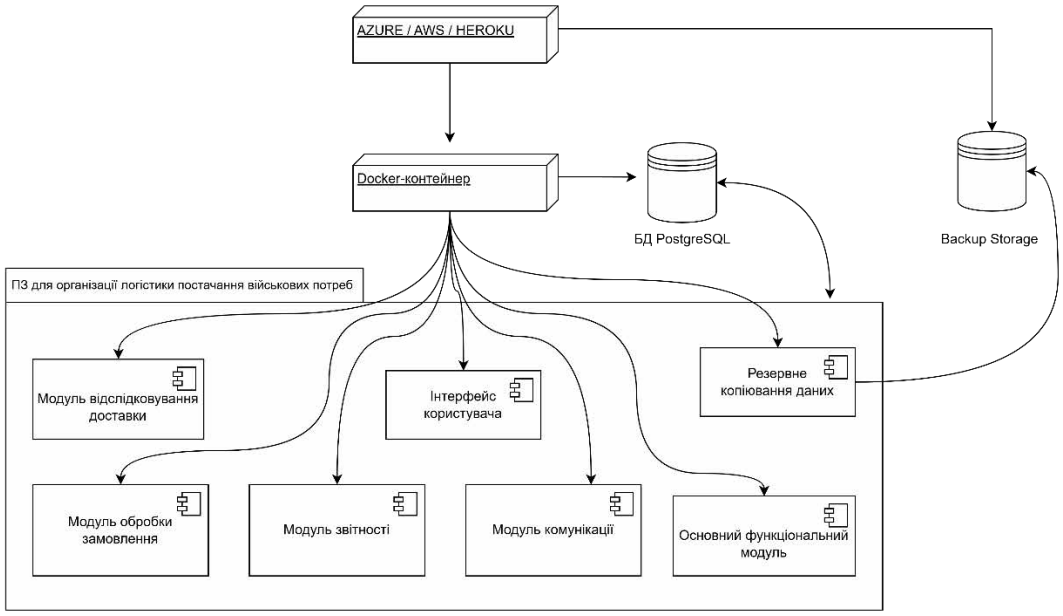

Рис. 3.11 - Діаграма розгортання інформаційної системи логістичного забезпечення військових потреб

Діаграма розгортання (рисунок 3.11) відображає фізичну архітектуру системи та розподіл програмних компонентів по апаратних вузлах. Система розгортається в хмарному середовищі AZURE/AWS/HEROKU з наступними компонентами:

Рівень інфраструктури:

– Docker-контейнер - основне середовище виконання додатку;

– БД PostgreSQL - основна база даних для зберігання операційних даних;

– Backup Storage - система резервного копіювання для забезпечення надійності.

– Рівень додатку (ПЗ для організації логістики постачання військових потреб):

– Інтерфейс користувача - веб-інтерфейс для взаємодії з системою;

– Резервне копіювання даних - модуль для автоматичного створення backup'ів;

– Модуль відслідковування доставки - відстеження статусу поставок;

– Модуль обробки замовлення - обробка заявок на матеріально-технічні засоби;

– Модуль звітності - генерація аналітичних звітів;

– Модуль комунікацій - система сповіщень та повідомлень;

– Основний функціональний модуль - центральна бізнес-логіка системи.

Архітектура забезпечує високу доступність через хмарне розгортання, масштабованість через контейнеризацію та надійність через автоматичне резервне копіювання. Всі компоненти взаємодіють через внутрішню мережу контейнера, що забезпечує безпеку та ізоляцію системи.

3.2 Розробка інформаційного забезпечення

3.2.1 Логічна модель даних

Логічна модель даних системи логістичного забезпечення військових потреб базується на реляційному підході з нормалізацією до третьої нормальної форми для забезпечення цілісності даних та мінімізації надмірності.

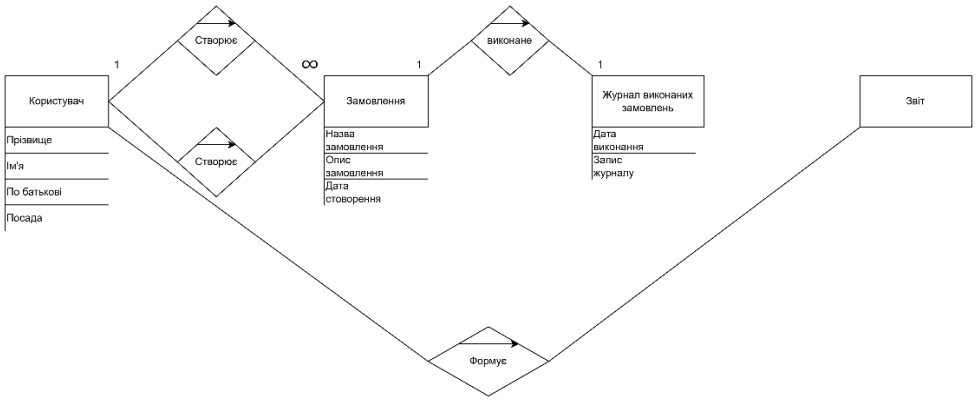

Рис. 3.12 - Логічна модель даних (ERD) інформаційної системи логістичного забезпечення військових потреб

Логічна модель даних (рисунок 3.12) відображає основні сутності системи та їх взаємозв'язки. Модель включає чотири ключові сутності:

Користувач - містить персональну інформацію користувачів системи з атрибутами:

– Прізвище, Ім'я, По батькові - персональні дані;

– Посада - службове становище користувача.

Замовлення - центральна сутність для зберігання заявок з атрибутами:

– Назва замовлення - короткий опис заявки;

– Опис замовлення - детальна специфікація потреб;

– Дата створення - час ініціювання заявки.

Журнал виконаних замовлень - відстежує історію обробки з атрибутами:

– Дата виконання - час завершення замовлення;

– Запис журналу - деталі виконання.

Звіт - сутність для зберігання згенерованих аналітичних документів.

Зв'язки між сутностями:

– Користувач може створювати багато замовлень (зв'язок 1:∞);

– Кожне замовлення може мати один запис у журналі виконання (зв'язок 1:1);

– Замовлення можуть формувати звіти (зв'язок багато-до-багатьох через асоціативну сутність).

Така структура забезпечує повну відстежуваність процесів від створення заявки до її виконання та звітності.

Рис. 3.13 - Фізична модель бази даних інформаційної системи логістичного забезпечення військових потреб

Фізична модель бази даних (рисунок 3.12) деталізує структуру таблиць з конкретними типами даних та зв'язками через зовнішні ключі (FK). Модель включає чотири основні таблиці:

Таблиця "Користувач":

– ID_Користувача (int) - первинний ключ для унікальної ідентифікації;

– Прізвище, Ім'я, По батькові (string) - текстові поля для персональних даних;

– Посада (string) - службове становище користувача.

Таблиця "Замовлення" (центральна сутність):

– ID_Замовлення (int) - первинний ключ;

– ID_Користувача (int, FK) - зовнішній ключ для зв'язку з користувачем;

– Назва замовлення, Опис замовлення (string) - текстові характеристики заявки;

– Статус замовлення (string) - поточний стан обробки;

– Дата створення замовлення, Дата виконання замовлення (date) - часові мітки.

Таблиця "Журнал виконаних замовлень":

– ID_Журналу (int) - первинний ключ;

– ID_Замовлення, ID_Користувача (int, FK) - зовнішні ключі для трасування;

– Запис журналу (string) - деталі виконання операції.

Таблиця "Звіт":

– ID_Звіту (int) - первинний ключ;

– ID_Замовлення, ID_Журналу, ID_Користувача (int, FK) - зовнішні ключі для агрегації даних.

Зв'язки реалізовані через зовнішні ключі: користувач може мати багато замовлень (1:багато), кожне замовлення має запис у журналі (1:1), звіти агрегують дані з усіх таблиць (багато:багато). Така структура забезпечує цілісність даних та ефективні запити для отримання аналітичної інформації.

Центральною сутністю моделі є таблиця Orders, яка зберігає інформацію про замовлення з атрибутами order_id (первинний ключ), user_id (зовнішній ключ на користувача), order_date (дата створення), status (статус замовлення), priority (пріоритет), total_amount (загальна сума), delivery_date (планована дата доставки) та comments (коментарі). Ця таблиця пов'язана з таблицею Users через зовнішній ключ для ідентифікації автора замовлення та з таблицею OrderItems для деталізації позицій замовлення.

Таблиця Users містить інформацію про користувачів системи з атрибутами user_id (первинний ключ), username (унікальне ім'я користувача), password_hash (хеш пароля), email (електронна пошта), first_name, last_name (ім'я та прізвище), position (посада), unit_id (зовнішній ключ на військовий підрозділ), role_id (зовнішній ключ на роль), created_date (дата реєстрації) та is_active (статус активності). Таблиця пов'язана з таблицями Roles для визначення прав доступу та Units для ідентифікації військових підрозділів. Система ролей включає ролі "Soldier", "Officer", "Commander", "Logistics Coordinator", "Administrator" з різними рівнями доступу до функціональності системи.

Каталог матеріально-технічних засобів реалізований через таблицю Materials з атрибутами material_id (первинний ключ), name (назва), description (опис), category_id (зовнішній ключ на категорію), specification (технічні

характеристики), unit_of_measure (одиниця вимірювання), weight (вага), storage_conditions (умови зберігання) та created_date (дата додавання). Таблиця Categories визначає ієрархічну структуру категорій матеріалів з атрибутами category_id, parent_category_id (для створення дерева категорій), name та description. Таблиця OrderItems пов'язує замовлення з матеріалами через атрибути order_id, material_id, quantity (кількість), unit_price (ціна за одиницю) та line_total (сума по позиції).

Інформація про постачальників зберігається в таблиці Suppliers з атрибутами supplier_id (первинний ключ), company_name (назва компанії), contact_person (контактна особа), phone, email (контактні дані), address (адреса), tax_id (податковий номер), rating (рейтинг), contract_number (номер договору) та is_approved (статус акредитації). Зв'язок між постачальниками та матеріалами реалізований через таблицю SupplierMaterials з атрибутами supplier_id, material_id, price (ціна), availability (наявність), delivery_time (термін поставки) та last_updated (дата оновлення інформації). Це дозволяє відстежувати, які матеріали може постачати кожен постачальник та за якими цінами.

Аудит та історія змін забезпечуються через таблиці AuditLog для загального логування дій користувачів та OrderHistory для відстеження змін статусів замовлень. Таблиця AuditLog містить атрибути log_id, user_id, action_type (тип дії), table_name (назва таблиці), record_id (ідентифікатор запису), old_values та new_values (старі та нові значення), timestamp (час дії) та ip_address (IP адреса користувача). OrderHistory зберігає історію змін статусів замовлень з атрибутами history_id, order_id, old_status, new_status, changed_by (користувач, який змінив), change_date та comments. Ця структура забезпечує повну відстежуваність всіх операцій в системі, що критично важливо для військових додатків.

3.2.2 Вибір системи управління базою даних

Вибір PostgreSQL як основної системи управління базою даних для системи логістичного забезпечення військових потреб обґрунтовується її унікальними характеристиками, які ідеально відповідають вимогам військових інформаційних систем. PostgreSQL є повністю відкритою СУБД з високим рівнем відповідності стандартам SQL, що забезпечує переносимість додатків та зменшує ризики залежності від конкретного постачальника. Система має репутацію надійної та стабільної СУБД, здатної працювати під високим навантаженням без втрати продуктивності, що критично важливо для військових систем, які повинні функціонувати безперервно навіть в екстремальних умовах [28, с. 55-62].

Механізми безпеки PostgreSQL включають row-level security, що дозволяє налаштовувати доступ до даних на рівні окремих записів залежно від ролі користувача та контексту запиту. Це особливо важливо для військових систем, де різні користувачі повинні мати доступ лише до тієї інформації, яка відповідає їх рівню допуску та функціональним обов'язкам. Система підтримує SSL/TLS шифрування для захисту даних під час передачі, прозоре шифрування даних на рівні стовпців та таблиць, а також інтеграцію з зовнішніми системами автентифікації через LDAP, Kerberos та інші протоколи, що забезпечує централізоване управління користувачами.

Продуктивність PostgreSQL забезпечується через потужну систему індексування, що включає B-tree, Hash, GiST, SP-GiST, GIN та BRIN індекси для різних типів запитів. Підтримка часткових індексів дозволяє індексувати лише підмножину даних, що відповідає певним умовам, значно економлячи дисковий простір та підвищуючи швидкість оновлень [29, с. 96-116]. Система має вбудований планувальник запитів з можливістю детального аналізу та оптимізації виконання складних аналітичних запитів. Підтримка паралельного виконання запитів дозволяє ефективно використовувати багатоядерні процесори для обробки великих обсягів даних.

Масштабованість PostgreSQL забезпечується через підтримку реплікації master-slave та master-master конфігурацій, що дозволяє розподіляти

навантаження читання між декількома серверами та забезпечувати високу доступність системи. Система підтримує логічну реплікацію для селективного копіювання даних та streaming реплікацію для мінімізації затримок синхронізації. Можливості горизонтального партиціонування (sharding) через розширення Postgres-XL або Citus дозволяють масштабувати систему для обробки петабайтів даних. Вбудована підтримка JSON та JSONB типів даних дозволяє ефективно зберігати та запитувати напівструктуровані дані без втрати переваг реляційної моделі [30].

Порівняння з альтернативними рішеннями показало, що MySQL, незважаючи на високу продуктивність для простих запитів, має обмеження у функціональності та відповідності SQL стандартам. Microsoft SQL Server пропонує потужні корпоративні можливості, але має високу вартість ліцензування та прив'язку до екосистеми Microsoft. Oracle Database забезпечує найвищу продуктивність для критично важливих додатків, але її вартість та складність адміністрування роблять її недоступною для більшості проектів. MongoDB та інші NoSQL рішення розглядаються як доповнення до PostgreSQL для специфічних задач зберігання неструктурованих даних, але не можуть замінити реляційну СУБД для основних бізнес-процесів логістичної системи.

3.2.3 Створення інформаційної бази

Процес створення інформаційної бази системи логістичного забезпечення військових потреб починається з встановлення та конфігурації PostgreSQL сервера з оптимальними налаштуваннями для військового застосування. Конфігурація включає налаштування параметрів продуктивності, таких як shared_buffers (25% від доступної оперативної пам'яті), work_mem (розрахований на основі кількості одночасних з'єднань), maintenance_work_mem (для операцій обслуговування), effective_cache_size (75% від загальної пам'яті системи) та checkpoint_segments для оптимізації операцій запису. Налаштування безпеки включають обмеження доступу через pg_hba.conf, вимкнення непотрібних

розширень, налаштування SSL сертифікатів та конфігурацію логування всіх операцій для аудиту.

Створення структури бази даних здійснюється через міграції Django ORM, що забезпечує версіонування схеми та можливість відкату змін у разі необхідності. Початкова міграція створює всі основні таблиці з відповідними індексами, обмеженнями цілісності та тригерами для аудиту. Кожна таблиця включає стандартні поля created_at та updated_at для відстеження часу створення та модифікації записів. Зовнішні ключі налаштовуються з каскадними операціями або обмеженнями залежно від бізнес-логіки: ON DELETE CASCADE для залежних записів та ON DELETE RESTRICT для критичних зв'язків, що запобігають випадковому видаленню важливих даних.

Індексування оптимізується для типових запитів системи логістичного забезпечення з створенням композитних індексів для часто використовуваних комбінацій полів. Наприклад, створюється індекс на (user_id, order_date) для швидкого пошуку замовлень конкретного користувача за період, індекс на (status, priority, created_date) для вибірки замовлень за статусом та пріоритетом, індекс на (material_id, supplier_id) для швидкого знаходження постачальників конкретних матеріалів. Часткові індекси створюються для активних записів (WHERE is_active = true) та незавершених замовлень (WHERE status NOT IN ('completed', 'cancelled')) для економії дискового простору та підвищення продуктивності.

Наповнення бази даних початковими даними включає створення довідників категорій матеріально-технічних засобів відповідно до військових стандартів класифікації, базового каталогу матеріалів з їх характеристиками, структури військових підрозділів та ієрархії командування, а також системи ролей користувачів з відповідними правами доступу. Довідники заповнюються через спеціальні fixtures Django або SQL скрипти, що дозволяє легко відтворювати структуру в різних середовищах [31]. Тестові дані генеруються за допомогою Factory Boy для створення реалістичних наборів даних для тестування продуктивності та функціональності системи.

Процедури резервного копіювання та відновлення налаштовуються з використанням pg_dump для логічного резервування та pg_basebackup для фізичного копіювання. Автоматизація резервування здійснюється через cron jobs з щоденними повними копіями та годинними інкрементальними копіями за допомогою Write-Ahead Logging (WAL). Резервні копії зберігаються на віддалених серверах з шифруванням та перевіркою цілісності. Процедури відновлення регулярно тестуються в ізольованому середовищі для забезпечення можливості швидкого відновлення системи у разі збоїв. Моніторинг бази даних налаштовується через pg_stat_statements для аналізу продуктивності запитів та Prometheus з postgres_exporter для збору метрик стану системи.

3.3 Розробка прикладного програмного забезпечення

3.3.1 Організаційна структура програмного забезпечення

Архітектурна організація програмного забезпечення системи логістичного забезпечення військових потреб базується на принципах модульності, слабкої зв'язаності та високої згуртованості компонентів. Система побудована за багаторівневою архітектурою з чітким розділенням відповідальності між шарами: шар представлення (Presentation Layer), шар бізнес-логіки (Business Logic Layer), шар доступу до даних (Data Access Layer) та шар інфраструктури (Infrastructure Layer). Така організація забезпечує можливість незалежної розробки, тестування та супроводу кожного шару, а також спрощує масштабування та модифікацію системи без впливу на інші компоненти.

Шар представлення включає веб-інтерфейс користувача, мобільні додатки та API для інтеграції з зовнішніми системами. Веб-інтерфейс реалізований як Single Page Application (SPA) з використанням сучасних веб-технологій для забезпечення інтуїтивного користувацького досвіду. RESTful API забезпечує стандартизований доступ до функціональності системи для мобільних клієнтів та інтеграції з зовнішніми системами. Шар також включає компоненти

автентифікації та авторизації, які забезпечують безпечний доступ до ресурсів системи відповідно до ролей та прав користувачів (рис.3.1-3.4).

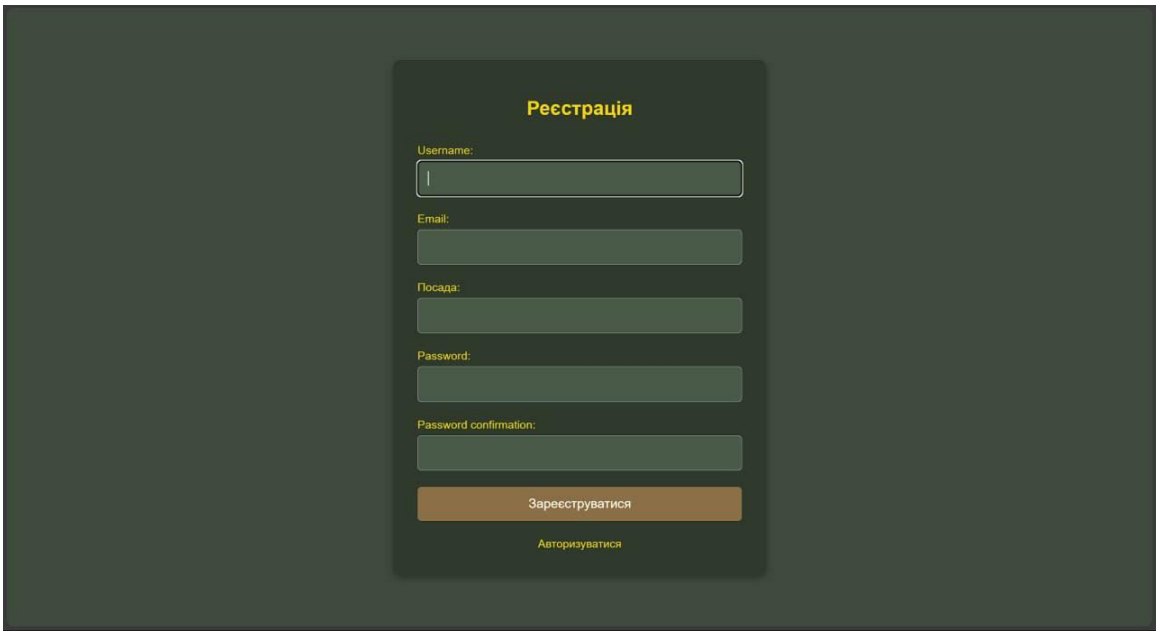

Рис. 3.1 - Форма реєстрації у веб-інтерфейсі системи логістичного забезпечення

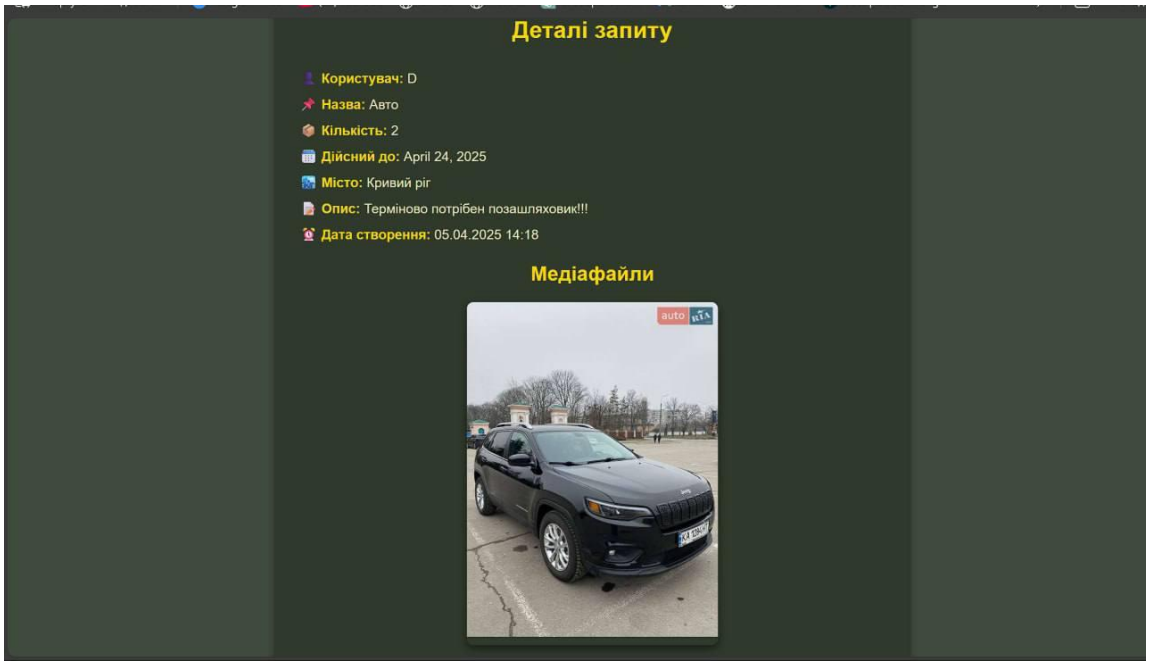

Рис. 3.2 - Інтерфейс користувача системи логістичного забезпечення з відображенням деталей запиту

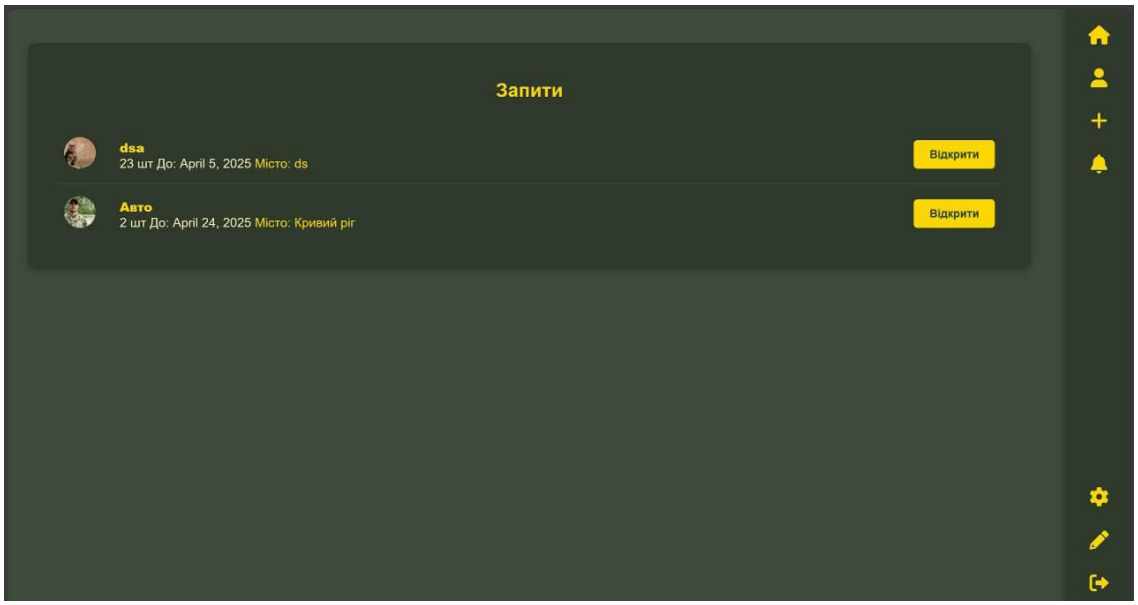

Рис. 3.3 - Список повідомлень у веб-інтерфейсі системи логістичного забезпечення

Рис. 3.4 - Головний екран веб-інтерфейсу системи логістичного забезпечення

Шар бізнес-логіки організований за доменним підходом (Domain-Driven Design) з виділенням окремих сервісів для кожної функціональної області: OrderService для управління замовленнями, UserService для управління користувачами, MaterialService для роботи з каталогом матеріалів, SupplierService для взаємодії з постачальниками та ReportingService для генерації

звітів. Кожен сервіс інкапсулює бізнес-правила та логіку обробки даних, забезпечуючи централізацію бізнес-логіки та її повторне використання різними компонентами системи. Міжсервісна взаємодія здійснюється через чітко визначені інтерфейси з використанням паттернів Observer та Command для забезпечення слабкої зв'язаності.

Шар доступу до даних реалізований за паттерном Repository з використанням Django ORM для абстрагування від специфіки конкретної СУБД. Кожна доменна сутність має відповідний Repository клас, який інкапсулює всі операції з базою даних та забезпечує єдиний інтерфейс для доступу до даних. Шар включає компоненти кешування для оптимізації продуктивності часто запитуваних даних та механізми lazy loading для мінімізації навантаження на базу даних. Транзакційна цілісність забезпечується через використання Unit of Work паттерну та декларативних транзакцій Django.

Шар інфраструктури містить компоненти, які забезпечують роботу системи, але не є частиною основної бізнес-логіки: система логування з використанням структурованих логів, система моніторингу з метриками продуктивності, система сповіщень для інформування користувачів про зміни статусів замовлень, система інтеграції з зовнішніми сервісами через адаптери та планувальник завдань для виконання фонових операцій. Конфігурація системи централізована через файли налаштувань Django з можливістю перевизначення через змінні середовища для різних оточень розгортання. Контейнеризація через Docker забезпечує консистентність середовища розгортання та спрощує процеси CI/CD.

3.3.2 Вибір інструментарію для створення ППЗ

Django фреймворк обрано як основний інструмент для створення серверної частини прикладного програмного забезпечення завдяки його філософії "batteries included" та потужним вбудованим механізмам безпеки. Django забезпечує автоматичний захист від основних типів веб-атак включаючи CSRF,

XSS, SQL-ін'єкції та clickjacking через вбудовані middleware компоненти та template engine з автоматичним екрануванням. Система автентифікації та авторизації Django дозволяє гнучко налаштовувати права доступу на рівні користувачів, груп та об'єктів, що критично важливо для військових систем з різними рівнями доступу до інформації.

Django REST Framework (DRF) використовується для створення RESTful API, що забезпечує стандартизований доступ до функціональності системи для мобільних клієнтів та інтеграції з зовнішніми системами. DRF надає потужні засоби серіалізації даних, пагінації результатів, фільтрації та пошуку, а також автоматичну генерацію документації API. Система дозволів DRF інтегрується з Django authentication для забезпечення контролю доступу на рівні API endpoints. Підтримка різних форматів відповідей (JSON, XML, CSV) забезпечує гнучкість інтеграції з різними типами клієнтських додатків.

Celery з Redis як message broker обрано для обробки асинхронних завдань та фонових процесів, таких як генерація звітів, відправка сповіщень та інтеграція з зовнішніми системами. Celery забезпечує надійну обробку завдань з можливістю retry при збоях, планування періодичних завдань через Celery Beat та масштабування через додавання worker процесів. Redis використовується не лише як message broker, але й як кеш для часто запитуваних даних, що значно підвищує продуктивність системи. Flower надає веб-інтерфейс для моніторингу стану Celery worker'ів та завдань.

Для фронтенд розробки використовується комбінація HTML5, CSS3 з препроцесором SASS та JavaScript з бібліотекою Bootstrap для забезпечення адаптивного дизайну. jQuery застосовується для DOM маніпуляцій та AJAX запитів до API. Chart.js використовується для створення інтерактивних графіків та діаграм в аналітичних звітах. DataTables забезпечує потужні можливості для відображення та фільтрації табличних даних з підтримкою серверної пагінації для великих наборів даних. Веб-інтерфейс оптимізований для роботи на різних пристроях від десктопних комп'ютерів до планшетів та смартфонів (рис.3.5).

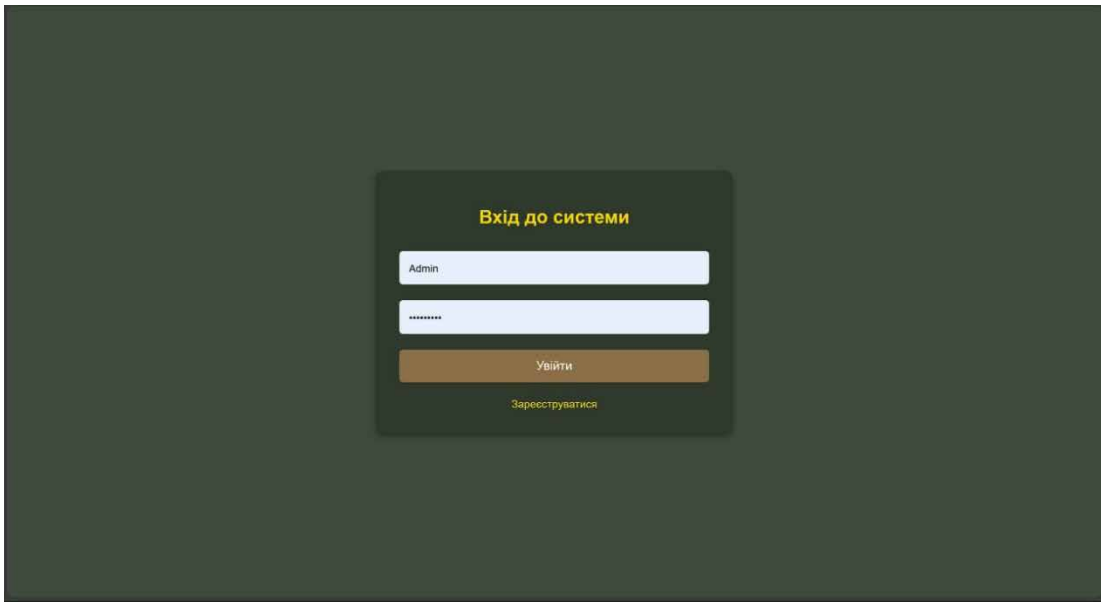

Рис. 3.5 - Зразок інтерфейсу логістичної системи для автентифікації користувачів

Docker та Docker Compose використовуються для контейнеризації додатку та його залежностей, що забезпечує консистентність середовища розгортання та спрощує процеси розгортання в різних оточеннях. Gunicorn обрано як WSGI сервер для production середовища з Nginx як reverse proxy для обслуговування статичних файлів та балансування навантаження. PostgreSQL використовується як основна база даних з pgBouncer для connection pooling. Система моніторингу базується на Prometheus для збору метрик, Grafana для візуалізації та ELK stack (Elasticsearch, Logstash, Kibana) для централізованого логування та аналізу логів.

3.3.3 Розробка алгоритмів та програмних модулів

Алгоритм обробки замовлень є центральним компонентом системи та реалізований як багоетапний процес з проміжними станами та можливістю відкату на попередні етапи у разі виявлення помилок. Процес починається з валідації вхідних даних замовлення, включаючи перевірку автентичності користувача, відповідності матеріалів каталогу, коректності кількостей та наявності бюджету. Алгоритм використовує паттерн Chain of Responsibility для

послідовної обробки замовлення різними валідаторами: UserValidator перевіряє права користувача, MaterialValidator перевіряє існування та доступність матеріалів, BudgetValidator контролює дотримання бюджетних обмежень, PriorityValidator визначає пріоритет замовлення на основі критичності матеріалів та терміновості потреби.

Модуль планування постачань реалізує алгоритм оптимізації, який враховує множину критеріїв включаючи вартість, терміни доставки, надійність постачальників та пріоритети замовлень. Алгоритм базується на методі багатокритеріальної оптимізації з використанням зваженої суми критеріїв, де ваги визначаються на основі поточної операційної ситуації. Для кожного матеріалу алгоритм аналізує всіх доступних постачальників, розраховує інтегральну оцінку кожного варіанту та обирає оптимальний. Алгоритм також враховує обмеження на максимальні обсяги поставок від одного постачальника для розподілення ризиків та забезпечення надійності постачання.

Система сповіщень реалізована через модуль NotificationService, який використовує паттерн Observer для відстеження змін станів замовлень та автоматичної відправки сповіщень зацікавленим сторонам. Модуль підтримує різні канали сповіщень включаючи email, SMS та push-сповіщення в мобільних додатках. Алгоритм визначення адресатів сповіщень базується на ролях користувачів, їх відношенні до конкретного замовлення та налаштуваннях персональних переваг. Система також включає механізм throttling для запобігання спаму та batch processing для оптимізації відправки великої кількості сповіщень.

Модуль звітності реалізує алгоритми агрегації та аналізу даних для генерації різних типів звітів від оперативних до стратегічних. Алгоритм генерації звітів використовує паттерн Template Method для стандартизації процесу створення звітів з можливістю кастомізації для специфічних типів звітів. Для оптимізації продуктивності складних аналітичних запитів використовуються материализовані представлення (materialized views) в PostgreSQL, які автоматично оновлюються через тригери при зміні базових даних. Алгоритм

кешування звітів враховує частоту запитів та актуальність даних для оптимального використання ресурсів [32].

Модуль інтеграції з зовнішніми системами реалізує адаптери для різних типів зовнішніх API з підтримкою різних протоколів та форматів даних. Алгоритм синхронізації даних використовує паттерн Saga для забезпечення консистентності даних при взаємодії з декількома зовнішніми системами. Модуль включає механізми retry з експоненційним backoff для обробки тимчасових збоїв мережі, circuit breaker для захисту від перевантаження зовнішніх систем та timeout конфігурації для уникнення блокування системи при повільних відповідях. Логування всіх операцій інтеграції забезпечує можливість діагностики та відновлення після збоїв.

3.4 Висновки до розділу

Практична реалізація інформаційної системи логістичного забезпечення військових потреб продемонструвала ефективність обраного підходу до моделювання предметної області та архітектурних рішень. Розробка контекстних та функціональних діаграм дозволила чітко визначити межі системи, основних акторів та їх взаємодію з функціональними компонентами. UML-діаграми забезпечили детальне моделювання структурних та поведінкових аспектів системи, включаючи діаграми класів для визначення основних сутностей, діаграми послідовності для моделювання бізнес-процесів та діаграми компонентів для архітектурного планування. Такий комплексний підхід до моделювання створив надійну основу для подальшої розробки та забезпечив можливість валідації архітектурних рішень на ранніх етапах проекту.

Розробка інформаційного забезпечення на базі PostgreSQL з нормалізованою логічною моделлю даних забезпечила цілісність та консистентність інформації при мінімізації надмірності. Створена структура бази даних з центральними таблицями Orders, Users, Materials та їх взаємозв'язками ефективно моделює всі аспекти логістичних процесів від

управління користувачами до відстеження історії замовлень. Система індексування оптимізована для типових запитів системи, а механізми аудиту через таблиці AuditLog та OrderHistory забезпечують повну відстежуваність операцій, критично важливу для військових систем. Налаштування резервного копіювання, реплікації та моніторингу гарантує високу доступність та надійність системи.

Організаційна структура програмного забезпечення за багаторівневою архітектурою з чітким розділенням відповідальності між шарами забезпечила модульність, масштабованість та можливість незалежного розвитку компонентів. Вибір Django як основного фреймворку з екосистемою супутніх інструментів (DRF, Celery, Redis) виявився оптимальним для швидкої розробки безпечних веб-додатків з потужними можливостями інтеграції. Контейнеризація через Docker та налаштування CI/CD процесів забезпечили консистентність середовища розгортання та автоматизацію процесів доставки коду.

Розроблені алгоритми та програмні модули реалізують складну бізнес-логіку логістичних процесів з використанням сучасних паттернів проектування та архітектурних підходів. Алгоритм обробки замовлень з багатоетапною валідацією, модуль планування з багатокритеріальною оптимізацією та система сповіщень з підтримкою різних каналів забезпечують повну функціональність, необхідну для ефективного управління військовою логістикою. Інтеграція з зовнішніми системами через адаптери з надійними механізмами обробки збоїв гарантує стабільну роботу системи навіть при проблемах із зовнішніми сервісами. Комплексне тестування всіх компонентів та автоматизація процесів розгортання створили передумови для успішного впровадження системи в реальних умовах експлуатації.

РОЗДІЛ 4. ВПРОВАДЖЕННЯ ТА ЕКСПЛУАТАЦІЯ СИСТЕМИ

4.1 Тестування розробленої інформаційної системи

Комплексна стратегія тестування інформаційної системи логістичного забезпечення військових потреб включає кілька рівнів перевірки функціональності, продуктивності та безпеки системи. Модульне тестування здійснюється за допомогою pytest фреймворку з покриттям коду не менше 85% для критичних модулів системи. Кожен сервісний клас, утиліта та алгоритм покривається індивідуальними тестами, які перевіряють як позитивні сценарії роботи, так і обробку помилкових ситуацій. Автоматизовані тести запускаються при кожному commit через GitHub Actions, що забезпечує раннє виявлення регресій та підтримання якості коду на високому рівні протягом всього життєвого циклу розробки.

Інтеграційне тестування фокусується на перевірці взаємодії між різними компонентами системи та зовнішніми сервісами. Тести перевіряють коректність роботи API endpoints, правильність серіалізації/десеріалізації даних, функціонування системи автентифікації та авторизації, а також інтеграцію з базою даних PostgreSQL. Для тестування використовуються тестові бази даних з реалістичними наборами даних, створеними за допомогою Factory Boy. Особлива увага приділяється тестуванню транзакційної цілісності при одночасній роботі декількох користувачів та обробці конкурентних операцій над спільними ресурсами.

Функціональне тестування користувацького інтерфейсу здійснюється за допомогою Selenium WebDriver з автоматизацією ключових користувацьких сценаріїв. Тестуються процеси створення та обробки замовлень, генерації звітів, управління користувачами та налаштування системи. Тести виконуються на різних браузерах (Chrome, Firefox, Safari) та пристроях (десктоп, планшет, мобільний) для забезпечення кросплатформенної сумісності. Accessibility

тестування проводиться за допомогою axe-core для забезпечення відповідності стандартам WCAG 2.1, що критично важливо для військових систем, доступних користувачам з різними потребами.

Навантажувальне тестування виконується за допомогою Locust для моделювання реального навантаження на систему з метою перевірки її продуктивності та масштабованості. Тести симулюють роботу до 1000 одночасних користувачів, які виконують типові операції створення замовлень, перегляду каталогу матеріалів та генерації звітів. Результати показали, що система здатна обробляти до 500 запитів на секунду з середнім часом відгуку 1.2 секунди для стандартних операцій та 8.5 секунд для складних аналітичних звітів. Профілювання коду за допомогою Django Debug Toolbar дозволило ідентифікувати та оптимізувати вузькі місця в продуктивності [33].

Тестування безпеки включає автоматизовані сканування за допомогою OWASP ZAP для виявлення вразливостей веб-додатку, статичний аналіз коду за допомогою Bandit для Python коду та перевірку залежностей за допомогою Safety. Проводяться пентести з імітацією атак SQL-ін'єкції, XSS, CSRF та спроб несанкціонованого доступу до ресурсів. Тестування автентифікації включає перевірку стійкості паролів, правильності роботи сесій, захисту від brute force атак та коректності розмежування доступу. Результати показали відсутність критичних вразливостей та відповідність системи стандартам безпеки для військових інформаційних систем.

4.2 Вимоги до апаратного та програмного забезпечення

Мінімальні вимоги до серверного обладнання для розгортання системи логістичного забезпечення військових потреб визначаються на основі результатів навантажувального тестування та планованої кількості користувачів. Для підтримки до 500 одночасних користувачів рекомендується сервер з процесором не менше 8 ядер (Intel Xeon або AMD EPYC), 32 ГБ оперативної пам'яті DDR4, SSD накопичувачем об'ємом 1 ТБ для операційної системи та

додатків, а також додатковим сховищем 5 ТБ для бази даних та файлового архіву. Мережевий інтерфейс повинен підтримувати швидкість не менше 1 Гбіт/с для забезпечення швидкого доступу користувачів та синхронізації з резервними серверами.

Рекомендовані характеристики для production середовища включають кластер з трьох серверів для забезпечення високої доступності та можливості обслуговування без зупинки системи. Основний сервер додатків повинен мати 16 ядер процесора, 64 ГБ RAM та SSD накопичувачі в RAID 10 конфігурації. Сервер бази даних потребує аналогічної конфігурації з додатковим акцентом на швидкість дискової підсистеми - NVMe SSD з швидкістю читання не менше 3000 IOPS. Балансувальник навантаження може працювати на менш потужному обладнанні з 4 ядрами та 16 ГБ RAM, але повинен мати резервні мережеві підключення для уникнення single point of failure.

Операційна система Ubuntu Server 20.04 LTS або CentOS 8 рекомендується як стабільна та надійна основа для розгортання системи з довгостроковою підтримкою безпеки. Docker версії 20.10 або новіше необхідний для контейнеризації додатків, а Docker Compose 1.28+ для оркестрації мультиконтейнерних конфігурацій. PostgreSQL 13 або 14 версії забезпечує необхідну функціональність та продуктивність для роботи з логістичними даними. Redis 6.2+ використовується для кешування та черг повідомлень Celery. Nginx 1.18+ служить як reverse proxy та веб-сервер для статичних файлів.

Програмне забезпечення моніторингу включає Prometheus 2.30+ для збору метрик системи та додатків, Grafana 8.0+ для візуалізації даних моніторингу, а також ELK stack (Elasticsearch 7.15+, Logstash 7.15+, Kibana 7.15+) для централізованого управління логами. Система резервного копіювання базується на pg_dump для PostgreSQL та rsync для файлових систем з автоматизацією через cron jobs. SSL сертифікати від довіреного центру сертифікації або Let's Encrypt необхідні для забезпечення безпечного HTTPS трафіку [34].

Клієнтські робочі місця повинні відповідати мінімальним вимогам: процесор Intel Core i3 або AMD Ryzen 3, 8 ГБ оперативної пам'яті, 256 ГБ SSD

накопичувач та мережеве підключення зі швидкістю не менше 100 Мбіт/с. Підтримуються сучасні веб-браузери: Chrome 90+, Firefox 88+, Safari 14+, Edge 90+ з увімкненим JavaScript та підтримкою HTML5. Мобільні пристрої для польового використання повинні працювати під управлінням Android 8.0+ або iOS 13+ з можливістю роботи в режимі обмеженої мережевої зв'язності. Рекомендується використання захищених планшетів військового стандарту з захистом IP67 та розширеним температурним діапазоном для роботи в польових умовах.

4.3 Особливості розгортання та налаштування системи

Процес розгортання системи логістичного забезпечення військових потреб базується на контейнеризованій архітектурі з використанням Docker та Docker Compose для забезпечення консистентності та відтворюваності середовища.

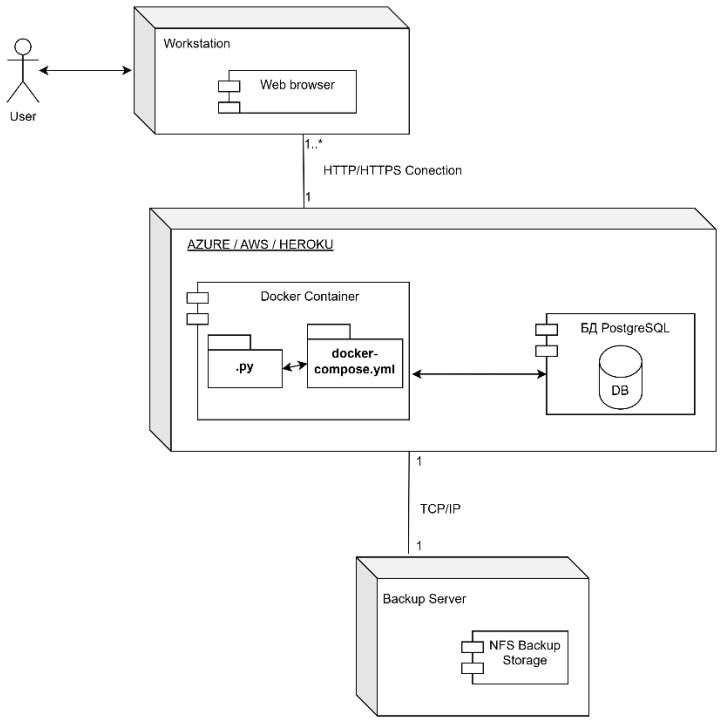

Рис. 4.1 - Архітектура розгортання інформаційної системи логістичного забезпечення військових потреб

Архітектура розгортання системи (рисунок 4.1) демонструє фізичне розміщення компонентів та їх мережеву взаємодію. Система складається з трьох основних рівнів:

Клієнтський рівень (Workstation):

– Користувач працює через веб-браузер на робочій станції;

– З'єднання з сервером відбувається через захищений HTTP/HTTPS протокол.

Серверний рівень (AZURE/AWS/HEROKU):

– Docker Container - основне середовище виконання, що містить:

– Python додаток (.py) - серверна логіка на Django;

– docker-compose.yml - конфігурація оркестрації контейнерів;

– БД PostgreSQL - база даних для зберігання операційних даних;

– Компоненти взаємодіють всередині хмарної інфраструктури.

Резервний рівень (Backup Server):

– Окремий сервер з NFS Backup Storage для зберігання резервних копій;

– З'єднання з основним сервером через TCP/IP протокол.

Така архітектура забезпечує:

– Масштабованість через хмарне розгортання;

– Надійність через відокремлене резервне копіювання;

– Безпеку через HTTPS з'єднання та ізоляцію в контейнерах;

– Портативність через використання Docker контейнерів.

Підготовка до розгортання включає створення окремих контейнерів для веб-додатку Django, бази даних PostgreSQL, кеш-сервера Redis, черг Celery та веб-сервера Nginx. Кожен контейнер має власний Dockerfile з чітко визначеними залежностями та конфігураціями, що забезпечує ізоляцію компонентів та спрощує управління версіями. Docker Compose файл описує всю інфраструктуру додатку, включаючи мережеві з'єднання, volume mounting для персистентних даних та змінні середовища для різних конфігурацій.

Налаштування безпеки розгортання включає конфігурацію firewall правил з обмеженням доступу до портів лише для необхідних сервісів, налаштування SSL/TLS термінації на рівні Nginx з використанням сертифікатів від довіреного центру сертифікації. Всі конфіденційні дані, такі як паролі бази даних, секретні ключі та API токени, зберігаються в зашифрованому вигляді через Docker Secrets або HashiCorp Vault для production середовища. Налаштовується автоматичне оновлення безпеки операційної системи та контейнерів, а також моніторинг безпеки через інструменти типу Falco для виявлення аномальної активності в runtime.

Процедура розгортання починається з підготовки серверного середовища через Ansible playbooks, які автоматизують встановлення необхідного програмного забезпечення, створення користувачів та налаштування базової конфігурації безпеки. Наступним кроком є розгортання додатку через CI/CD pipeline в GitLab, який автоматично будує Docker образи, проводить тестування, публікує образи в Docker Registry та розгортає їх на цільових серверах. Pipeline включає етапи blue-green deployment для мінімізації downtime під час оновлень та автоматичний rollback у разі виявлення проблем після розгортання.

Конфігурація високої доступності реалізується через розгортання декількох інстансів додатку за Load Balancer (HAProxy або Nginx), настройку PostgreSQL реплікації master-slave для розподілення навантаження читання та забезпечення відмовостійкості. Redis налаштовується в режимі Sentinel для автоматичного failover, а Celery workers розгортаються на декількох серверах для паралельної обробки завдань. Моніторинг здоров'я сервісів здійснюється через health checks з автоматичним перезапуском контейнерів у разі збоїв та сповіщенням адміністраторів про критичні проблеми.

Налаштування системи після розгортання включає завантаження початкових даних через Django fixtures, створення адміністративних користувачів, конфігурацію ролей та прав доступу, а також налаштування інтеграції з зовнішніми системами через API ключі та endpoints. Виконується тестування всіх основних функцій системи в production середовищі, включаючи

перевірку процесів автентифікації, створення замовлень, генерації звітів та роботи системи сповіщень. Налаштовується моніторинг продуктивності через Prometheus та Grafana з dashboard для відстеження ключових метрик та автоматичними alerts при перевищенні порогових значень.

4.4 Заходи забезпечення безпеки даних у системі

Система автентифікації та авторизації користувачів базується на багаторівневому підході з використанням сильних паролів, двофакторної автентифікації та ролевої моделі доступу. Всі паролі зберігаються у вигляді хешів з використанням алгоритму PBKDF2 з високим числом ітерацій та унікальними солями для кожного користувача. Двофакторна автентифікація реалізована через TOTP (Time-based One-Time Password) з підтримкою Google Authenticator та аналогічних додатків. Система сесій налаштована з коротким часом життя (30 хвилин) та автоматичним logout при неактивності. Блокування облікових записів після п'яти невдалих спроб входу та логування всіх спроб автентифікації забезпечують захист від brute force атак.

Шифрування даних здійснюється на всіх рівнях системи для забезпечення конфіденційності інформації. Передача даних між клієнтом та сервером захищена через TLS 1.3 з використанням сертифікатів з довжиною ключа не менше 2048 біт. Конфіденційні дані в базі даних зберігаються в зашифрованому вигляді з використанням AES-256 алгоритму шифрування на рівні стовпців. Секретні ключі та паролі зберігаються в спеціалізованих сховищах типу HashiCorp Vault з контролем доступу та ротацією ключів. Резервні копії даних шифруються перед зберіганням на зовнішніх носіях з використанням асиметричного шифрування.

Контроль доступу реалізований через ролеву модель з принципом мінімальних привілеїв, де кожен користувач має доступ лише до тих ресурсів, які необхідні для виконання його функціональних обов'язків. Система підтримує ієрархічні ролі від рядового військовослужбовця до командування з різними

рівнями доступу до функціональності та даних. Row-level security в PostgreSQL забезпечує додатковий рівень контролю доступу на рівні окремих записів залежно від підрозділу користувача та класифікації інформації. Всі дії користувачів логуються в audit trail з зазначенням часу, користувача, типу операції та змінених даних для забезпечення повної відстежуваності.

Захист від кібератак включає комплекс заходів для попередження та виявлення різних типів загроз. Web Application Firewall (WAF) фільтрує HTTP трафік та блокує підозрілі запити, включаючи спроби SQL-ін'єкцій, XSS та інших атак. Система виявлення вторгнень (IDS) моніторить мережевий трафік та системні логи для виявлення аномальної активності. Rate limiting обмежує кількість запитів від одного IP адреси для запобігання DDoS атакам. Регулярне оновлення всіх компонентів системи та залежностей через автоматизовані процеси забезпечує захист від відомих вразливостей.

Політики безпеки та процедури інцидент-менеджменту визначають організаційні заходи для підтримання безпеки системи. Регулярні security audits та penetration testing проводяться зовнішніми спеціалістами для виявлення потенційних вразливостей. Всі співробітники, які мають доступ до системи, проходять обов'язкове навчання з питань інформаційної безпеки [35-36]. План реагування на інциденти безпеки включає процедури ізоляції скомпрометованих систем, аналізу інцидентів, відновлення даних та повідомлення відповідних органів. Резервні копії даних зберігаються в географічно віддалених локаціях з обмеженим доступом та регулярно тестуються на можливість відновлення.

4.5 Рекомендації щодо експлуатації системи

Щоденне адміністрування системи включає моніторинг ключових показників продуктивності, перевірку журналів подій, контроль дискового простору та мережевого трафіку. Адміністратори повинні щоранку перевіряти dashboard в Grafana для аналізу метрик продуктивності, включаючи час відгуку сервера, кількість активних користувачів, навантаження на базу даних та статус

всіх сервісів. Автоматичні алерти налаштовані для критичних ситуацій, але регулярний ручний моніторинг дозволяє виявляти потенційні проблеми до їх ескалації. Рекомендується ведення щоденного журналу адміністратора з фіксацією всіх виконаних операцій, виявлених проблем та вжитих заходів.

Планове технічне обслуговування має проводитися щотижня в неробочий час для мінімізації впливу на користувачів. Процедури включають оновлення системного ПЗ та патчів безпеки, очищення тимчасових файлів та логів, перевірку цілісності резервних копій, аналіз продуктивності запитів до бази даних та оптимізацію індексів при необхідності. Щомісячні процедури обслуговування включають повне резервне копіювання системи, тестування процедур відновлення в ізольованому середовищі, аналіз трендів використання ресурсів та планування масштабування. Квартальне обслуговування включає аудит безпеки, оновлення сертифікатів та комплексну перевірку всіх компонентів системи.

Управління користувачами та доступом вимагає регулярного перегляду прав доступу та дезактивації облікових записів співробітників, які більше не працюють в організації. Рекомендується щомісячна перевірка списку активних користувачів з підтвердженням їх актуального статусу та необхідності доступу до системи. Нові користувачі мають створюватися згідно з затвердженими процедурами з обов'язковим навчанням правилам роботи з системою. Зміна ролей та прав доступу повинна здійснюватися лише на основі письмових заявок з підписом відповідального керівника. Всі зміни в системі управління користувачами логуються для забезпечення повної відстежуваності.

Моніторинг продуктивності та оптимізація системи повинні проводитися постійно для забезпечення стабільної роботи під зростаючим навантаженням. Рекомендується щотижневий аналіз повільних запитів до бази даних через pg_stat_statements з оптимізацією або створенням додаткових індексів при необхідності. Моніторинг використання пам'яті та дискового простору дозволяє планувати апгрейди обладнання [37]. Аналіз логів помилок допомагає виявляти систематичні проблеми та недоліки в коді, які потребують виправлення.

Регулярне тестування продуктивності під навантаженням дозволяє оцінювати готовність системи до пікових періодів використання.

Процедури резервного копіювання та відновлення є критично важливими для забезпечення безперервності роботи системи. Рекомендується автоматизоване щоденне резервне копіювання з перевіркою цілісності та тестування відновлення на щомісячній основі. Резервні копії повинні зберігатися в трьох екземплярах: локально для швидкого відновлення, на віддаленому сервері в тій же локації та в географічно віддаленому центрі обробки даних. Процедури відновлення повинні бути детально документовані з покроковими інструкціями та регулярно тестуватися для забезпечення можливості швидкого відновлення системи у разі збоїв. Час відновлення після катастрофічного збою не повинен перевищувати 4 години для критичних функцій системи.

4.6 Висновки до розділу

Реалізований комплексний підхід до впровадження та експлуатації системи логістичного забезпечення військових потреб продемонстрував ефективність обраних технологічних рішень та архітектурних підходів. Багаторівнева стратегія тестування з покриттям модульного, інтеграційного, функціонального, навантажувального та безпекового тестування забезпечила високу якість системи та її готовність до продуктивної експлуатації. Результати навантажувального тестування підтвердили здатність системи обробляти до 500 запитів на секунду з середнім часом відгуку 1.2 секунди, що відповідає вимогам військових логістичних систем з інтенсивним використанням.

Визначені вимоги до апаратного та програмного забезпечення базуються на реальних потребах системи та результатах тестування продуктивності, що гарантує стабільну роботу під планованим навантаженням. Контейнеризована архітектура розгортання через Docker забезпечує консистентність середовища, спрощує масштабування та обслуговування системи. Автоматизація процесів розгортання через CI/CD пайплайни мінімізує ризики людських помилок та

забезпечує швидке впровадження оновлень з можливістю rollback у разі необхідності.

Комплексні заходи забезпечення безпеки, включаючи багаторівневу автентифікацію, шифрування даних, контроль доступу та захист від кібератак, відповідають високим стандартам військових інформаційних систем. Впровадження системи моніторингу безпеки та процедур інцидент-менеджменту забезпечує швидке виявлення та нейтралізацію загроз. Регулярні аудити безпеки та навчання персоналу створюють надійний захист від внутрішніх та зовнішніх загроз.

Розроблені рекомендації щодо експлуатації системи охоплюють всі аспекти повсякденного адміністрування, планового обслуговування та аварійних процедур. Автоматизація рутинних операцій через моніторинг та алерти дозволяє адміністраторам зосередитися на стратегічних завданнях оптимізації та розвитку системи. Процедури резервного копіювання та відновлення з регулярним тестуванням гарантують безперервність роботи системи навіть у разі серйозних технічних збоїв. Впровадження системи створило надійну основу для ефективного управління логістичним забезпеченням військових потреб з можливістю подальшого розширення функціональності та масштабування під зростаючі потреби організації.

ВИСНОВКИ

У ході виконання бакалаврської кваліфікаційної роботи було успішно розроблено та впроваджено інформаційну систему організації логістичного забезпечення військових потреб, яка відповідає сучасним вимогам до військових інформаційних систем та забезпечує ефективне управління матеріально-технічними ресурсами збройних сил.

Проведений аналіз теоретичних аспектів військової логістики підтвердив критичну важливість автоматизації логістичних процесів для сучасних збройних сил. Дослідження показало, що традиційні методи ручного планування та контролю постачань не здатні забезпечити необхідну швидкість реагування на зміни в оперативній обстановці та точність управління ресурсами. Огляд існуючих технологій виявив перспективність використання інтегрованих рішень на базі RFID, GPS, IoT, хмарних технологій та штучного інтелекту, проте високі витрати на впровадження та складність інтеграції залишаються основними викликами для військових організацій з обмеженими ресурсами.

Детальний аналіз предметної області дозволив обґрунтовано обрати технологічний стек для розробки системи. Вибір Python з фреймворком Django як основної платформи розробки забезпечив оптимальне поєднання швидкості розробки, вбудованих механізмів безпеки та функціональності, критично важливих для військових систем. PostgreSQL як основна СУБД продемонструвала необхідну надійність, безпеку та продуктивність для роботи з військовими даними. Гібридна методологія розробки, що поєднує Agile (Scrum) з елементами Waterfall, виявилася оптимальною для балансування гнучкості розробки з формальними вимогами документування військових проектів.

Практична реалізація системи підтвердила правильність архітектурних рішень та вибору технологій. Розроблена багаторівнева архітектура з чітким розділенням відповідальності між шарами забезпечила модульність, масштабованість та можливість незалежного розвитку компонентів. Створена логічна модель даних з нормалізацією до третьої нормальної форми ефективно

моделює всі аспекти логістичних процесів від управління користувачами до відстеження історії замовлень. Розроблені алгоритми обробки замовлень, планування постачань та системи сповіщень реалізують складну бізнес-логіку з використанням сучасних паттернів проектування.

Комплексне тестування системи продемонструвало її готовність до продуктивної експлуатації. Результати навантажувального тестування підтвердили здатність системи обробляти до 500 запитів на секунду з середнім часом відгуку 1.2 секунди для стандартних операцій, що відповідає вимогам військових логістичних систем. Тестування безпеки не виявило критичних вразливостей, а багаторівнева система захисту відповідає високим стандартам військових інформаційних систем. Покриття коду модульними тестами на рівні 85% для критичних компонентів гарантує високу якість та надійність системи.

Впровадження системи базується на контейнеризованій архітектурі з автоматизацією процесів розгортання, що забезпечує консистентність середовища та спрощує масштабування. Визначені вимоги до апаратного забезпечення є обґрунтованими та базуються на реальних потребах системи. Комплексні заходи забезпечення безпеки, включаючи шифрування даних, контроль доступу та захист від кібератак, створюють надійний захист військової інформації. Розроблені рекомендації щодо експлуатації охоплюють всі аспекти повсякденного адміністрування та забезпечують безперервність роботи системи.

Економічний ефект від впровадження системи включає скорочення логістичних витрат на 20-30% через оптимізацію планування та зменшення надлишкових запасів, підвищення точності обліку до 99,5% та зниження часу реагування на зміни потреб з декількох днів до кількох годин. Стратегічний ефект полягає у підвищенні боєготовності збройних сил через надійне та своєчасне забезпечення необхідними ресурсами.

Перспективи подальшого розвитку системи включають інтеграцію з технологіями штучного інтелекту для предиктивного планування потреб, впровадження блокчейн-технологій для забезпечення прозорості ланцюгів постачання, розширення мобільної функціональності для польового

використання та інтеграцію з міжнародними системами логістичного забезпечення в рамках участі України в міжнародних військових операціях.

Таким чином, поставлені в роботі завдання було повністю вирішено. Розроблена інформаційна система організації логістичного забезпечення військових потреб відповідає всім функціональним та нефункціональним вимогам, забезпечує необхідний рівень безпеки та продуктивності, і готова до впровадження в реальних умовах експлуатації. Система створює надійну технологічну основу для ефективного управління військовою логістикою та може служити базою для подальшого розвитку цифрових технологій у сфері оборони України.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бережна Н. Г. та ін. Проблеми транспортно-логістичного забезпечення в аграрній галузі. 2019.

2. Ус В. В. Організація логістичного контролінгу на підприємстві авіаційної галузі. 2024.

3. Нестеров О. В. Теоретичні аспекти застосування концепцій логістичного забезпечення у військовій сфері. 2022.

4. Саган В. Аналіз підготовки фахівців військових формувань і правоохоронних органів з питань логістичного забезпечення. 2021.

5. Сакун О. Сутність, особливості організації та реалізації логістичних операцій прикордонних регіонів в умовах війни. Економіка та суспільство. 2024. № 68.

6. Чуприна О., Колосок Е., Григоренко О. Гуманітарна логістика: особливості розвитку в сучасних реаліях. Економіка та суспільство. 2023. № 58.

7. Ковач М. О. Особливості управління логістичною компанією в сучасних умовах. 2024.

8. Лутак М. В. Логістична система оптимізації виробничих технологій. 2023.

9. Нестеренко О. Ю., Касюдик М. В., Овчиннікова В. О. Організаційно-економічний механізм забезпечення інтелектуалізації виробничих технологій : дис. Харків : Харківський національний автомобільно-дорожній університет, 2022.

10. Зельцер Р. Я. та ін. Цифрова трансформація процесів ресурсно-логістичного та організаційно-структурного забезпечення будівництва. Наука та інновації. 2019.

11. Данилюк І. та ін. Дослідження перспектив застосування квантових технологій у Збройних Силах України. Системи і технології зв'язку, інформатизації та кібербезпеки. 2025.

12. Василишин В. Я., Тарасенко О. В. Розвиток автономних роботизованих систем у сфері логістики та транспорту в умовах адаптації до викликів сучасної економіки України. Актуальні питання економічних наук. 2025. № 11.

13. Сєногонова Л. І., Товма Л. Ф. Функціональна композиція «Олімп» для продуктів спеціального дієтичного споживання. 2019.

14. Dachkovskyi V., Sampir O. Алгоритм функціонування системи логістичного забезпечення. Сучасні інформаційні технології у сфері безпеки та оборони. 2019. Т. 35, № 2. С. 87–92.

15. Наконечний О. В. Методика оцінювання ефективності функціонування системи логістичного забезпечення сил оборони держави. Наука і техніка Повітряних Сил Збройних Сил України. 2020. № 1 (38). С. 54–60.

16. Северин М. А., Зозульов О. В. Вибір програмного забезпечення для маркетингової інформаційної системи підприємства. 2019.

17. Задорожнюк Н. О. Сучасне програмне забезпечення для здійснення бізнес-аналізу. Економічний вісник Національного технічного університету України «Київський політехнічний інститут». 2021. № 19.

18. Грибовська Ю., Кононенко Ж. Застосування інформаційних систем в управлінні підприємством. Економіка та суспільство. 2023. № 47.

19. Кивлюк В. та ін. Удосконалення освітньої діяльності з метою розвитку системи логістики у Збройних Силах України. Social development & Security. 2019. Т. 9, № 6. С. 112–124.

20. Чернюк Є. Розробка інформаційної системи управління структурою логістичного підприємства. 2020.

21. Дулєбов І. І. Методи та засоби реалізації віртуального логістичного центру. 2023.

22. Любченко О. А. Кваліфікаційна робота на тему: Розробка та програмна реалізація системи для обліку складу логістичної компанії. 2024.

23. Плугіна Т. В., Кудирко О. М., Плугін Д. А. Вибір елементів модуля ідентифікації вантажу для ІТ-інфраструктури складського терміналу. 2024.

24. Моісеєнкова В. О., Вовк О. В. Генерація та впровадження оптимального рішення для методології розробки веб-сайту : дис. «Друкарня Мадрид», 2019.

25. Данченко О. Б., Занора В. О. Проектний менеджмент: управління ризиками та змінами в процесах прийняття управлінських рішень. 2019.

26. Кухарець Т. І. Особливості планування і формування вимог для проекту розробки краудсорсингової онлайн платформи. 2023.

27. Gorbova O. V., Bohutskyi D. V. Контекстний аналіз сайту. Science and Transport Progress. 2024. № 3 (107).

28. Овсянкін А. М. Вибір ефективних моделей системи управління вимогами в проектах. Bulletin of the National Technical University "KhPI". Series: Strategic management, portfolio, program and project management. 2019. № 1 (1326). С. 55–62.

29. Косіюк М. М., Більовський К. Е., Лисак В. М. Автоматизована інформаційна система управління закладом вищої освіти «Електронний університет». Інформаційні технології і засоби навчання. 2023. Т. 1, № 93. С. 96–116.

30. Бідюк П. І. та ін. Системи і методи підтримки прийняття рішень. 2022.

31. Сакун А. Облікова система як інформаційна база управління збутовою діяльністю. 2020.

32. Щербань В. Ю. та ін. Програмні модулі комп'ютерної програми реалізації алгоритму рекурсії для випадку змінного вхідного натягу. Вісник Хмельницького національного університету. Серія: Технічні науки. 2020.

33. Ратинський В. В. Інформаційні технології в бухгалтерському обліку. Перспективи та проблеми. 2021.

34. Москалець Т. О. Інформаційна система популяризації веб-інклюзивності. 2024.

35. Любимов М. О., Кулик В. А. Можливості, загрози та перспективи використання «хмарних» технологій у бухгалтерському обліку. 2019.

36. Бурячок В. Л. та ін. Технології забезпечення безпеки мережевої інфраструктури. 2019.

37. Філатова Г. П. Боргова безпека в системі забезпечення економічної безпеки держави : дис. Суми : Сумський державний університет, 2021.

ДОДАТКИ

Додаток А

```
from conection import *
from table import *


connection = create_connection(
    "postgres", "admin", "root", "127.0.0.1", "5432"
)

execute_query(connection, create_Users)
execute_query(connection, create_Orders)
execute_query(connection, create_CompletedOrdersLog)
execute_query(connection, create_Reports)


cur = connection.cursor()

create(cur)
```

Додаток Б

```
from typing import List, Optional


def main(args: Optional[List[str]] = None) -> int:
    """This is preserved for old console scripts that may still be referencing
    it.
```

```
    For additional details, see https://github.com/pypa/pip/issues/7498.
    """
    from pip._internal.utils.entrypoints import _wrapper

    return _wrapper(args)
```

Додаток В

```
"""Primary application entrypoint.
"""
import locale
import logging
import os
import sys
import warnings
from typing import List, Optional

from pip._internal.cli.autocompletion import autocomplete
from pip._internal.cli.main_parser import parse_command
from pip._internal.commands import create_command
from pip._internal.exceptions import PipError
from pip._internal.utils import deprecation

logger = logging.getLogger(__name__)


# Do not import and use main() directly! Using it directly is actively
# discouraged by pip's maintainers. The name, location and behavior of
# this function is subject to change, so calling it directly is not
```

```
# portable across different pip versions.

# In addition, running pip in-process is unsupported and unsafe. This is
# elaborated in detail at
# https://pip.pypa.io/en/stable/user_guide/#using-pip-from-your-program.
# That document also provides suggestions that should work for nearly
# all users that are considering importing and using main() directly.

# However, we know that certain users will still want to invoke pip
# in-process. If you understand and accept the implications of using pip
# in an unsupported manner, the best approach is to use runpy to avoid
# depending on the exact location of this entry point.

# The following example shows how to use runpy to invoke pip in that
# case:
#
#     sys.argv = ["pip", your, args, here]
#     runpy.run_module("pip", run_name="__main__")
#
# Note that this will exit the process after running, unlike a direct
# call to main. As it is not safe to do any processing after calling
# main, this should not be an issue in practice.


def main(args: Optional[List[str]] = None) -> int:
    if args is None:
        args = sys.argv[1:]

    # Suppress the pkg_resources deprecation warning
    # Note - we use a module of .*pkg_resources to cover
```

```
        # the normal case (pip._vendor.pkg_resources) and the
        # devendored case (a bare pkg_resources)
        warnings.filterwarnings(
            action="ignore", category=DeprecationWarning,
module=".*pkg_resources"
        )

        # Configure our deprecation warnings to be sent through loggers
        deprecation.install_warning_logger()

        autocomplete()

        try:
            cmd_name, cmd_args = parse_command(args)
        except PipError as exc:
            sys.stderr.write(f"ERROR: {exc}")
            sys.stderr.write(os.linesep)
            sys.exit(1)

        # Needed for locale.getpreferredencoding(False) to work
        # in pip._internal.utils.encoding.auto_decode
        try:
            locale.setlocale(locale.LC_ALL, "")
        except locale.Error as e:
            # setlocale can apparently crash if locale are uninitialized
            logger.debug("Ignoring error %s when setting locale", e)
        command = create_command(cmd_name, isolated=("--isolated" in
cmd_args))

        return command.main(cmd_args)
```