КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра технологій управління

Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма «Управління проектами»

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему:

"Дослідження моделей управління проєктом створення та впровадження системи збору, систематизації та аналізу даних з продажу фармацевтичної продукції."

Студента 2-го курсу групи УПз-21

Юрія ГРИНІВА
(прізвище, ім'я, по батькові)

(підпис студента)

Науковий керівник:

к.т.н., асистент
(науковий ступінь, вчене звання)

Тетяна ЛАТИШЕВА
(прізвище, ім'я, по батькові)

(дата) *(підпис)*

Попередній захист:

(Висновок: "До захисту в Екзаменаційній комісії")

Завідувач
кафедри технологій
управління

(підпис) Віктор МОРОЗОВ *(прізвище, ініціали)* *(дата)*

Київ – 2024

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра технологій управління

Освітній рівень Магістр

Спеціальність 122 Комп'ютерні науки

Освітньо-професійна програма «Управління проектами»

ЗАТВЕРДЖУЮ

Завідувач кафедри

професор Віктор МОРОЗОВ

_____

"___"_____20__року

ЗАВДАННЯ

НА ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

СТУДЕНТ: Юрій ГРИНІВ

ГРУПА: УПз-21

1. **Тема кваліфікаційної роботи.**

"Дослідження моделей управління проєктом створення та впровадження системи збору, систематизації та аналізу даних з продажу фармацевтичної продукції"

Протокол затвердження теми КРМ № 13 від 28 червня 2024 року.

2. **Строк подання студентом готової роботи - "13" грудня 2024 р.**

3. **Цільова установка та вихідні дані до роботи**

Розробити та впровадити інформаційну систему для автоматизованого збору, систематизації та аналізу даних про продажі фармацевтичної продукції з метою підвищення ефективності управлінських рішень.

Вихідні дані:

- Аналітичні методи: SWOT, PESTEL, регресійний аналіз, кластерний аналіз.
- Технологічні засоби: Java, Spring Boot, React, PostgreSQL, Docker.
- Системи для інтеграції: ERP та CRM системи фармацевтичних компаній.
- Тестові та реальні дані про продажі фармацевтичної продукції для перевірки функціональності системи.

4. Зміст роботи

Дослідження та обґрунтування доцільності проєкту, математична постановка задачі та моделювання, розробка інформаційного забезпечення проєкту, розробка програмного забезпечення, тестування системи, оформлення та висновки.

5. Перелік графічного матеріалу (слайдів):

Концептуальна модель ІС, концептуальна модель бази даних, даталогічна модель бази даних, фізична модель бази даних, дерево цілей, дерево проблем, WBS, OBS, календарний план, віхи графіки, зразки коду в HTML, CSS, Javascript, Java.

6. Календарний план виконання роботи:

№ п/п	Назва частини роботи	План виконання роботи
1.	Вибір теми кваліфікаційної роботи магістра	1.09.24 - 4.09.24

2	Вивчення літературних джерел та аналіз методів оцінки впливу на ІТ-проєкти. Пошук та аналіз наукових статей, книг та інтернет-джерел з теми дослідження	5.09.24 - 10.09.24
3	Формулювання проблемної області та постановка задач дослідження. Визначення ключових проблем і формування завдань дипломної роботи	11.09.24 - 15.09.24
4	Написання Розділу 1: Дослідження та обґрунтування доцільності проєкту. Аналіз методів оцінки впливів оточення, SWOT та PESTEL-аналізи, огляд джерел	16.09.24 - 27.09.24
5	Написання Розділу 2: Математична постановка задачі та моделювання. Розробка концептуальної моделі, формалізація математичних моделей (лінійна регресія, кластерний аналіз)	28.09.24 - 12.10.24
6	Написання Розділу 3: Розробка інформаційного забезпечення проєкту. Проєктування концептуальної та логічної моделі бази даних	13.10.24 - 24.10.24
7	Написання Розділу 4: Розробка програмного забезпечення. Реалізація модулів збору,	25.10.24 - 10.11.24

	аналізу, візуалізації даних, аутентифікації користувачів	
8	Тестування програмного забезпечення та моделювання розроблених моделей. Тестування на реальних та тестових даних, перевірка коректності математичних моделей	11.11.24 - 22.11.24
9	Оформлення та фіналізація кваліфікаційної роботи. Об'єднання всіх розділів, оформлення відповідно до вимог університету.	23.11.24 - 30.11.24
10	Підготовка до попереднього захисту	1.12.24 - 4.12.24
11	Остаточне оформлення кваліфікаційної роботи	5.12.24
12	Передача КРМ в електронному вигляді на кафедру на перевірку роботи на плагіат	6.12.24
13	Передача кваліфікаційної роботи на рецензію науковому керівнику	7.12.24
14	Передача кваліфікаційної роботи рецензенту для рецензування	9.12.24

	Відпрацювання виступу, підготовка презентації, коригування роботи за рекомендаціями наукового керівника.	11.12.24
15	Попередній захист роботи	13.12.23

Дата видачі завдання “___” _____ 20__р.

Керівник роботи Тетяна ЛАТИШЕВА
(посада, прізвище, ім’я, по батькові)

_____
(підпис)

Завдання прийняв до виконання студент групи УПз-21

Юрій ГРИНІВ
(прізвище, імя, по батькові)

_____
(підпис)

ЗМІСТ

АНОТАЦІЯ

кваліфікаційної роботи магістра на тему**:**

«Дослідження моделей управління проєктом створення та впровадження системи збору, систематизації та аналізу даних з продажу фармацевтичної продукції»

Студент: Гринів Юрій Романович
Науковий керівник: Латишева Т.В.
Рік захисту: 2024

Мета – провести комплексний аналіз та дослідження процесів для створення ефективного інформаційної системи створення, розробки, впровадження інформаційної системи для автоматизованого збору, систематизації та аналізу даних з продажу фармацевтичної продукції..

Ціль проєкту - розробка та впровадження інформаційної системи, спрямованої на покращення якості управлінських рішень, оптимізацію бізнес-процесів та забезпечення інтеграції з існуючими ERP і CRM системами для швидкого доступу до якісних даних.

Практична цінність роботи полягає у розробці інформаційної системи, яка дозволяє автоматизувати збір, систематизацію та аналіз даних про продажі фармацевтичної продукції. Впровадження системи забезпечить покращення ефективності прийняття управлінських рішень, оптимізацію бізнес-процесів, а також підвищення конкурентоспроможності фармацевтичних компаній. Результати можуть бути адаптовані для інших галузей, що мають потребу в обробці великих обсягів даних та автоматизації аналітики.

Наукова новизна роботи полягає у застосуванні сучасних моделей управління проєктами, формалізації математичних методів для обробки даних, впровадженні аналітичного інструментарію та автоматизації процесів у фармацевтичній галузі та включає побудову концептуальної моделі інформаційної системи, розробку нових математичних моделей, а також моделей бази даних.

Кваліфікаційна робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел та додатків.

У першому розділі проведено аналіз проблемної області, досліджено методи оцінки проєктів (SWOT, PESTEL), декомпозовано задачі за методом WBS.

У другому розділі розроблено концептуальну модель системи, математичні моделі аналізу даних, включаючи лінійну регресію та кластеризацію.

У третьому розділі спроєктовано інформаційне забезпечення, розроблено логічну та концептуальну моделі бази даних.

У четвертому розділі розроблено програмне забезпечення, описано структуру, алгоритми та функціональність системи. Проведено тестування та моделювання.

За результатами роботи зроблено висновки, які підкреслюють необхідність комплексного підходу до управління проєктом, включаючи аналіз впливу оточення, розробку математичних моделей та програмного забезпечення, а також детальне планування та управління ресурсами.

У висновках формулюються ключові результати та рекомендації для подальшого вдосконалення управління проєктом створення та впровадження інформаційної системи для збору, систематизації та аналізу даних з продажу фармацевтичної продукції. Отримані результати дозволяють підвищити ефективність прийняття управлінських рішень, оптимізувати бізнес-процеси та забезпечити інтеграцію із сучасними ERP і CRM системами, що сприяє покращенню конкурентоспроможності компаній.

Ключові слова: управління проєктами, фармацевтична продукція, автоматизація, моделювання, аналітика даних, інформаційна система, база даних, календарний план.

Робота містить 110 сторінок без додатків, 32 рисунків та 10 таблиць. Додатки складають 9 сторінок.

ВСТУП

*Актуальність теми***.** В сучасних умовах глобалізації та інтенсивного розвитку фармацевтичної індустрії ефективне управління даними про продажі фармацевтичної продукції стає надзвичайно важливим. Здатність швидко і точно збирати, систематизувати та аналізувати дані про продажі дозволяє компаніям оперативно реагувати на зміни ринку, покращувати логістичні процеси, оптимізувати запаси продукції та підвищувати рівень обслуговування клієнтів. Це, в свою чергу, сприяє підвищенню конкурентоспроможності та стабільності компанії.

Існуючі методи управління даними часто не відповідають сучасним вимогам, через що виникає потреба в створенні нових, більш ефективних систем. Інформаційна система для автоматизованого збору, систематизації та аналізу даних про продажі фармацевтичної продукції здатна забезпечити компаніям необхідні інструменти для прийняття обґрунтованих управлінських рішень.

Метою дослідження є створення та впровадження інформаційної системи для автоматизованого збору, систематизації та аналізу даних з продажу фармацевтичної продукції, яка сприятиме покращенню процесів прийняття рішень, забезпечує швидкий доступ до актуальної інформації та підвищить ефективність управління фармацевтичною компанією.

Завдання роботи:

1. Проведення аналізу існуючих методів оцінки впливів оточення на IT проєкти та визначити найефективніші з них для використання у даному проєкті.
2. Сформулювати проблемну область та визначити конкретні проблеми, які будуть вирішуватися в рамках проєкту.
3. Провести огляд літературних та інформаційних джерел з теми дослідження для визначення актуальних підходів та методів.
4. Сформулювати задачі дослідження та технічне завдання на розробку

інформаційної системи.

5. Розробити концептуальні моделі інформаційної системи, що включають всі необхідні компоненти та взаємозв'язки.
6. Формалізувати математичні моделі, які будуть використовуватися для аналізу даних.
7. Провести моделювання розроблених моделей та проаналізувати отримані результати.
8. Розробити концептуальну та логічну модель бази даних проєкту.
9. Описати структуру програмного забезпечення, яке буде використовуватися для реалізації системи.
10. Розробити алгоритми та інтерфейси програмного забезпечення, необхідні для забезпечення функціональності системи.
11. Провести тестування системи для забезпечення її коректної роботи та відповідності вимогам.
12. Підготувати рекомендації щодо подальшого розвитку системи та можливих напрямів її вдосконалення.

Об'єкт дослідження: це процеси розробки та впровадження інформаційної системи для збору, систематизації та аналізу даних про продажі фармацевтичної продукції, що забезпечує ефективне управління та обробку даних у фармацевтичній галузі.

*Предмет дослідження***:** конкретні аспекти та характеристики цих процесів управління проєктом, такі як методи оцінки впливу оточення ІТ проєкту, розробка концептуальної моделі інформаційної системи, модель бази даних, розробка математичних моделей, реалізація програмного забезпечення, ресурсне планування проєкту та інші аспекти, пов’язані з управлінням та реалізації проєкту.

Наукова новизна цього проекту полягає у впровадженні нових технологій та включає побудову концептуальної моделі інформаційної системи, розробку нових математичних моделей, а також моделей бази даних.

Основні аспекти новизни включають:

- Автоматизація збору даних з різних джерел у реальному часі, що знижує ручну роботу та помилки.
- Покращення якості даних через очищення, нормалізацію та усунення дублікатів.
- Розширені можливості аналітики з інтерактивними інструментами візуалізації та дашбордами.
- Інтеграція з хмарними ERP та CRM системами для єдиного інформаційного простору.
- Високий рівень безпеки завдяки передовим методам шифрування та аутентифікації.

Практична цінність дослідження полягає у створенні інноваційної інформаційної системи, яка дозволяє автоматизувати процеси збору та аналізу даних, що є критично важливими для ефективного функціонування фармацевтичних компаній. Система допоможе:

1. Підвищити точність та швидкість обробки даних.
2. Знизити витрати на логістику та управління запасами.
3. Забезпечити інтеграцію з існуючими ERP та CRM системами для створення єдиного інформаційного середовища.
4. Сприяти прийняттю обґрунтованих управлінських рішень за допомогою аналітичних інструментів. Отримані результати можуть бути використані на практиці як у фармацевтичній галузі, так і в інших секторах, де важливими є автоматизація та обробка великих обсягів даних.

Отримані результати мають велике практичне значення для сфери управління проєктами та інформаційних технологій, забезпечуючи підприємствам засоби для результативного ведення бізнесу в умовах глобалізації та високої конкуренції. Ці результати допоможуть підприємствам створити інноваційні та

конкурентоспроможні продукти, що відповідають потребам сучасного ринку. Впровадження створеної інформаційної системи сприятиме підвищенню ефективності управління бізнес-процесами, що, в свою чергу, забезпечить зростання прибутковості та конкурентності підприємств на ринку.

Дана дипломна робота відповідає вимогам наукових програм у сферах інформаційних технологій та управління проєктами, спрямованих на розвиток нових знань та впровадження практичних рішень у сфері комп'ютерних наук та бізнес-процесів. Дослідження є актуальним у контексті сучасного розвитку комп'ютерних наук та інформаційних технологій, а також відповідає потребам практичного бізнес-середовища у вдосконаленні управлінських процесів та впровадженні новітніх технологічних рішень

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ТА ОБҐРУНТУВАННЯ ДОЦІЛЬНОСТІ ПРОЄКТУ

1.1 Аналіз методів оцінки впливів оточення ІТ проєктів

Оцінка впливів оточення на ІТ проєкти є критично важливою для успішного впровадження будь-якого ІТ проекту. Це дозволяє виявити потенційні ризики, загрози та можливості, які можуть вплинути на хід проекту. Існує декілька методів оцінки впливів, які можуть бути застосовані для аналізу оточення ІТ проектів:

1. *SWOT-аналіз (Strengths, Weaknesses, Opportunities, Threats):*

   Опис методу: SWOT-аналіз ідентифікує сильні та слабкі сторони, а також можливості та загрози для проєкту.

   - Сильні сторони (Strengths): Визначаються внутрішні позитивні фактори, які сприяють успіху проекту [6].
   - Слабкі сторони (Weaknesses): Виявляються внутрішні негативні фактори, які можуть стати перешкодою для реалізації проекту.
   - Можливості (Opportunities): Аналізуються зовнішні позитивні фактори, які можуть сприяти успіху проекту.
   - Загрози (Threats): Ідентифікуються зовнішні негативні фактори, які можуть негативно вплинути на проект [6].

   Критичний аналіз:

   - Переваги: Простота реалізації, можливість залучення різних фахівців.
   - Недоліки: Суб'єктивність оцінок, відсутність кількісної оцінки впливу.

2. *PESTEL-аналіз (Political, Economic, Social, Technological, Environmental, Legal):*

   Опис методу: Аналіз політичних, економічних, соціальних, технологічних, екологічних та правових факторів.

   - Політичні фактори (Political): Включають політичну стабільність, урядові політики та регуляції, які можуть впливати на проект.

- Економічні фактори (Economic): Аналіз економічних умов, таких як інфляція, обмінні курси, рівень зайнятості, які можуть вплинути на бюджет проекту [7].
- Соціальні фактори (Social): Вивчення соціальних трендів, культурних аспектів та демографічних змін, які можуть вплинути на користувачів системи.
- Технологічні фактори (Technological): Оцінка технологічних змін, інновацій та розвитку, які можуть вплинути на вибір технологій для проекту.
- Екологічні фактори (Environmental): Аналіз екологічних умов, які можуть вплинути на проект, такі як екологічні закони та регуляції.
- Юридичні фактори (Legal): Включають правові аспекти, такі як регуляції, стандарти, ліцензування та інші юридичні вимоги, які повинні бути враховані під час реалізації проекту [7].

Критичний аналіз:

- Переваги: Розглядає широкий спектр зовнішніх факторів.
- Недоліки: Висока складність у зборі релевантної інформації для всіх аспектів.

3. *Аналіз зацікавлених сторін (Stakeholder Analysis):*
    - Визначення основних зацікавлених сторін проекту, таких як спонсори, користувачі, регулятори, та оцінка їхніх інтересів, впливу та рівня підтримки проекту.

    Критичний аналіз:

    - Переваги: Розглядає широкий спектр зовнішніх факторів.
    - Недоліки: Висока складність у зборі релевантної інформації для всіх аспектів.
4. *Аналіз п'яти сил Портера (Porter's Five Forces Analysis):*
    - Конкурентне суперництво (Competitive Rivalry): Аналіз рівня

конкуренції серед існуючих гравців на ринку.

- Загроза нових гравців (Threat of New Entrants): Оцінка бар'єрів для входу нових конкурентів на ринок.
- Загроза замінників (Threat of Substitutes): Аналіз можливості появи альтернативних продуктів або послуг.
- Сила постачальників (Bargaining Power of Suppliers): Оцінка впливу постачальників на проект.
- Сила покупців (Bargaining Power of Buyers): Аналіз впливу клієнтів на ціни та якість продуктів або послуг [5].

Критичний аналіз:

- Переваги: Глибоке розуміння конкуренції.
- Недоліки: Менше фокусується на внутрішніх факторах.

5. *Аналіз життєвого циклу (Life Cycle Analysis - LCA):*
   - Цей метод дозволяє оцінити вплив проекту на оточення на різних етапах його життєвого циклу. Включає оцінку початкових інвестицій, операційних витрат, обслуговування, оновлень та утилізації.
   - Застосування LCA допомагає зрозуміти довгострокові екологічні, економічні та соціальні впливи проекту, що є особливо важливим для комплексних IT систем.

   Критичний аналіз:

   - Переваги: Дозволяє врахувати довгострокові наслідки.
   - Недоліки: Вимагає великої кількості початкових даних.
6. *Критичний аналіз успішності проектів (Critical Success Factors - CSF):*
   - Метод визначення ключових факторів успішності проекту, які мають бути враховані для забезпечення його успішного виконання.
   - Включає ідентифікацію та аналіз факторів, таких як якість управління проектом, підтримка керівництва, адекватність ресурсів, ефективність комунікацій та інші критичні елементи.

Застосування цих методів дозволяє забезпечити комплексний підхід до оцінки впливів оточення на IT проекти, що сприяє прийняттю обґрунтованих рішень, підвищенню ефективності управління проектами та зниженню ризиків. Вибір конкретного методу або комбінації методів залежить від специфіки проекту, його масштабів, цілей та ресурсів, які є у розпорядженні проектної команди.

1.2 Доцільність обраного проєкту

Обрана тема дослідження, а саме *створення та впровадження системи збору, систематизації та аналізу даних з продажу фармацевтичної продукції*, є вкрай актуальною у контексті сучасних потреб бізнесу та технологічного розвитку. Доцільність впровадження даного проєкту обґрунтовується наступними ключовими факторами:

1. *Ринковий попит*.

   Умови глобалізації та зростаюча конкуренція на фармацевтичному ринку вимагають впровадження нових інструментів для аналізу даних та оптимізації процесів. За даними ринкових досліджень, ефективне управління даними про продажі дозволяє компаніям підвищити швидкість реагування на зміни попиту, поліпшити планування запасів та зменшити витрати на логістику. Зокрема, фармацевтичні компанії активно потребують рішень, які здатні забезпечити автоматизовану аналітику для підвищення ефективності бізнес-процесів [16].

2. *Конкурентоспроможність*.

   Запровадження інформаційної системи надає значні переваги перед конкурентами завдяки інтеграції з ERP та CRM-системами, покращенню точності даних, прогнозуванню продажів та аналізу поведінки споживачів. В умовах швидкого розвитку фармацевтичної галузі автоматизовані системи сприяють оптимізації витрат, покращенню управління ресурсами та забезпеченню персоналізованого підходу до клієнтів [34].

3. *Технічна реалізованість.*

   Використання сучасних технологій, таких як *Spring Boot, React, PostgreSQL та Docker*, дозволяє забезпечити надійність, масштабованість та продуктивність розробленої системи. Крім того, інтеграція з існуючими системами підприємств є технічно можливою завдяки застосуванню API та стандартних протоколів обміну даними. Технологічна інфраструктура також передбачає можливість роботи системи у хмарному середовищі, що підвищує доступність та знижує витрати на підтримку [35].

4. *Ефективність та продуктивність.*

   Автоматизація процесів збору та аналізу даних забезпечить зменшення часу на обробку інформації на 40-60% у порівнянні з ручним введенням даних. Це також дозволить мінімізувати помилки, пов’язані з людським фактором, а точність прогнозів попиту підвищиться на 20-30% завдяки використанню алгоритмів математичного моделювання, таких як лінійна регресія та кластерний аналіз [43].

5. *Аналіз ринків та стратегічне планування.*

   Система дозволяє отримувати вичерпну аналітику ринку на основі великих обсягів даних (Big Data), що забезпечує можливість:

   - Визначення тенденцій попиту на фармацевтичну продукцію [36];
   - Формування персоналізованих стратегій просування продукції для різних регіонів та клієнтських груп;
   - Підвищення ефективності маркетингових кампаній та прогнозування обсягів продажів на основі історичних даних [36].

6. *Покращення управлінських рішень.*

   Сучасні інструменти аналітики та інтерактивна візуалізація даних дозволять керівникам компаній приймати обґрунтовані рішення, знижуючи ризики та забезпечуючи оперативне реагування на зміни ринку. Система буде спрямована на підвищення прозорості бізнес-процесів та надання

можливості глибокого аналізу ключових показників ефективності (KPI) [32].

7. *Інформаційна безпека.*

   Проєкт передбачає високий рівень захисту даних завдяки використанню методів шифрування (AES, RSA) та контролю доступу користувачів. Це відповідає сучасним стандартам інформаційної безпеки та забезпечує збереження конфіденційності даних про продажі фармацевтичної продукції.

8. *Соціально-економічна значимість.*

   Впровадження проєкту сприяє підвищенню ефективності роботи фармацевтичних компаній, що, у свою чергу, покращує доступ споживачів до необхідних лікарських засобів. Оптимізовані процеси дозволять знизити витрати на логістику, мінімізувати дефіцит продукції та забезпечити швидке реагування на попит [36].

9. *Екологічна відповідальність.*

   Автоматизація та централізоване зберігання даних зменшують потребу у використанні паперової документації, що позитивно впливає на довкілля. Це відповідає глобальним тенденціям до сталого розвитку та цифровізації.

10. *Перспективи масштабування*.

    Система може бути легко адаптована для інших галузей, таких як медична техніка, косметологія чи ветеринарна фармацевтика. Крім того, її функціонал можна розширити для підтримки нових джерел даних, використання хмарних сервісів та впровадження алгоритмів штучного інтелекту для глибшого аналізу даних [28].

1.3 Формулювання проблемної області

Проблемна область даного дослідження включає кілька ключових аспектів, що потребують вирішення для забезпечення ефективного управління даними про продажі фармацевтичної продукції. Основні проблеми, які будуть розглядатися в рамках роботи, наступні:

1. *Фрагментація даних:*
    - У багатьох фармацевтичних компаніях дані про продажі зберігаються у різних форматах та в різних системах, що ускладнює їх аналіз та інтеграцію [23].
    - Відсутність централізованої бази даних призводить до втрат інформації та помилок при збиранні даних [34].
    - Вплив: Зростання витрат на обробку даних, сповільнення прийняття рішень.
2. *Низька якість даних:*
    - Дані про продажі часто містять помилки, дублікати та неповні записи, що ускладнює їх використання для аналітики[[34].
    - Відсутність стандартів та протоколів збирання даних призводить до різнорідності даних та їх невідповідності [36].
    - Вплив: Зниження точності прогнозів, хибні управлінські рішення.
3. *Недостатня автоматизація процесів:*
    - Багато процесів збору та обробки даних виконуються вручну, що є трудомістким та збільшує ймовірність помилок [25].
    - Відсутність автоматизованих інструментів для систематизації та аналізу даних затримує процес прийняття рішень [20].
    - Вплив: Трудомісткість процесів, зниження продуктивності.
4. *Складність доступу до актуальної інформації:*
    - Керівники та аналітики часто не мають своєчасного доступу до актуальних даних про продажі, що ускладнює прийняття обґрунтованих управлінських рішень [39].
    - Відсутність інтерактивних аналітичних інструментів обмежує можливості глибокого аналізу даних.
5. *Інформаційна безпека:*
    - Зберігання та обробка конфіденційних даних про продажі

фармацевтичної продукції потребує високого рівня захисту.

- Недостатній рівень інформаційної безпеки може призвести до витоку даних та фінансових втрат.

6. *Складність інтеграції з іншими системами:*
    - Інтеграція з існуючими інформаційними системами, такими як ERP (Enterprise Resource Planning) та CRM (Customer Relationship Management), є складним завданням.
    - Відсутність стандартних інтерфейсів та протоколів для інтеграції ускладнює обмін даними між системами.
7. *Аналіз та прогнозування продажів:*
    - Існуючі методи аналізу даних не завжди дозволяють точно прогнозувати продажі та виявляти тенденції ринку [21].
    - Відсутність інструментів для прогнозування ускладнює планування виробництва та управління запасами.
8. *Недостатня аналітична підтримка:*
    - Багато компаній не мають доступу до сучасних інструментів аналітики, що обмежує їхні можливості щодо глибокого аналізу даних та виявлення прихованих патернів [21].
    - Відсутність можливостей для проведення комплексного аналізу великих обсягів даних (Big Data) обмежує стратегічне планування та прогнозування [21].
9. *Проблеми з обробкою великих обсягів даних:*
    - У фармацевтичній галузі обсяги даних можуть бути дуже великими, і традиційні методи обробки даних часто не справляються з такими обсягами [25].
    - Потреба у високопродуктивних обчислювальних ресурсах та спеціалізованих алгоритмах для обробки та аналізу великих даних (Big Data).

Розв'язання цих проблем вимагатиме розробки та впровадження комплексної інформаційної системи, яка забезпечуватиме ефективне управління даними про продажі фармацевтичної продукції. Така система повинна інтегрувати сучасні технології для збору, систематизації та аналізу даних, забезпечувати високу якість даних, підтримувати мобільність та масштабованість, а також відповідати усім регуляторним вимогам. Це дозволить фармацевтичним компаніям підвищити свою ефективність, конкурентоспроможність та оперативність у прийнятті управлінських рішень [20].

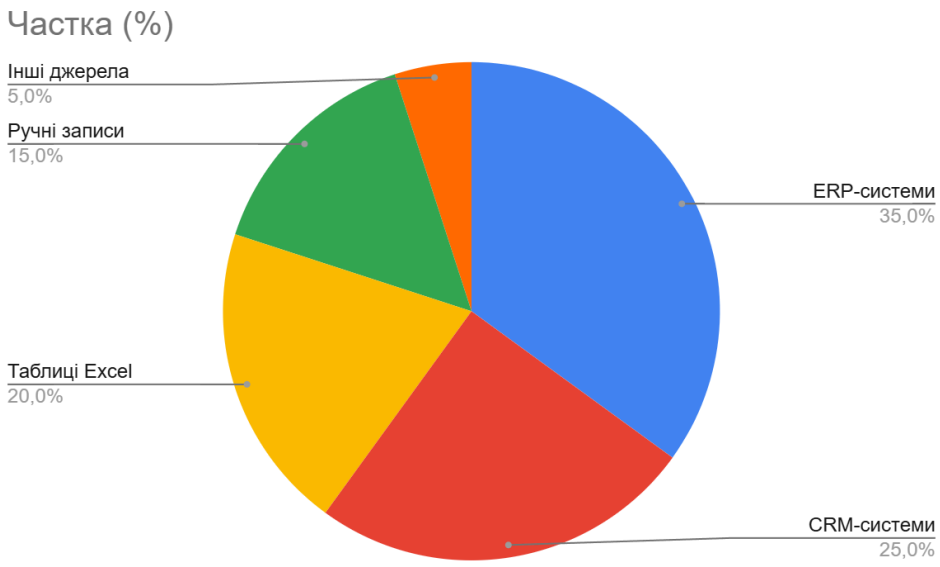


Рис. 1.1 Розподіл джерел даних

На діаграмі відображено розподіл джерел даних, які використовуються у фармацевтичних компаніях для збору інформації про продажі. Найбільшу частку займають ERP-системи (35%), що підкреслює важливість автоматизації і централізованого управління даними. CRM-системи займають 25%, демонструючи зосередженість на аналізі клієнтської бази. Значна частка (20%) даних все ще зберігається у таблицях Excel, що свідчить про потребу у впровадженні сучасних технологій. Ручні записи складають 15%, а інші джерела — лише 5%, що вказує на необхідність стандартизації та автоматизації процесів збору даних.

Вирішення зазначених проблем через впровадження комплексної інформаційної системи дозволить:

- Знизити витрати на обробку даних.
- Підвищити точність прогнозування.
- Поліпшити інтеграцію з іншими системами.
- Оптимізувати управління фармацевтичною компанією.

1.4. Аналіз літературних та інформаційних джерел – Проведення огляду літератури та інформаційних джерел з теми дослідження.

Інновації у використанні великих даних у фармацевтичній галузі**.** Фармацевтичні компанії активно впроваджують великі дані для покращення ефективності лікування та досліджень. Наприклад, *F. Hoffmann-La Roche* використовує аналітику великих даних для створення персоналізованих стратегій лікування [1]. За допомогою геномних аналізів, які дозволяють передбачати фенотипи пухлин, компанія розробляє індивідуальні плани лікування, що підвищує ефективність терапії та знижує побічні ефекти. Це дозволяє лікарям точно визначати найбільш ефективні ліки та дози для кожного пацієнта, враховуючи їхні генетичні особливості.

Thrive Bioscience застосовує машинне навчання для автоматизованого управління ресурсами в культурах клітин, що дозволяє покращити ефективність досліджень і скоротити кількість помилок [2]. Машинне навчання використовується для аналізу великої кількості даних про клітини, включаючи їх ріст, розвиток і реакцію на різні умови. Це допомагає дослідникам швидше і точніше визначати оптимальні умови для вирощування клітин і проведення експериментів.

Pharmaceutical Technology зазначає: "Використання машинного навчання дозволяє автоматизувати рутинні процеси та зменшити кількість помилок, що сприяє швидкому впровадженню нових ліків на ринок." [3].

Такі інновації підвищують точність і ефективність досліджень, а також скорочують час на розробку нових ліків, що є важливим для фармацевтичної індустрії. Застосування великих даних дозволяє проводити більш детальний аналіз ринку, виявляти нові можливості для розвитку та покращувати ефективність бізнес-процесів. Це включає аналіз продажів, прогнозування попиту та управління запасами, що сприяє оптимізації логістики та зменшенню витрат.

"Великі дані дозволяють фармацевтичним компаніям проводити більш детальний аналіз ринку, виявляти нові можливості для розвитку та покращувати ефективність бізнес-процесів." [8].

Машинне навчання не буде розглядатись в рамках нашого проєкту, проте в цьому літературному джерелі доволі непогано було підкреслено те, якими важливими є правильні і головне доцільні дані для створення персональних стратегій лікування, вивчення динаміки закупівель, тощо.

*Роль ІКТ у фармацевтичних ланцюгах постачання***.** Дослідження в BMC Health Services Research підкреслює важливість використання інформаційних та комунікаційних технологій (ІКТ) для підвищення ефективності ланцюгів постачання фармацевтичної продукції. Застосування ІКТ допомагає покращити координацію між учасниками ланцюга постачання, підвищити прозорість даних і зменшити ризики, пов'язані з управлінням запасами та постачанням ліків.

"ІКТ сприяють покращенню комунікації та співпраці між учасниками ланцюга постачання, що призводить до підвищення ефективності операцій і зменшення витрат." [4].

Впровадження ІКТ у фармацевтичні ланцюги постачання дозволяє:

- Підвищити прозорість і видимість даних про продажі, що сприяє оперативному прийняттю управлінських рішень. Це досягається завдяки інтеграції різних систем і забезпеченню доступу до актуальної інформації в режимі реального часу.
- Оптимізувати логістичні процеси та управління запасами, що зменшує

витрати і підвищує конкурентоспроможність компанії. Наприклад, автоматизовані системи управління запасами можуть прогнозувати попит на основі історичних даних і поточних тенденцій, що дозволяє зменшити запаси та уникнути дефіциту.

Дослідження також виявило, що ІКТ відіграють часткову посередницьку роль між практиками ланцюга постачання та операційною ефективністю. Однак, незважаючи на позитивний вплив ІКТ, існують виклики, пов'язані з видимістю даних та координацією між різними учасниками ланцюга постачання [4]. Наприклад, різні системи можуть мати несумісні формати даних або протоколи обміну інформацією, що ускладнює інтеграцію і призводить до втрат даних або помилок.

"*ІКТ відіграють часткову посередницьку роль між практиками ланцюга постачання та операційною ефективністю, покращуючи координацію та зменшуючи ймовірність помилок.*"

Отже, в контексті нашого проекту*:*

1. Оптимізація процесів збору та аналізу даних: Використовуючи інноваційні підходи, описані в роботах F. Hoffmann-La Roche та Thrive Bioscience, програмне забезпечення дозволить компаніям автоматизувати рутинні процеси збору даних. Це значно зменшить кількість помилок та підвищить точність і швидкість обробки даних.
2. Персоналізація маркетингових стратегій: Аналітика великих даних допоможе компаніям аналізувати поведінкові аспекти клієнтів, визначати найбільш ефективні стратегії продажів для різних сегментів ринку. Це підвищить ефективність маркетингових кампаній та дозволить компаніям краще адаптувати свої продукти до потреб ринку.
3. Покращення управління запасами: Інтеграція інформаційних та комунікаційних технологій (ІКТ), як показано в дослідженні BMC Health Services Research, дозволить компаніям покращити видимість

даних про продажі та оптимізувати управління запасами. Це допоможе уникнути дефіциту продукції, зменшити витрати на зберігання та підвищити оперативність реакції на зміни попиту.

4. Підвищення ефективності ланцюга постачання: Програмне забезпечення, яке інтегрується з існуючими ERP та CRM системами, сприятиме покращенню координації між різними відділами компанії та учасниками ланцюга постачання. Це забезпечить єдине джерело даних і сприятиме оперативному прийняттю рішень, що підвищить ефективність та конкурентоспроможність компанії.
5. Прогнозування попиту: Використання алгоритмів машинного навчання для прогнозування попиту на основі історичних даних про продажі та поточних тенденцій дозволить компаніям більш точно планувати свої запаси та виробничі потужності, що зменшить витрати та покращить обслуговування клієнтів [13].

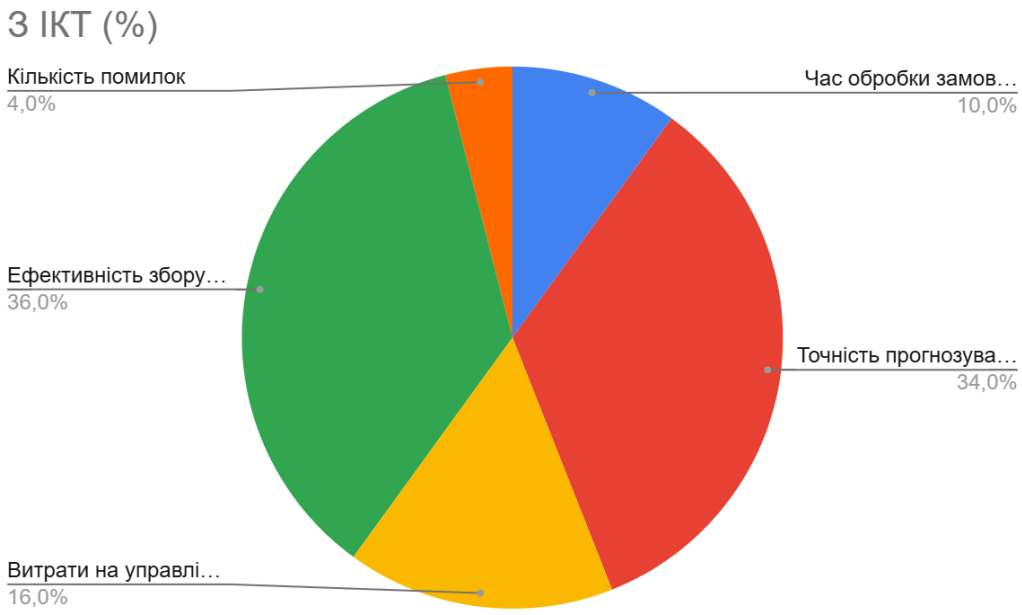


Рис. 1.2 Із ІКТ

Ця категорія демонструє покращення, досягнуті завдяки впровадженню ІКТ:

- Час обробки замовлення: Значно зменшується завдяки автоматизації бізнес-процесів і використанню інтегрованих систем.
- Точність прогнозування: Підвищується завдяки застосуванню аналітичних алгоритмів та великих даних.
- Витрати на управління запасами: Скорочуються завдяки використанню систем прогнозування попиту і автоматизованого контролю запасів.
- Ефективність збору даних: Максимізується через інтеграцію даних з різних джерел у єдину базу.
- Кількість помилок: Зменшується завдяки автоматичній перевірці і стандартизації даних.

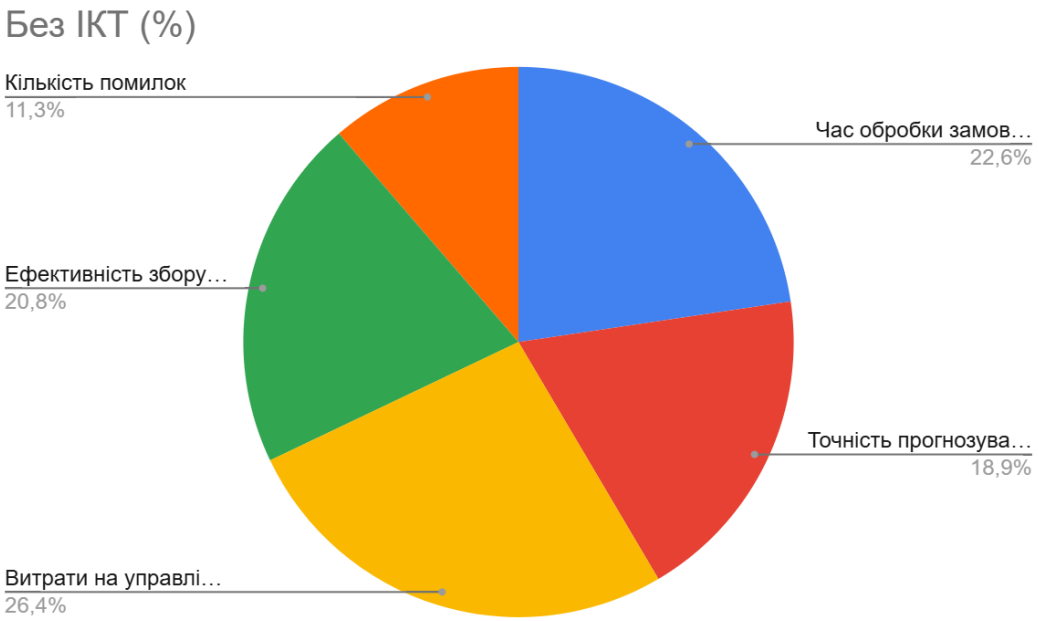


Рис. 1.3 Без ІКТ

Ця категорія відображає стан процесів у фармацевтичній компанії за відсутності автоматизації та використання сучасних технологій:

- Час обробки замовлення: Значний через ручне введення даних і неефективну координацію.

- Точність прогнозування: Обмежена, оскільки базується на історичних даних та ручному аналізі.
- Витрати на управління запасами: Вищі через відсутність автоматизованих інструментів оптимізації.
- Ефективність збору даних: Низька через фрагментацію інформації між різними джерелами.
- Кількість помилок: Більша через людський фактор і відсутність перевірки даних у реальному часі.

1.5 Постановка задачі дослідження та технічного завдання

Основною метою даного дослідження є розробка та впровадження інформаційної системи для автоматизованого збору, систематизації та аналізу даних про продажі фармацевтичної продукції. Задля досягнення цієї мети необхідно вирішити наступні задачі:

1. Провести аналіз існуючих методів оцінки впливів оточення на IT проєкти та визначити найефективніші з них для використання у даному проєкті:
    - Оцінка зовнішніх та внутрішніх факторів, що впливають на проект.
    - Використання методів SWOT, PESTEL та інших для аналізу.
2. Сформулювати проблемну область та визначити конкретні проблеми, які будуть вирішуватися в рамках проєкту:
    - Виявлення ключових проблем у зборі, зберіганні та аналізі даних про продажі.
    - Опис поточних викликів та їх впливу на фармацевтичні компанії.
3. Провести огляд літературних та інформаційних джерел з теми дослідження для визначення актуальних підходів та методів:
    - Вивчення сучасних тенденцій у використанні ІКТ у фармацевтичній галузі.
    - Аналіз впроваджених рішень та їх результативності.

4. Сформулювати задачі дослідження та технічне завдання на розробку інформаційної системи:
    - Визначення функціональних вимог до системи.
    - Опис технічних специфікацій та архітектури системи.

    Технічне завдання

    Цільова система:
    - Розробка інформаційної системи для збору, систематизації та аналізу даних про продажі фармацевтичної продукції.
2. Функціональні вимоги:
    - Автоматизований збір даних про продажі з різних джерел.
    - Централізоване зберігання даних у базі даних.
    - Інструменти для аналізу та візуалізації даних.
    - Підтримка інтеграції з існуючими ERP та CRM системами.
    - Забезпечення безпеки даних та контроль доступу.
3. Нефункціональні вимоги:
    - Висока надійність та доступність системи.
    - Масштабованість для підтримки зростання обсягів даних.
    - Підтримка мобільних платформ.
    - Відповідність регуляторним вимогам фармацевтичної галузі.
4. Архітектура системи:
    - База даних:
        - Вибір реляційної або NoSQL бази даних залежно від вимог до зберігання даних.
        - Моделювання концептуальної та логічної структури бази даних.
    - Збір даних:
        - Використання API для автоматизованого збору даних з джерел.
        - Модулі для ручного введення даних за потреби.
    - Аналіз даних:

        - Вбудовані аналітичні інструменти та алгоритми для обробки даних.
        - Візуалізація даних за допомогою графіків, діаграм та звітів.
    - Інтерфейси:
        - Розробка користувацьких інтерфейсів для доступу до функціональності системи.
        - Інтерфейси для інтеграції з іншими системами.
5. Безпека та контроль доступу:
    - Впровадження механізмів аутентифікації та авторизації користувачів.
    - Шифрування даних для захисту конфіденційної інформації.
    - Логування та моніторинг дій користувачів для забезпечення безпеки.
6. Тестування та верифікація:
    - Проведення юніт-тестування, інтеграційного тестування та системного тестування.
    - Верифікація системи на відповідність технічному завданню та вимогам користувачів.
7. Підготовка до впровадження:
    - Написання документації для користувачів та адміністраторів системи.
    - Навчання персоналу для роботи з новою системою.
    - План впровадження системи у виробниче середовище.

Очікувані результати:

- Розроблена та впроваджена інформаційна система для автоматизованого збору, систематизації та аналізу даних про продажі фармацевтичної продукції.
- Підвищена ефективність управління даними та прийняття управлінських рішень у фармацевтичних компаніях.
- Оптимізовані бізнес-процеси та зменшені витрати завдяки автоматизації та інтеграції даних.

Ці завдання та технічне завдання будуть детально розглянуті та реалізовані в наступних розділах дипломної роботи.

1.6 Декомпозиція задач проєкту за методом WBS.

Work Breakdown Structure (WBS) — це метод, що дозволяє декомпозувати проєкт на менші, керовані завдання. Його використання забезпечує структурований підхід до управління проєктом і дозволяє чітко визначити всі необхідні етапи реалізації.

Призначення WBS

Основна мета WBS полягає у створенні ієрархічної структури завдань для досягнення основної цілі проєкту. У контексті розробки інформаційної системи WBS забезпечує:

- Чітке визначення етапів реалізації.
- Полегшення управління ресурсами та часом.
- Спрощення моніторингу та контролю проєкту.

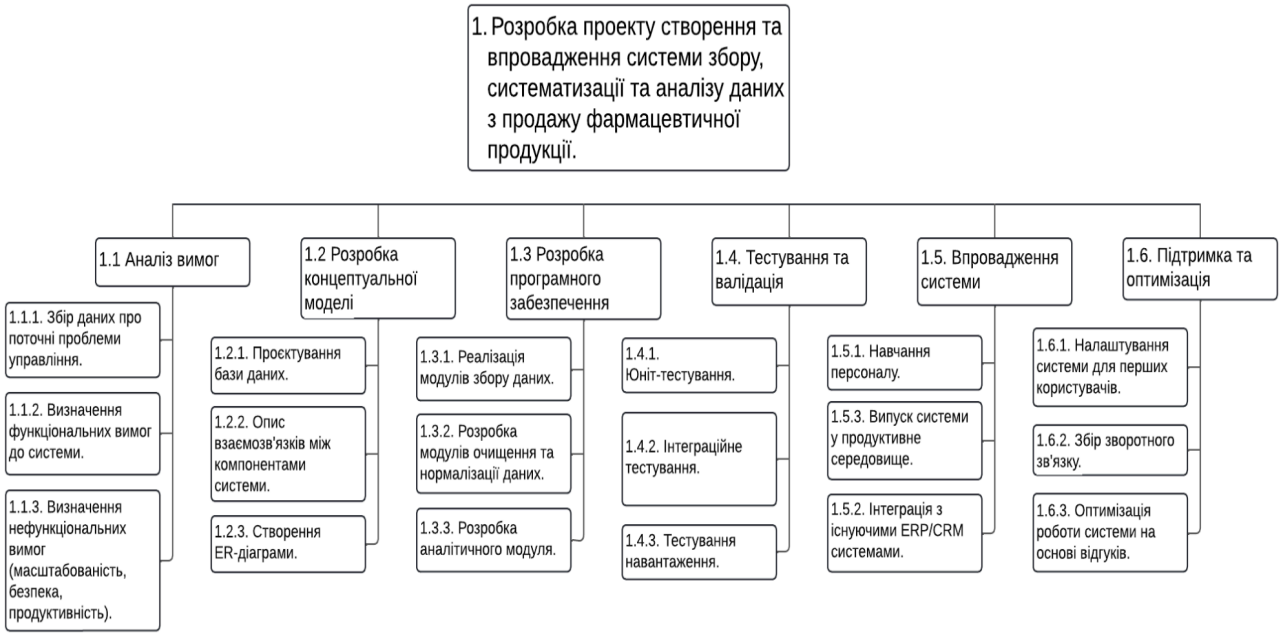


Рис. 1.6 WBS

1.1. Аналіз вимог

- 1.1.1. Збір даних про поточні проблеми управління.
- 1.1.2. Визначення функціональних вимог до системи.
- 1.1.3. Визначення нефункціональних вимог (масштабованість, безпека, продуктивність).

1.2. Розробка концептуальної моделі

- 1.2.1. Проєктування бази даних.
- 1.2.2. Опис взаємозв'язків між компонентами системи.
- 1.2.3. Створення ER-діаграми.

1.3. Розробка програмного забезпечення

- 1.3.1. Реалізація модулів збору даних.
- 1.3.2. Розробка модулів очищення та нормалізації даних.
- 1.3.3. Розробка аналітичного модуля.

1.4. Тестування та валідація

- 1.4.1. Юніт-тестування.
- 1.4.2. Інтеграційне тестування.
- 1.4.3. Тестування навантаження.

1.5. Впровадження системи

- 1.5.1. Навчання персоналу.
- 1.5.2. Інтеграція з існуючими ERP/CRM системами.
- 1.5.3. Випуск системи у продуктивне середовище.

1.6. Підтримка та оптимізація

- 1.6.1. Налаштування системи для перших користувачів.
- 1.6.2. Збір зворотного зв'язку.
- 1.6.3. Оптимізація роботи системи на основі відгуків.

РОЗДІЛ 2. ФОРМАЛІЗАЦІЇ ЗАДАЧІ ДОСЛІДЖЕННЯ

2.1 Розробка концептуальної моделі інформаційної системи

Концептуальна модель — це абстрактний опис інформаційної системи, який визначає її основні компоненти та взаємозв'язки між ними. Для створення концептуальної моделі інформаційної системи збору, систематизації та аналізу даних про продажі фармацевтичної продукції потрібно визначити ключові компоненти системи, їх функціональність та взаємодію (Рис 2.1). Нижче наведено основні компоненти концептуальної моделі:

1. Джерела даних:
    - Аптеки
    - Дистриб'ютори
    - Виробники
    - Інші зовнішні джерела
2. Сервіс збору даних:
    - Автоматизовані скрипти для збору даних з різних джерел
    - API для інтеграції з зовнішніми системами
    - Інтерфейси для ручного введення даних
3. База даних:
    - Реляційна PostgreSQL база даних для зберігання даних про продажі
    - Структура бази даних (таблиці, зв'язки, індекси)
4. Сервіс очищення та нормалізації:
    - Алгоритми очищення та нормалізації даних
    - Інструменти для виявлення та усунення дублюючих записів
5. Аналітичний сервіс:
    - Статистичні методи аналізу даних
    - Інструменти для візуалізації даних (графіки, діаграми, звіти)
6. Інтерфейси користувачів:

- Веб-інтерфейси для доступу до функціональності системи
- Мобільні додатки для доступу до даних з мобільних пристроїв

7. Сервіс аутентифікації та авторизації:
    - Механізми аутентифікації та авторизації користувачів
    - Шифрування даних для захисту конфіденційної інформації
    - Логування та моніторинг дій користувачів
8. Інтерфейси інтеграції:
    - API для інтеграції з існуючими ERP та CRM системами на кожному презентаційному рівні сервісу
    - Інтерфейси для обміну даними з іншими системами
9. Сервіс шифрування:
    - Модулі шифрування: Алгоритми для шифрування даних, використовуючи стандарти AES, RSA, або інші відповідні методи шифрування.
    - Модулі дешифрування: Алгоритми для дешифрування даних.
    - Управління ключами: Логіка для генерації, зберігання та управління ключами шифрування.

Взаємодія компонентів

Дані про продажі надходять від різних джерел до модулів збору даних, де вони автоматично або вручну вводяться в систему. Зібрані дані зберігаються в базі даних, після чого проходять процес систематизації. Після цього дані аналізуються за допомогою відповідних модулів аналізу, результати аналізу візуалізуються та надаються користувачам через інтерфейси. Система забезпечує високий рівень безпеки та контролю доступу, а також інтеграцію з іншими системами через інтерфейси інтеграції.

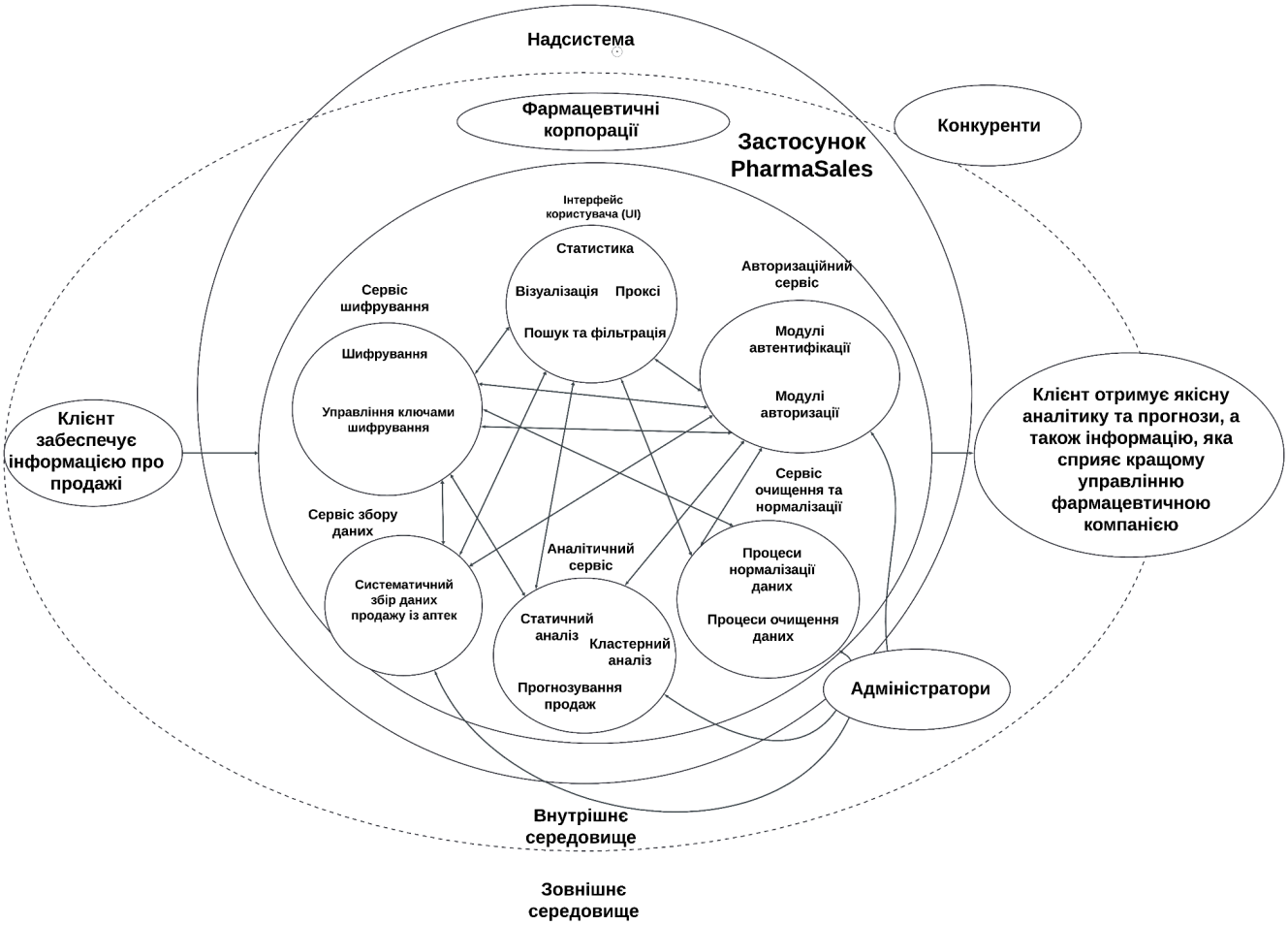


Рис. 2.1 Концептуальна модель інформаційної системи

Перспективи масштабування системи

У сучасних умовах динамічного розвитку бізнесу та технологій інформаційна система повинна забезпечувати можливість розширення та адаптації до зростаючих вимог [16]. Масштабованість є ключовим аспектом для довготривалої ефективності та стабільності системи [4].

1. Додавання нових модулів
    - Система може бути доповнена новими модулями без значних змін у її архітектурі. Зокрема:
        - Модуль прогнозування: для аналізу тенденцій продажів і попиту.
        - Модуль інтеграції з новими джерелами даних: наприклад, з

маркетинговими платформами чи системами управління логістикою.

- Модуль автоматизованого звітування: для створення аналітичних звітів у режимі реального часу.

2. Перехід на хмарну архітектуру
    - Перехід на використання хмарних платформ, таких як AWS, Microsoft Azure чи Google Cloud, відкриває нові можливості для масштабування:
        - Гнучке управління ресурсами: масштабування обчислювальних потужностей залежно від навантаження [22].
        - Збільшення сховища даних: можливість обробляти великі обсяги даних, включаючи історичні записи.
        - Висока доступність: забезпечення безперервної роботи системи через механізми автоматичного відновлення.
3. Підтримка багатокористувацького середовища
    - Впровадження багатокористувацьких інтерфейсів із розширеним керуванням правами доступу дозволить обслуговувати більшу кількість користувачів із різними ролями.
4. Розширення функціональності аналітичного модуля
    - Додавання можливостей для інтеграції алгоритмів машинного навчання для більш глибокого аналізу та прогнозування.
5. Міжнародна адаптація
    - Розширення системи для підтримки різних мов, валют і регуляторних стандартів дозволить використовувати її на міжнародному рівні.

Роль Use Cases у розробці системи

Use Cases, або сценарії використання, є одним із ключових інструментів для моделювання функціональності інформаційної системи [4]. Вони дозволяють описати основні взаємодії між користувачами (акторами) та системою,

деталізуючи, як саме система забезпечує виконання тих чи інших задач. У рамках розробки концептуальної моделі інформаційної системи фармацевтичної галузі Use Cases використовуються для:

- Визначення ключових функцій системи.
- Демонстрації взаємодії користувачів із системою.
- Покращення розуміння вимог до системи як з боку розробників, так і користувачів.

Нижче наведено приклади двох базових Use Cases, які відображають основні процеси, реалізовані в системі.

*Use Case 1***:** Збір даних про продажі з різних джерел

Ціль: Забезпечити автоматизований збір даних для подальшого аналізу.

- Актори: Менеджер з продажу, Система збору даних.
- Передумови:
    1. Система підключена до зовнішніх джерел даних (аптеки, дистриб'ютори).
    2. Налаштовані API для збору даних.
- Основний сценарій:
    1. Менеджер ініціює збір даних через інтерфейс.
    2. Система запускає автоматизовані скрипти для збору даних з джерел.
    3. Дані перевіряються на наявність дублювань та помилок.
    4. Оброблені дані записуються в централізовану базу.
- Результат: Успішно зібрані та систематизовані дані доступні для аналізу.

Use Case 2: Генерація звітів про продажі

Ціль: Забезпечити аналітичні звіти для стратегічного планування.

- Актори: Керівник відділу, Система аналізу.
- Передумови:

1. Дані вже зібрані та систематизовані.
2. Налаштований доступ користувача до системи.

- Основний сценарій:
  1. Керівник запитує створення звіту через веб-інтерфейс.
  2. Система отримує необхідні дані з бази.
  3. Алгоритми аналізу обчислюють ключові метрики (продажі за регіонами, популярність продуктів).
  4. Звіт генерується у вигляді графіків і таблиць.
  5. Користувач отримує доступ до звіту.
- Результат: Керівник отримує точний звіт для прийняття рішень

2.2 Розробка структурної моделі цілей та проблем ІТ-проєкту.

2.2.2 Дерево проблем ІТ-проєкту

Дерево проблем – це інструмент, що дозволяє візуалізувати основну проблему, її причини та наслідки. Воно будується у вигляді ієрархії, де головна проблема знаходиться у центрі, над нею розташовуються наслідки, а під нею – причини.

Контекст проєкту:

Основна проблема:

Неефективне управління даними про продажі фармацевтичної продукції.

Причини проблеми та їхній деталізований рівень:

1. *Фрагментація даних*:
   - Децентралізоване зберігання:
     - Дані розкидані по різних Excel-файлах.
     - Відсутність єдиного сховища.
   - Неефективний обмін даними:
     - Відсутність інтегрованих систем.
2. *Низька якість даних*:

- Наявність дублікатів:
    - Неконтрольований ручний ввід даних.
- Неповні записи:
    - Відсутність валідації під час збору даних.

3. *Відсутність автоматизації*:
    - Ручний збір:
        - Тривалий процес з високою вірогідністю помилок.
    - Застаріле ПЗ:
        - Відсутність підтримки сучасних технологій.
4. *Складність доступу до актуальних даних*:
    - Неефективні інтерфейси:
        - Відсутність швидкого пошуку та фільтрації.
    - Обмежена аналітика:
        - Відсутність автоматизованих звітів.
5. *Низький рівень безпеки*:
    - Відсутність шифрування:
        - Дані зберігаються у відкритому вигляді.
    - Слабкий контроль доступу:
        - Відсутність обмежень за ролями користувачів.

Наслідки проблем:

Зростання витрат на управління даними:

- Потреба у додаткових ресурсах для ручного введення та обробки.

Втрата цінних ресурсів через неефективну обробку:

- Неповноцінне використання зібраних даних.

Низька якість управлінських рішень:

- Неправильні прогнози щодо попиту та постачання.

Уповільнення прийняття стратегічних рішень:

- Затримка доступу до актуальної інформації.

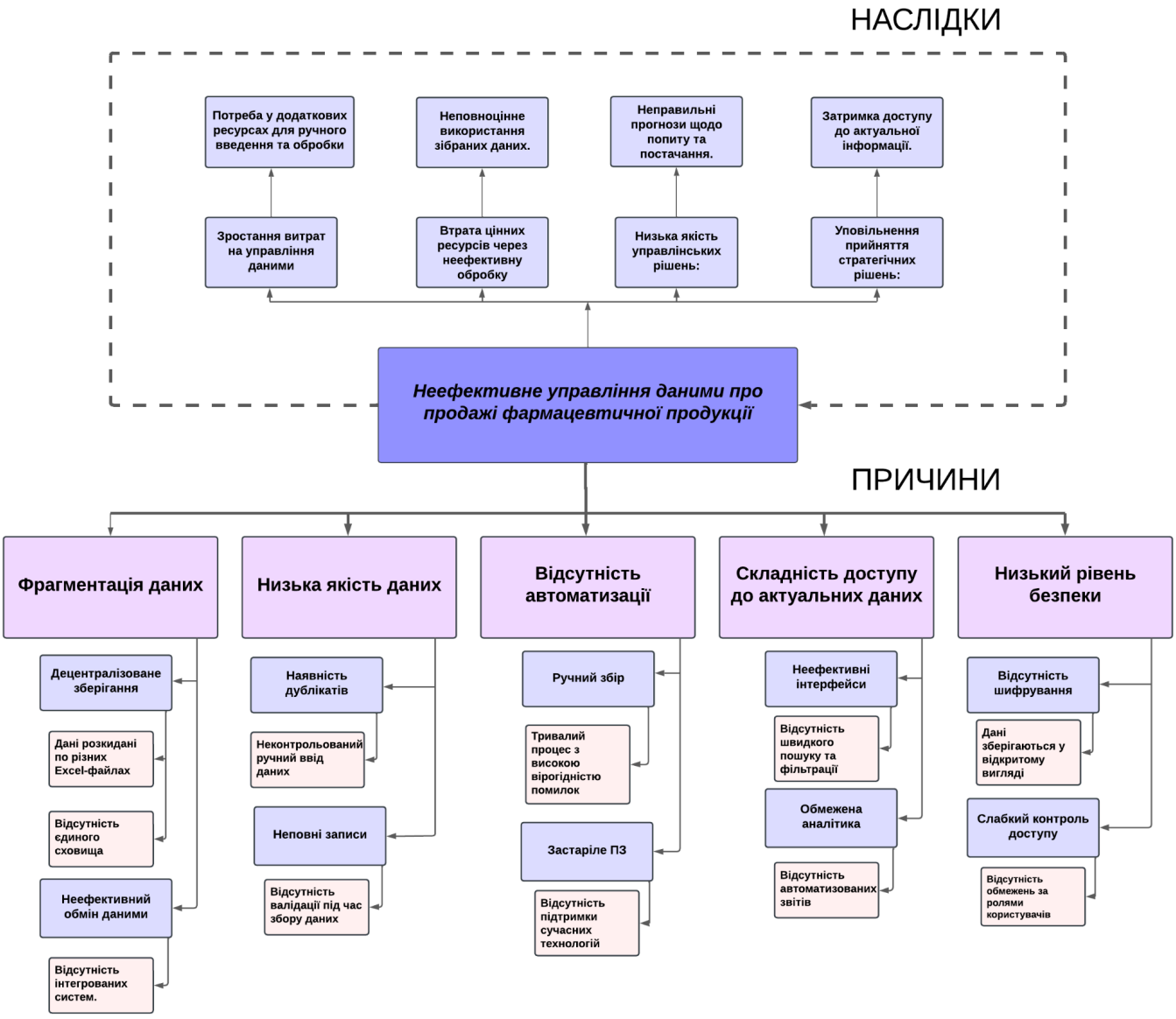


Рис. 2.3 Дерево проблем

2.2.2 Дерево цілей IT-проєкту

Дерево цілей – це графічне представлення основної мети проєкту, підцілей та конкретних завдань, які необхідно реалізувати для досягнення головної мети.

Контекст проєкту:

Головною метою є "*Розробка та впровадження системи збору, систематизації та аналізу даних про продажі фармацевтичної продукції*".

1. Головна мета: Розробка ефективної інформаційної системи.
2. Підцілі:

1. Управління проєктом

- Планування та розробка концепції проєкту
- Оцінка та управління ризиками
- Контроль виконання завдань
- Забезпечення якості виконання проєкту
- Завершення проєкту

2. Автоматизований збір даних

- Інтеграція систем
    - Розробка інтерфейсу для ручного додавання даних
    - Валідація даних у процесі вводу
- Ручний ввід
    - Розробка інтерфейсу для ручного додавання даних
    - Валідація даних у процесі вводу

3. Оптимізація бізнес-процесів

- Прогнозування попиту
    - Використання математичних моделей для прогнозу
    - Розробка інструментів для моніторингу відхилень прогнозів
- Аналіз продажів
    - Впровадження звітів за регіонами, періодами та категоріями товарів
    - Динамічний аналіз трендів продажів

4. Підвищення якості даних

- Очищення даних

- Впровадження алгоритмів для видалення дублікатів
- Нормалізація форматів даних

- Перевірка достовірності
  - Автоматичний аудит даних для виявлення помилок

5. Забезпечення аналітики

- Звіти та дашборди
  - Візуалізація ключових метрик (KPI)
  - Розробка інструментів для побудови аналітичних діаграм
- Глибинний аналіз даних
  - Впровадження інструментів для аналізу кореляцій та аномалій

6. Безпека даних

- Шифрування даних
  - Реалізація AES/RSA для збереження даних
- Аутентифікація користувачів
  - Впровадження ролей та рівнів доступу

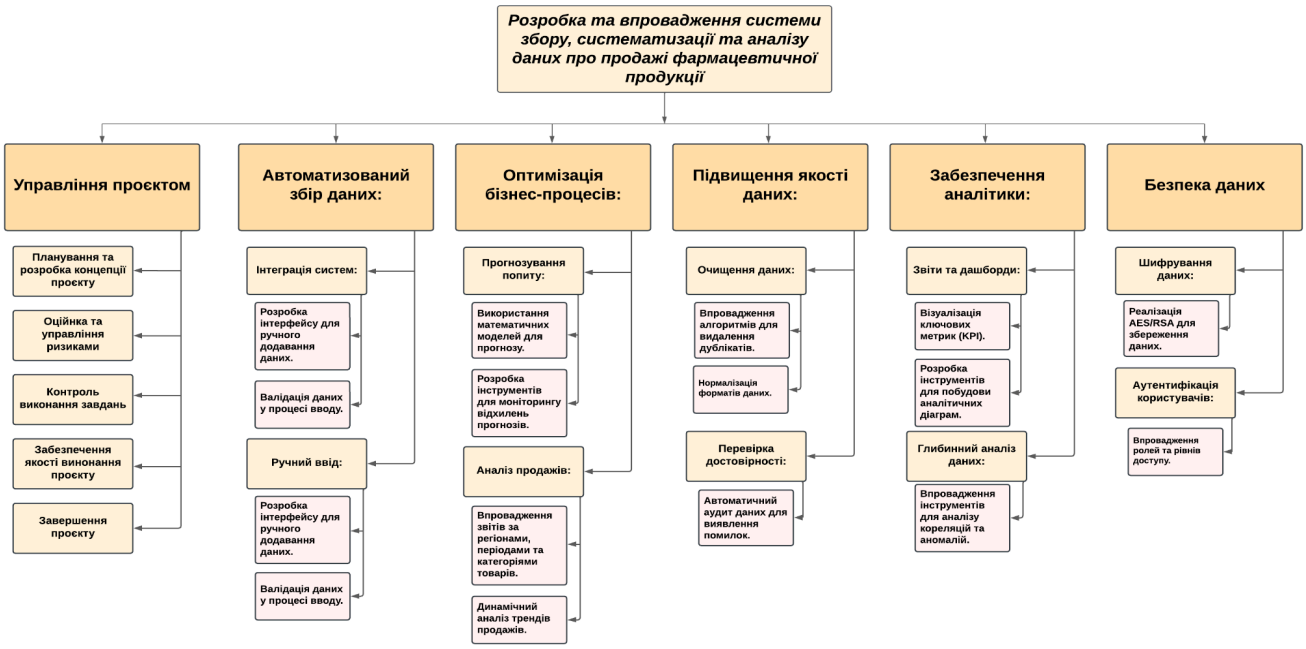


Рис. 2.2 Дерево цілей

2.3 Розробка паспорту проекту

Паспорт проєкту є невід'ємною складовою успішного управління проєктом. Він сприяє ефективному плануванню, реалізації та управлінню проєктом, допомагаючи чітко визначити мету і завдання, раціонально розподіляти ресурси, керувати ризиками, здійснювати контроль і моніторинг, а також забезпечувати ефективну комунікацію між усіма учасниками проєкту (Таблиця 1.1).

Таблиця 1.1

Паспорт проекту

Елемент	Характеристика
1	2
Назва проекту	PharmaSales
Тривалість проекту	12 місяців
Автор проекту	Гринів Юрій yuriihryniv@knu.ua +38(099) 311 65 16
Відповідальна особа керівник проекту	
Мета проекту	Розробка та впровадження інформаційної системи для автоматизованого збору, систематизації та аналізу даних про продажі фармацевтичної продукції.
Проблема на яку орієнтований проект	Сучасні методи управління даними про продажі фармацевтичної продукції часто не відповідають вимогам, що ускладнює прийняття рішень, оптимізацію логістичних процесів та підвищення рівня обслуговування клієнтів.
Цільова аудиторія	Фармацевтичні компанії, що прагнуть підвищити ефективність управління продажами, логістикою та оптимізувати бізнес-процеси.

Продовження табл 1.1

Цільовий ринок	Україна
Бюджет проекту	5 000 000 грн
Цілі проекту	● Запустити інформаційну систему до кінця року. ● Досягнути автоматизації збору даних з різних джерел. ● Забезпечити інтеграцію з існуючими ERP та CRM системами. ● Налаштувати аналітичні інструменти для ефективного аналізу даних. ● Провести тестування та впровадження системи у виробниче середовище.
Стейкхолдери	Команда розробників, проектний менеджер, бізнес-аналітики, маркетологи, фармацевтичні компанії, інвестори.
Фази життєвого циклу	● Підготовка проекту ● Проектування ● Виконання ● Випуск ● Підтримка ● Завершення
Функціональні вимоги	● Автоматизований збір даних ● Централізоване зберігання даних ● Аналітичні інструменти для візуалізації ● Інтеграція з ERP та CRM ● Забезпечення безпеки даних
Ризики та обмеження	Відсутність даних, технічні проблеми інтеграції, бюджетні обмеження, загроза витоку даних.

Закінчення табл. 1.1

Технічне завдання	Використання Java, Spring Boot, React, PostgreSQL; забезпечення CI/CD процесів та масштабованості.
Технології	Основні технології: - Мови програмування: Java, JavaScript - Фреймворки: Spring Boot, React - Бази даних: PostgreSQL - Інструменти контейнеризації: Docker, Kubernetes - Системи управління версіями: Git (GitLab) - Інструменти CI/CD: Jenkins, GitLab

2.4 Формалізація математичних моделей

Формалізація математичних моделей — це процес перетворення концептуальних моделей у формалізовані математичні описи, які можуть бути використані для аналізу даних та моделювання процесів. Для нашої інформаційної системи можна виділити наступні математичні моделі [15]:

1. *Модель очищення та нормалізації даних***:**
    - Використання алгоритмів для виявлення та усунення дублюючих записів
    - Нормалізація даних для забезпечення їх узгодженості та точності
2. *Статистичні моделі аналізу даних***:**
    - Регресійний аналіз для виявлення залежностей між різними змінними
    - Кластерний аналіз для групування даних
3. *Моделі прогнозування***:**
    - Моделі часових рядів для аналізу та прогнозування тенденцій
4. *Оптимізаційні моделі***:**

- Моделі лінійного програмування для оптимізації запасів продукції
- Алгоритми для оптимізації логістичних процесів

Розглянемо модель лінійної регресії для прогнозування продажів:

$$Y = \beta 0 + \beta 1X1 + \beta 2X2 + \ldots + \beta nXn + \epsilon \qquad (2.1)$$

де:

- Y — прогнозований обсяг продажів,
- $X1, X2, \ldots, Xn$— фактори, що впливають на продажі (наприклад, ціни, реклама, сезонність),
- $\beta 0, \beta 1, \beta 2, \ldots, \beta n$ — коефіцієнти моделі,
- ϵ — випадкова похибка.

Приклад використання моделі лінійної регресії для прогнозування продажів

Контекст:

Фармацевтична компанія хоче спрогнозувати обсяг продажів залежно від витрат на рекламу (X_1) та сезонності (X_2)

Вхідні дані:

- Витрати на рекламу: 15 000 грн.
- Сезонність: 1 (літній сезон), 0 (інші сезони).
- Коефіцієнти моделі:
  - $\beta 0 = 5000$,
  - $\beta 1 = 2.2$,
  - $\beta 3 = 1200$

Розрахунок:

Для літнього сезону: ($X_2 = 1$)

$$Y = 5000 + 2.2 \cdot 15000 + 1200 \cdot 1$$

$$Y = 5000 + 33000 + 1200 = 39200 \text{ грн.}$$

Для іншого сезону: ($X_2 = 0$)

$$Y = 5000 + 2.2 \cdot 15000 + 1200 \cdot 0$$

$$Y = 5000 + 33000 + 0 = 38000 \text{ грн.}$$

Результат:

- У літній сезон прогнозований обсяг продажів становить 39 200 грн..
- В інші сезони прогнозований обсяг продажів становить 38 000 грн.

У контексті моделі лінійної регресії, коефіцієнти є результатом навчання моделі на історичних даних. Ось як вони отримуються:

1. Збір історичних даних:
    - Наприклад, компанія збирає дані за минулі 12 місяців про:
        - X_1 - витрати на рекламу (у гривнях).
        - X_2 - сезонність (1 - літо, 0 - інші сезони).
        - Y - обсяги продажів (у гривнях).

Таблиця 2.2

Таблиця прикладу набору даних

Місяць	**Витрати на рекламу X_1**	**Сезонність X_2**	**Обсяг продажів Y**
Січень	10 000	0	28 000
Лютий	12 000	0	30 000
Липень	15 000	1	39 000
Грудень	20 000	0	45 000

2. Навчання моделі:

- Регресійний алгоритм (наприклад, методом найменших квадратів) знаходить такі значення

3. Розраховані коефіцієнти:
   - $\beta 0$ – базовий обсяг продажів, якщо всі інші фактори дорівнюють нулю.
   - $\beta 1$ – зміна обсягу продажів на кожну гривню витрат на рекламу.
   - $\beta 3$ – додатковий вплив літнього сезону на продажі.

Висновок:

Модель дозволяє оцінити, як різні фактори (витрати на рекламу, сезонність) впливають на обсяг продажів, що допомагає приймати обґрунтовані рішення для збільшення прибутку.

У контексті моделі лінійної регресії, коефіцієнти (β0,β1,β2) є результатом навчання моделі на історичних даних. Ось як вони отримуються:

Розглянемо модель кластерного аналізу для групування даних:

$$C_i = \frac{1}{|Si|} \sum_{x \epsilon |Si|} x \qquad (2.2)$$

де:

- C_i — новий центроїд кластера i,
- S_i — множина точок, що належать кластеру i,
- x — точка даних,
- $|Si|$ — кількість точок у кластері i.

Алгоритм k-середніх включає наступні кроки:

1. Вибір kkk початкових центроїдів випадковим чином.
2. Призначення кожної точки даних найближчому центроїду.

3. Оновлення центроїдів кластера на основі середнього значення точок, що належать до кожного кластера.
4. Повторення кроків 2-3 до збіжності.

Таблиця 2.3

Таблиця використання моделі кластерного аналізу для групування аптек

Аптека	Продажі (тис. грн)	Координати (x, y)
Аптека A	100	(10, 20)
Аптека C	80	(8, 18)
Аптека D	300	(50, 60)
Аптека E	320	(52, 62)

Алгоритм k-середніх:

1. Вибір початкових центроїдів:

- Випадково обрано два центроїди:
  - Кластер 1: (10, 20),
  - Кластер 2: (50, 60).

2. Призначення точок даних до найближчого центроїду:

- Відстань до центроїда розраховується за формулою Евклідової відстані

$$d = \sqrt{(x1 - x2)^2 + (y1 - y2)^2}$$

- Наприклад, для Аптеки A (10, 20) відстань:

- До центроїда 1:

$$d = \sqrt{(10 - 20)^2 + (20 - 20)^2} = 0$$

- До центроїда 2:

$$d = \sqrt{(10 - 50)^2 + (20 - 60)^2} = 56.57$$

- Аптека A приєднується до Кластера 1

3. Оновлення центроїдів:

- Для кластера 1 (Аптеки A, B, C):

$$C_1 = \left(\frac{10 + 12 + 8}{3}, \frac{20 + 22 + 18}{3}\right) = (10, 20)$$

- Для кластера 2 (Аптеки D, E):

$$C_1 = \left(\frac{50 + 52}{2}, \frac{60 + 62}{2}\right) = (51, 61)$$

4. Повторення кроків 2-3 до збіжності:

Результат:

Аптеки розділено на два кластери:

- Кластер 1: Аптеки A, B, C — низькі продажі, центральне розташування.
- Кластер 2: Аптеки D, E — високі продажі, віддалене розташування.

Висновок:

Кластерний аналіз допомагає оптимізувати логістику, формуючи групи аптек із подібними характеристиками.

2.5 Моделювання розроблених моделей

Моделювання — це процес перевірки та валідації розроблених моделей за допомогою комп'ютерного моделювання та аналізу даних [32]. Для нашої інформаційної системи моделювання включатиме наступні етапи:

1. *Підготовка даних:*

- Збір та підготовка тестових даних
- Очищення та нормалізація даних

2. *Моделювання процесів збору та систематизації даних:*
   - Тестування алгоритмів збору та очищення даних на тестових наборах
3. *Аналіз та валідація математичних моделей:*
   - Використання тестових даних для перевірки коректності математичних моделей
   - Порівняння прогнозованих результатів з фактичними даними
4. *Оцінка ефективності моделей:*
   - Вимірювання точності та надійності моделей
   - Оптимізація моделей для підвищення їх продуктивності

Приклад моделювання: Розглянемо процес прогнозування продажів за допомогою моделі лінійної регресії:

1. Збір даних: Зібрати історичні дані про продажі, ціни, витрати на рекламу тощо.
2. Підготовка даних: Очищення даних, видалення пропущених значень, нормалізація змінних.
3. Моделювання: Використання алгоритму лінійної регресії для побудови моделі на основі підготовлених даних.
4. Валідація: Тестування моделі на нових даних та порівняння прогнозованих значень з фактичними продажами.

Ці етапи забезпечать розробку ефективних та надійних математичних моделей для нашої інформаційної системи, що дозволить фармацевтичним компаніям оптимізувати управління даними про продажі та приймати обґрунтовані управлінські рішення.

2.5.1 Обґрунтування вибору моделей

Розробка інформаційної системи передбачає використання декількох

математичних моделей для вирішення різних задач. Основними критеріями вибору моделей були:

1. Простота реалізації
    - Вибрані моделі, такі як лінійна регресія та кластерний аналіз, є відносно простими в реалізації за допомогою сучасних мов програмування та бібліотек, таких як Python (scikit-learn) чи R. Це дозволяє швидко впровадити рішення без значних витрат на розробку.
2. Точність прогнозів
    - Модель лінійної регресії забезпечує високий рівень точності для завдань прогнозування, де залежність між змінними є лінійною. Це ідеально підходить для аналізу залежності продажів від витрат на рекламу чи сезонних факторів [36].
    - Кластерний аналіз дозволяє ефективно групувати дані на основі схожості, що корисно для сегментації клієнтів чи аптек.
3. Гнучкість і масштабованість
    - Моделі легко адаптуються до змінних даних. Наприклад, модель кластерного аналізу може бути налаштована для різної кількості кластерів, залежно від потреб бізнесу. Аналогічно, модель регресії може бути розширена для включення нових змінних, якщо це потрібно для підвищення точності прогнозів.
4. Практичне застосування
    - Ці моделі добре зарекомендували себе у багатьох галузях, зокрема у фармацевтиці, де вони застосовуються для прогнозування попиту, оптимізації логістики, аналізу ринкових сегментів і виявлення прихованих патернів у даних.
5. Ефективність в обчисленнях
    - Вибрані алгоритми мають низьку обчислювальну складність і добре працюють навіть із великими наборами даних, що важливо для

системи, яка обробляє великі обсяги інформації про продажі.

6. Інтеграція з іншими компонентами системи
    - Лінійна регресія та кластерний аналіз інтегруються з аналітичними модулями системи, забезпечуючи автоматизацію процесу прийняття рішень.

2.5.2 Аналіз обмежень розроблених моделей

1. Обмеженість вихідних даних
    - Проблема: Для ефективної роботи математичних моделей необхідна велика кількість якісних даних. Недостатній обсяг або погана якість даних можуть суттєво вплинути на точність прогнозів.
    - Приклад: Якщо дані про продажі зібрані лише за короткий період, модель лінійної регресії може дати неточні результати через недостатню репрезентативність історії продажів.
2. Припущення про лінійність
    - Проблема: Лінійна регресія передбачає лінійний зв'язок між змінними, що може не завжди відповідати реальності.
    - Приклад: У випадку, коли вплив витрат на рекламу на продажі має нелінійний характер (наприклад, ефект насичення), модель може некоректно оцінити внесок змінної.
3. Складність визначення оптимальної кількості кластерів
    - Проблема: У кластерному аналізі необхідно визначити оптимальне значення k
    - *k* (кількість кластерів). Невірний вибір може призвести до непрактичного розподілу даних.
    - Приклад: Якщо аптечна мережа має декілька регіонів із різними особливостями, а кластерів вибрано занадто мало, групи можуть бути

занадто широкими для корисної інтерпретації.

4. Вплив шуму в даних
    - Проблема: Наявність аномалій чи шуму в даних може знизити точність моделей і збільшити похибки прогнозів.
    - Приклад: Некоректні записи в базі даних, такі як відсутність інформації про продажі або аномально великі значення, можуть спотворювати результати кластеризації.
5. Проблеми масштабованості
    - Проблема: Для великих обсягів даних обрані моделі можуть працювати неефективно без оптимізації алгоритмів і технічних рішень.
    - Приклад: Модель кластерного аналізу може бути обчислювально дорогою при роботі з мільйонами записів, що вимагає оптимізації за рахунок паралельних обчислень чи використання спеціалізованих бібліотек.
6. Ризики перевищення прогнозу
    - Проблема: При використанні історичних даних для прогнозів є ризик, що модель не врахує майбутніх змін у ринку чи умовах.
    - Приклад: Зміна законодавства або несподівані тренди, такі як пандемія, можуть зробити модель неактуальною.

2.5.3 Приклади результатів моделювання

Графік на рис. 2.4 демонструє залежність між витратами на рекламу (червона вісь) та прогнозованим обсягом продажів (синя вісь), побудовану на основі моделі лінійної регресії. Він використовується для:

- Виявлення ефективного рівня інвестицій у рекламу.
- Визначення точки, де збільшення витрат на рекламу перестає бути економічно вигідним.

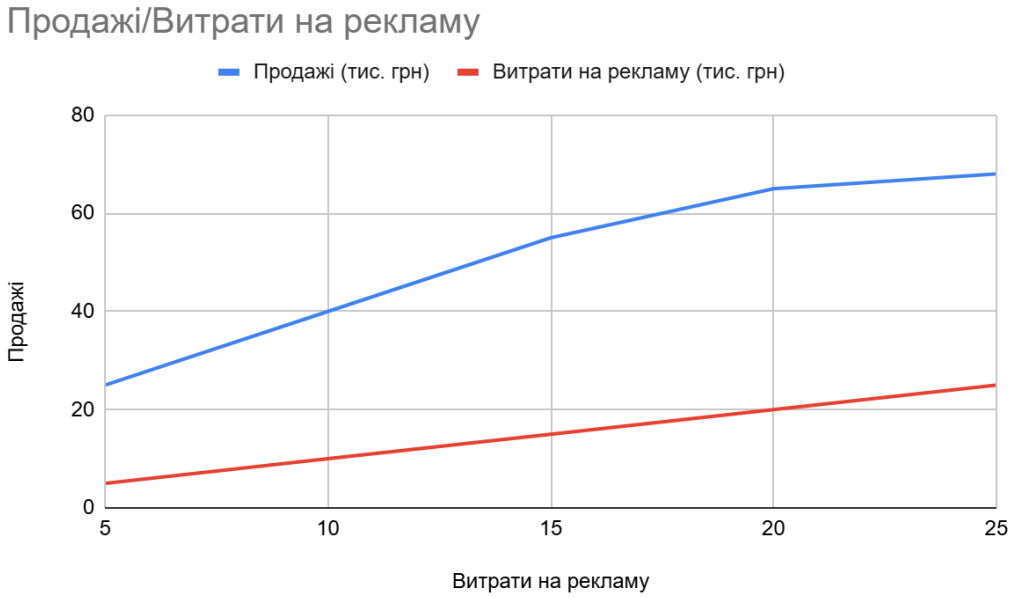


Рис. 2.4 Приклад результатів моделі лінійної регресії для прогнозування продажів

На графіку представлено залежність між витратами на рекламу та обсягом продажів. Аналіз цієї залежності дозволяє визначити оптимальний бюджет для рекламної кампанії, який максимізує прибуток компанії. Наприклад, при витратах понад 20 000 грн приріст продажів сповільнюється, що свідчить про досягнення межі ефективності реклами.

Наступна табличка ілюструє результати кластерного аналізу, яка групує аптеки за обсягами продажів та географічними координатами (таблиця 2.4). Спостерігається чітке розділення на два кластери: аптеки з меншими обсягами продажів (кластер 1) та аптеки з вищими показниками продажів (кластер 2).

Таблиця 2.4

Таблиця результатів кластерного аналізу по групуванню аптек за обсягами продажів та географічними координатами

Аптека	Продажі (тис. грн)	Координати X	Координати Y	Кластер
A	100	10	20	1
B	120	12	22	1
C	80	8	18	1
D	300	50	60	2
E	320	52	62	2

Результати кластерного аналізу відображено на таблиці 2.3. Аптеки згруповано у два основних кластери: аптеки з низькими продажами та центральним розташуванням, і аптеки з високими продажами та віддаленим розташуванням. Це дозволяє оптимізувати логістику, адаптувати маркетингові стратегії для різних регіонів та покращити управління запасами.

2.5.4 Приклади використання даних для ухвалення рішень

Результати, отримані за допомогою моделювання, активно застосовуються для підтримки управлінських рішень у фармацевтичній компанії. Нижче наведено приклади ключових застосувань:

1. Оптимізація маркетингових кампаній

   Результати кластерного аналізу дозволяють сегментувати аптеки за обсягами продажів і географічними характеристиками. Це допомагає:

   - Визначити пріоритетні аптеки з високим потенціалом продажів для

цільових маркетингових кампаній.

- Планувати рекламні бюджети ефективніше, спрямовуючи ресурси на кластери, де є найбільший потенціал для зростання [32].
- Виявити регіони з низькими продажами, які потребують додаткових стимулюючих заходів, таких як знижки чи спеціальні акції.

2. Планування виробництва та управління запасами

   Прогнозовані за допомогою моделі лінійної регресії обсяги продажів використовуються для:

   - Планування обсягів виробництва, щоб уникнути дефіциту або надлишкових запасів.
   - Оптимізації логістичних процесів, визначення необхідної кількості продукції для кожного регіону.
   - Прогнозування сезонних коливань попиту для заздалегідь підготовлених маркетингових і логістичних стратегій.

3. Фінансове планування

   Дані прогнозування допомагають оцінювати очікуваний дохід та рентабельність інвестицій у рекламу. Наприклад:

   - Якщо прогнозована рентабельність для певного регіону перевищує встановлений поріг, туди інвестуються додаткові ресурси [32].
   - Аналіз витрат на рекламу та їхнього впливу на продажі дозволяє скорегувати бюджети, уникаючи зайвих витрат.

4. Стратегічне планування

   - На основі результатів моделювання можна визначати довгострокові стратегії, такі як розширення мережі аптек у регіонах із високим потенціалом продажів [36].
   - Розробляти нові продукти або послуги, які відповідають потребам конкретних сегментів клієнтів, виявлених у процесі кластеризації.

2.6 Порівняння альтернативних моделей

При розробці інформаційної системи були розглянуті декілька альтернативних підходів до аналізу даних і прогнозування. Основними обраними моделями стали лінійна регресія та кластерний аналіз, оскільки вони забезпечують оптимальне співвідношення простоти реалізації та точності. Проте для порівняння розглянемо можливі альтернативи.

1. Нейронні мережі для прогнозування
   - Переваги:
     - Можливість моделювання складних нелінійних залежностей.
     - Висока точність прогнозів при достатній кількості даних.
   - Недоліки:
     - Складність реалізації та налаштування.
     - Велика потреба в обчислювальних ресурсах.
     - Чутливість до якості даних і ризик перенавчання.
2. Регресійний аналіз із нелінійними залежностями
   - Переваги:
     - Універсальність для моделювання різних типів залежностей.
     - Відносно проста реалізація в порівнянні з нейронними мережами.
   - Недоліки:
     - Складність налаштування моделі, особливо при великій кількості змінних.
     - Може вимагати більше часу на побудову та валідацію.
3. Метод дерев рішень для кластеризації та прогнозування
   - Переваги:
     - Інтуїтивно зрозумілі результати.
     - Добре працює з категоріальними даними.
   - Недоліки:
     - Може створювати занадто спрощені моделі.

РОЗДІЛ 3. РОЗРОБКА ІНФОРМАЦІЙНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ПРОЄКТУ

3.1 Концептуальна модель бази даних проєкту – Розробка концептуальної моделі бази даних.

Концептуальна модель бази даних є фундаментом для створення ефективної інформаційної системи збору, систематизації та аналізу даних про продажі фармацевтичної продукції. Вона відображає основні сутності, атрибути та взаємозв'язки між ними, що дозволяє забезпечити цілісність і послідовність даних. Нижче наведено опис основних сутностей, атрибутів та їх зв'язків на основі наданої ER-діаграми з урахуванням виправлень [12].

Основні сутності (Рис. 3.1):

1. Report (Звіт)
    - Атрибути:
        - ReportID (Ідентифікатор звіту)
        - ReportDate (Дата звіту)
        - SaleID (Ідентифікатор продажу)
        - UserID (Ідентифікатор юзера)
    - Зв'язки:
        - Один звіт формується на основі кількох продажів.
        - Один звіт формується для одного юзера.
2. User (Користувач)
    - Атрибути:
        - UserID (Ідентифікатор користувача)
        - UserName (Ім'я користувача)
        - UserEmail (Електронна пошта користувача)
        - UserRole (Роль користувача)
    - Зв'язки:

        - Один користувач може переглядати кілька звітів.

3. Sale (Продаж)
    - Атрибути:
        - SaleID (Ідентифікатор продажу)
        - SaleDate (Дата продажу)
        - ProductID (Ідентифікатор продукту)
        - Quantity (Кількість)
        - Price (Ціна)
    - Зв'язки:
        - Один продаж стосується одного продукту.
        - Один продаж може бути частиною кількох звітів.
4. Product (Продукт)
    - Атрибути:
        - ProductID (Ідентифікатор продукту)
        - ProductName (Назва продукту)
        - ManufacturerID (Ідентифікатор виробника)
        - PharmacyID (Ідентифікатор аптеки)
    - Зв'язки:
        - Один продукт має одного виробника.
        - Один продукт продається в одній аптеці.
        - Один продукт може бути проданий у кількох продажах.
5. Pharmacy (Аптека)
    - Атрибути:
        - PharmacyID (Ідентифікатор аптеки)
        - PharmacyName (Назва аптеки)
        - PharmacyLocation (Місцезнаходження аптеки)
        - ProductID (Ідентифікатор продукту)
    - Зв'язки:

- Одна аптека може мати кілька продуктів.

6. Manufacturer (Виробник)
    - Атрибути:
        - ManufacturerID (Ідентифікатор виробника)
        - ManufacturerName (Назва виробника)
        - ManufacturerCountry (Країна виробника)
        - ProductID (Ідентифікатор продукту)
    - Зв'язки:
        - Один виробник може виготовляти кілька продуктів.
7. Distributor (Дистриб'ютор)
    - Атрибути:
        - DistributorID (Ідентифікатор дистриб'ютора)
        - DistributorName (Назва дистриб'ютора)
        - DistributorCountry (Країна дистриб'ютора)
        - ManufacturerID (Ідентифікатор виробника)
        - PharmacyID (Ідентифікатор аптеки)
    - Зв'язки:
        - Один дистриб'ютор може постачати кілька продуктів.

Взаємозв'язки:

- Звіт формується автоматично на основі кількох продажів.
- Один користувач може переглядати кілька звітів.
- Продаж пов'язаний з одним продуктом.
- Продукт має одного виробника та продається в одній аптеці.
- Одна аптека може мати кілька продуктів.
- Один виробник може виготовляти кілька продуктів.
- Один дистриб'ютор може постачати кілька продуктів і пов'язаний з виробником та аптекою.

Ця концептуальна модель відображає основні компоненти інформаційної

системи та їх взаємозв'язки, що дозволяє забезпечити ефективне управління даними про продажі фармацевтичної продукції.

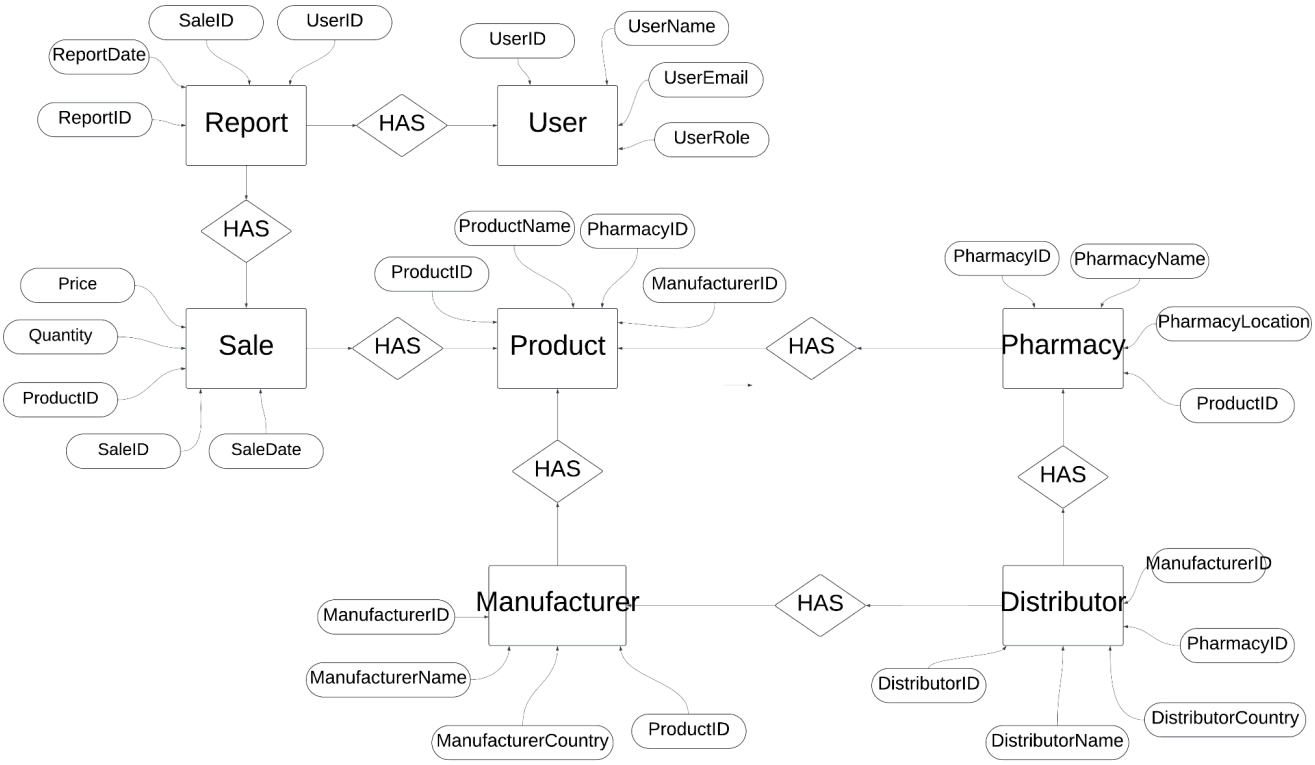


Рис. 3.1 Концептуальна модель бази даних.

3.1.1 Рівні нормалізації

У поточному описі концептуальної моделі основні сутності (наприклад, Report, Sale, User, Product) відображають базові принципи нормалізації. Рівні нормалізації, які можна деталізувати:

- Перша нормальна форма (1NF): Усі атрибути сутностей мають атомарні значення (наприклад, UserName, ProductName). Таблиці не містять списків або вкладених даних.

- Друга нормальна форма (2NF): Залежності між атрибутами усунуті. Атрибути, які не залежать від первинного ключа, переміщені до окремих таблиць (наприклад, інформація про виробників у таблиці Manufacturer).

- Третя нормальна форма (3NF): Усі неключові атрибути залежать тільки від первинного ключа (наприклад, ManufacturerCountry залежить тільки від ManufacturerID, а не від інших атрибутів).

У випадках, коли продуктивність критично важлива (наприклад, для часто виконуваних запитів), можуть використовуватися денормалізовані структури, такі як агреговані таблиці для звітів. Наприклад, створення окремої таблиці з підсумками продажів за період.

3.1.2 Безпека даних

Для забезпечення надійності та захисту інформації в базі даних, система повинна передбачати такі механізми:

1. Контроль доступу:
    - Реалізовано ролі та права доступу:
        - Адміністратор має повний доступ до всіх таблиць і даних для управління системою.
        - Менеджер може переглядати звіти та дані продажів, але не має доступу до конфіденційної інформації, наприклад, контактів виробників.
    - Всі операції CRUD (Create, Read, Update, Delete) контролюються на рівні бази даних через політики доступу.
2. Шифрування:
    - Конфіденційні дані, такі як контактна інформація користувачів (UserEmail) та інформація про виробників (ManufacturerCountry), повинні зберігатися у зашифрованому вигляді.
    - Використання алгоритму AES-256 для забезпечення високого рівня захисту даних.
3. Логування дій:
    - Запроваджено журнал дій користувачів:

- Усі операції, які виконуються користувачами можуть зберігаються в додатковій таблиці AuditLog, що містить поля:
    - ActionID (ідентифікатор дії),
    - UserID (користувач, що виконав дію),
    - ActionType (тип дії: створення, оновлення, видалення),
    - ActionDate (дата виконання дії).
- Журнал дозволяє виявити несанкціоновані дії та моніторити використання системи.

4. Резервне копіювання:
    - Передбачено регулярне резервне копіювання бази даних для мінімізації втрат у разі аварійних ситуацій.
5. Додаткові заходи:
    - Встановлено обмеження на кількість невдалих спроб входу в систему.
    - Забезпечено автоматичне видалення тимчасових файлів після обробки.

3.2 Логічна модель бази даних проєкту

Логічна модель бази даних — це детальне представлення структури даних, яке включає в себе точні визначення таблиць, атрибутів та їх типів даних, а також ключових зв'язків між таблицями. Логічна модель служить основою для фізичної реалізації бази даних. Вона забезпечує цілісність, несуперечність і узгодженість даних, що зберігаються в базі даних.

Основні таблиці та їх атрибути (Рис. 3.2):

1. Report (Звіт)
    - ReportID (Ідентифікатор звіту) — INT, PRIMARY KEY
    - ReportDate (Дата звіту) — DATE
    - SaleID (Ідентифікатор продажу) — INT, FOREIGN KEY (посилається на Sale.SaleID)
    - UserID (Ідентифікатор юзера) — INT, FOREIGN KEY (посилається на

User.UserID)

2. User (Користувач)
    - UserID (Ідентифікатор користувача) — INT, PRIMARY KEY
    - UserName (Ім'я користувача) — VARCHAR(255)
    - UserEmail (Електронна пошта користувача) — VARCHAR(255)
    - UserRole (Роль користувача) — VARCHAR(50)
3. Sale (Продаж)
    - SaleID (Ідентифікатор продажу) — INT, PRIMARY KEY
    - SaleDate (Дата продажу) — DATE
    - ProductID (Ідентифікатор продукту) — INT, FOREIGN KEY (посилається на Product.ProductID)
    - Quantity (Кількість) — INT
    - Price (Ціна) — DECIMAL(10, 2)
4. Product (Продукт)
    - ProductID (Ідентифікатор продукту) — INT, PRIMARY KEY
    - ProductName (Назва продукту) — VARCHAR(255)
    - ManufacturerID (Ідентифікатор виробника) — INT, FOREIGN KEY (посилається на Manufacturer.ManufacturerID)
    - PharmacyID (Ідентифікатор аптеки) — INT, FOREIGN KEY (посилається на Pharmacy.PharmacyID)
5. Pharmacy (Аптека)
    - PharmacyID (Ідентифікатор аптеки) — INT, PRIMARY KEY
    - PharmacyName (Назва аптеки) — VARCHAR(255)
    - PharmacyLocation (Місцезнаходження аптеки) — VARCHAR(255)
    - ProductID (Ідентифікатор продукту) — INT, FOREIGN KEY (посилається на Product.ProductID)
6. Manufacturer (Виробник)
    - ManufacturerID (Ідентифікатор виробника) — INT, PRIMARY KEY

- ManufacturerName (Назва виробника) — VARCHAR(255)
- ManufacturerCountry (Країна виробника) — VARCHAR(255)
- ProductID (Ідентифікатор продукту) — INT, FOREIGN KEY (посилається на Product.ProductID)

7. Distributor (Дистриб'ютор)
    - DistributorID (Ідентифікатор дистриб'ютора) — INT, PRIMARY KEY
    - DistributorName (Назва дистриб'ютора) — VARCHAR(255)
    - DistributorCountry (Країна дистриб'ютора) — VARCHAR(255)
    - ManufacturerID (Ідентифікатор виробника) — INT, FOREIGN KEY (посилається на Manufacturer.ManufacturerID)
    - PharmacyID (Ідентифікатор аптеки) — INT, FOREIGN KEY (посилається на Pharmacy.PharmacyID)

Зв'язки між таблицями:

- Report посилається на Sale через атрибут SaleID.
- User може мати доступ до декількох Report.
- Sale посилається на Product через атрибут ProductID.
- Product посилається на Manufacturer через атрибут ManufacturerID.
- Product посилається на Pharmacy через атрибут PharmacyID.
- Pharmacy посилається на Product через атрибут ProductID.
- Manufacturer посилається на Product через атрибут ProductID.
- Distributor посилається на Manufacturer через атрибут ManufacturerID.
- Distributor посилається на Pharmacy через атрибут PharmacyID.

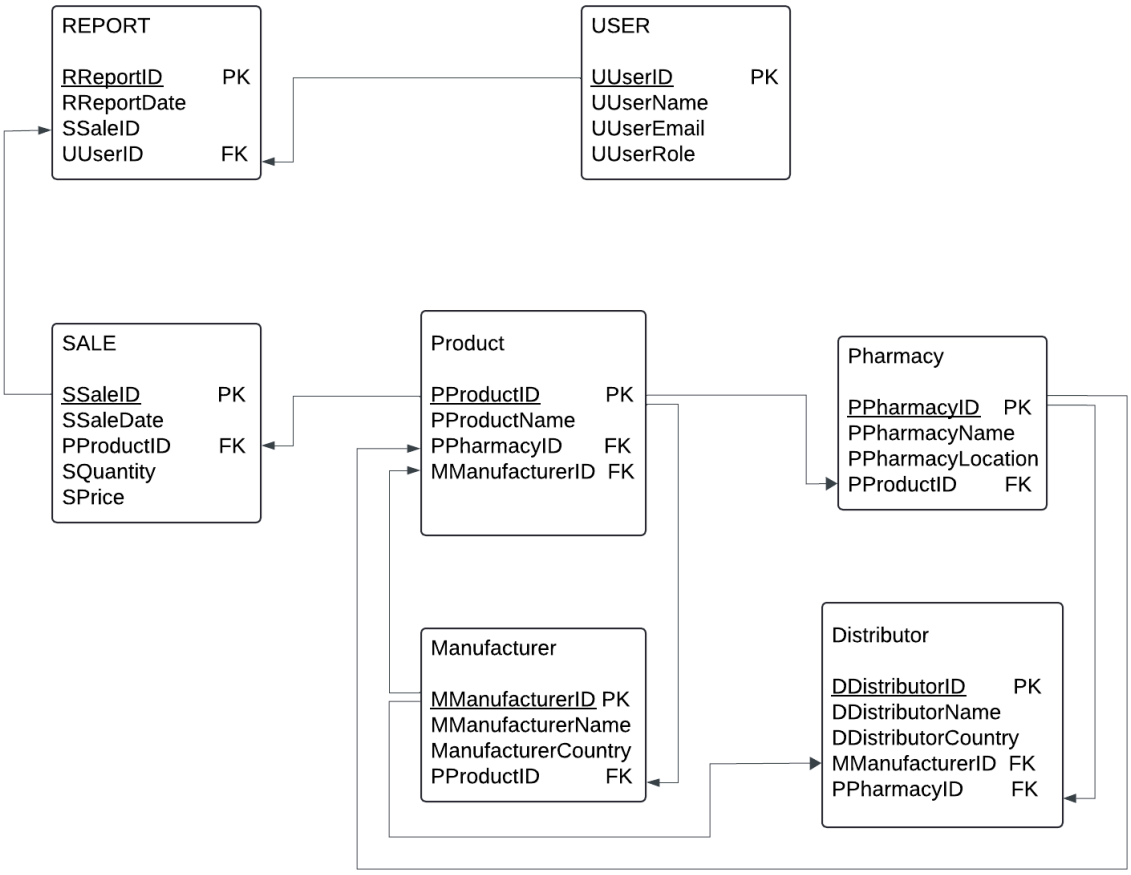


Рис. 3.2 Логічна модель бази даних

Фрагменти коду створення таблиць:

```
1 CREATE TABLE User (
2     UserID SERIAL PRIMARY KEY,
3     UserName VARCHAR(255) NOT NULL,
4     UserEmail VARCHAR(255) NOT NULL,
5     UserRole VARCHAR(50)
6 );
```

Рис. 3.3 Фрагмент створення таблиці User

```sql
CREATE TABLE Report (
    ReportID SERIAL PRIMARY KEY,
    ReportDate DATE NOT NULL,
    SaleID INT NOT NULL,
    UserID INT NOT NULL,
    FOREIGN KEY (SaleID) REFERENCES Sale(SaleID),
    FOREIGN KEY (UserID) REFERENCES User(UserID)
);
```

Рис. 3.4 Фрагмент створення таблиці Report

```sql
CREATE TABLE Sale (
    SaleID SERIAL PRIMARY KEY,
    SaleDate DATE NOT NULL,
    ProductID INT NOT NULL,
    Quantity INT NOT NULL,
    Price DECIMAL(10, 2) NOT NULL,
    FOREIGN KEY (ProductID) REFERENCES Product(ProductID)
);
```

Рис. 3.5 Фрагмент створення таблиці Sale

```sql
CREATE TABLE Manufacturer (
    ManufacturerID SERIAL PRIMARY KEY,
    ManufacturerName VARCHAR(255) NOT NULL,
    ManufacturerCountry VARCHAR(255),
    ProductID INT,
    FOREIGN KEY (ProductID) REFERENCES Product(ProductID)
);
```

Рис. 3.6 Фрагмент створення таблиці Manufacturer

```sql
CREATE TABLE Product (
    ProductID SERIAL PRIMARY KEY,
    ProductName VARCHAR(255) NOT NULL,
    ManufacturerID INT NOT NULL,
    PharmacyID INT NOT NULL,
    FOREIGN KEY (ManufacturerID) REFERENCES Manufacturer(ManufacturerID),
    FOREIGN KEY (PharmacyID) REFERENCES Pharmacy(PharmacyID)
);
```

Рис. 3.7 Фрагмент створення таблиці Product

```sql
CREATE TABLE Pharmacy (
    PharmacyID SERIAL PRIMARY KEY,
    PharmacyName VARCHAR(255) NOT NULL,
    PharmacyLocation VARCHAR(255),
    ProductID INT NOT NULL,
    FOREIGN KEY (ProductID) REFERENCES Product(ProductID)
);
```

Рис. 3.8 Фрагмент створення таблиці Pharmacy

```sql
CREATE TABLE Distributor (
    DistributorID SERIAL PRIMARY KEY,
    DistributorName VARCHAR(255) NOT NULL,
    DistributorCountry VARCHAR(255),
    ManufacturerID INT NOT NULL,
    PharmacyID INT NOT NULL,
    FOREIGN KEY (ManufacturerID) REFERENCES Manufacturer(ManufacturerID),
    FOREIGN KEY (PharmacyID) REFERENCES Pharmacy(PharmacyID)
);
```

Рис. 3.9 Фрагмент створення таблиці Distributor

3.2.1 Обґрунтування вибору PostgreSQL

- PostgreSQL була обрана як основна база даних проєкту через її високу продуктивність, масштабованість і відповідність вимогам реляційного моделювання, що є критично важливим для управління даними про

продажі фармацевтичної продукції.

- PostgreSQL забезпечує підтримку складних запитів, транзакцій та реплікації, що дозволяє ефективно обробляти великі обсяги даних і забезпечувати їхню цілісність.

Особливості PostgreSQL для реалізації проєкту:

- Реляційна структура: Відповідає логічній моделі даних, яка була розроблена для систематизації даних про продажі.
- Розширення: PostgreSQL підтримує додаткові модулі, наприклад, для геопросторового аналізу (PostGIS), що може бути корисним для аналізу даних продажів по регіонах.
- Безпека: Вбудовані механізми шифрування та контролю доступу гарантують високий рівень безпеки даних.

Практичні переваги PostgreSQL:

- Використання JSONB для зберігання напівструктурованих даних дозволяє інтегрувати не тільки структуровані, але й неструктуровані дані, що може знадобитися для додаткових джерел інформації.
- Автоматичне масштабування і можливість кластеризації дозволяють адаптувати систему до зростаючого обсягу даних.

Технічна реалізація:

- Архітектура бази даних побудована з урахуванням можливостей PostgreSQL, таких як індексація, яка дозволяє швидко здійснювати пошук і фільтрацію даних.
- Планується впровадження реплікації для підвищення доступності системи і забезпечення безперебійної роботи.

Налаштування оптимальної конфігурації PostgreSQL:

1. Буферизація:
    - Використано параметр shared_buffers, який було налаштовано на рівні 25-40% від обсягу оперативної пам'яті сервера, для ефективного

кешування даних, що зменшує кількість звернень до диску.

- Налаштовано work_mem, що дозволило оптимізувати операції сортування та роботи з індексами.

2. Кешування:
   - Увімкнено параметр effective_cache_size, що відповідає за використання кешу ОС, дозволяючи покращити планування виконання запитів.
   - Використано механізм pg_stat_statements для аналізу продуктивності запитів, що забезпечило їх оптимізацію.

Використання процедур і тригерів для автоматизації задач:

1. Реалізовано тригери для автоматичного оновлення пов’язаних таблиць у разі внесення змін до основних даних.

   Завдання: Оновлення таблиці Report при зміні даних у таблиці Sale.

```
1 -- Функція для автоматичного оновлення звітів при зміні продажу
2 CREATE OR REPLACE FUNCTION update_reports_on_sale_update()
3 RETURNS TRIGGER AS $$
4 BEGIN
5     UPDATE Report
6     SET ReportDate = CURRENT_DATE
7     WHERE SaleID = NEW.SaleID;
8     RETURN NEW;
9 END;
10 $$ LANGUAGE plpgsql;
11
12 -- Тригер для запуску функції при оновленні таблиці Sale
13 CREATE TRIGGER trigger_update_reports_on_sale_update
14 AFTER UPDATE ON Sale
15 FOR EACH ROW
16 EXECUTE FUNCTION update_reports_on_sale_update();
```

Рис. 3.10 Фрагмент функції для перевірки цілісності даних у таблиці Sale

2. Створено процедури для періодичного очищення історичних даних, що дозволяє зменшити обсяг збереженої інформації без втрати важливих відомостей.

   Завдання: Видалення старих звітів з таблиці Report.

```sql
-- Функція для очищення старих звітів (старше одного року)
CREATE OR REPLACE FUNCTION clean_old_reports()
RETURNS VOID AS $$
BEGIN
    DELETE FROM Report
    WHERE ReportDate < NOW() - INTERVAL '1 year';
END;
$$ LANGUAGE plpgsql;

-- Тригер для запуску очищення перед вставкою нового звіту
CREATE OR REPLACE FUNCTION trigger_clean_old_reports()
RETURNS TRIGGER AS $$
BEGIN
    PERFORM clean_old_reports();
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

-- Тригер для очищення звітів перед вставкою нового звіту
CREATE TRIGGER trigger_clean_old_reports_on_insert
BEFORE INSERT ON Report
FOR EACH ROW
EXECUTE FUNCTION trigger_clean_old_reports();
```

Рис. 3.11 Фрагмент функції для очищення старих звітів

3. Додано функції для перевірки цілісності даних перед внесенням змін, що гарантує високу якість та консистентність даних у базі.

Завдання: Перевірка відповідності зв'язків перед вставкою нових даних.

```sql
-- Функція для перевірки цілісності даних у таблиці Sale
CREATE OR REPLACE FUNCTION validate_sale_data()
RETURNS TRIGGER AS $$
BEGIN
    IF NOT EXISTS (SELECT 1 FROM Product WHERE ProductID = NEW.ProductID) THEN
        RAISE EXCEPTION 'ProductID % does not exist in Product table', NEW.ProductID;
    END IF;

    IF NEW.Quantity <= 0 THEN
        RAISE EXCEPTION 'Quantity must be greater than 0';
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

-- Тригер для перевірки даних перед вставкою у таблицю Sale
CREATE TRIGGER trigger_validate_sale_data
BEFORE INSERT OR UPDATE ON Sale
FOR EACH ROW
EXECUTE FUNCTION validate_sale_data();
```

Рис. 3.12 Фрагмент функції для перевірки цілісності даних у таблиці Sale

3.2.2 Масштабованість бази даних

1. Зростання обсягу даних:
    - У разі збільшення кількості даних у базі, система підтримує додавання нових таблиць або розширення існуючих без впливу на продуктивність завдяки використанню індексів та оптимізованих запитів.
    - Наявність механізмів архівування старих даних дозволяє зменшити навантаження на оперативну частину бази.
2. Масштабування на рівні таблиць:
    - Завдяки механізмам партиціонування PostgreSQL можна

розділити великі таблиці на частини за певними критеріями, наприклад, за датою або регіоном.

3. Розподілене зберігання:
    - PostgreSQL може бути інтегрована з рішеннями для горизонтального масштабування, такими як Citus, що дозволяє зберігати дані на декількох вузлах (кластерах) з автоматичним балансуванням навантаження.
4. Хмарні платформи:
    - Для забезпечення динамічного масштабування можливе використання хмарних платформ (AWS, Azure, GCP), які надають керовані бази даних з автоматичною підтримкою масштабування ресурсів.
5. Інструменти моніторингу:

Використання інструментів для моніторингу (наприклад, pgAdmin, Prometheus) дозволяє своєчасно виявляти проблеми продуктивності та планувати модернізацію системи.

У рамках дипломної роботи, я б хотів акцентувати особливу увагу саме на **реплікації даних**, що може бути використана як основний інструмент масштабування.

Реплікація даних — це процес копіювання даних з однієї бази даних на іншу, що забезпечує доступність, стійкість до збоїв та підвищення продуктивності системи. У проєктованій базі даних реплікація може реалізовуватись наступними способами:

1. Типи реплікації
    - Синхронна реплікація:
        - Дані записуються одночасно в головну (primary) і вторинну (replica) бази даних. Це гарантує, що всі репліки завжди містять найактуальнішу інформацію.

        - Підходить для критично важливих даних, де консистентність є пріоритетом.
        - Недолік: вища затримка під час запису через необхідність підтвердження від всіх реплік.
    - Асинхронна реплікація:
        - Головна база даних виконує запис, а репліки оновлюються із затримкою. Це забезпечує швидший запис у головній базі.
        - Підходить для систем, де пріоритет — швидкодія, а не моментальна консистентність (наприклад, для аналітичних запитів).
    - Логічна реплікація:
        - Реплікація обраних даних (таблиць чи рядів) замість всієї бази.
        - Підходить для інтеграції з іншими системами або для підтримки специфічних бізнес-завдань.

2. Архітектурні варіанти реплікації
    - Майстер-слейв (Master-Slave):
        - Головний вузол (master) обробляє всі операції запису, а дочірні вузли (slaves) обробляють лише операції читання.
        - Переваги:
            - Розвантаження головної бази.
            - Підвищення швидкості обробки запитів на читання.
    - Майстер-мастер (Master-Master):
        - Усі вузли можуть обробляти як операції запису, так і читання. Зазвичай використовується в географічно розподілених системах.
        - Переваги:
            - Висока доступність даних.
            - Менші затримки для локальних запитів.

        - Недолік: можливі конфлікти записів, які потребують розв'язання.
    - Каскадна реплікація:
        - Дані передаються через проміжні репліки. Це дозволяє зменшити навантаження на головний вузол.
        - Підходить для великих систем з великою кількістю реплік.

3. Резервування та відновлення
    - Реплікація сприяє підвищенню стійкості до збоїв, оскільки вторинні репліки можуть бути використані як резервні у випадку відмови основної бази.
    - Автоматичне перемикання на репліку (failover) дозволяє мінімізувати час простою.
4. Інструменти для реплікації у PostgreSQL
    - Streaming Replication:
        - Вбудована функція PostgreSQL, яка забезпечує поточну реплікацію змін.
    - Logical Replication:
        - Дозволяє вибірково реплікувати окремі таблиці або потоки змін.
    - Third-Party Solutions:
        - Інструменти, такі як Citus, Bucardo або Slony-I, можуть забезпечувати розширені функції реплікації, зокрема географічно розподілену реплікацію.
5. Переваги реплікації
    - Підвищення продуктивності за рахунок розподілу навантаження між вузлами.
    - Висока доступність даних навіть у випадку збоїв.
    - Можливість створення спеціалізованих реплік для виконання важких аналітичних запитів без навантаження на основну базу.

Моделювання сценаріїв роботи: Приклади типових запитів до бази даних

Для демонстрації роботи бази даних розглянемо кілька прикладів типових запитів, які можуть використовуватися в системі збору, систематизації та аналізу даних про продажі фармацевтичної продукції:

1. Отримання інформації про продажі за визначений період:

```sql
SELECT
    product_name,
    SUM(quantity_sold) AS total_quantity,
    SUM(total_price) AS total_revenue
FROM
    sales
WHERE
    sale_date BETWEEN '2024-01-01' AND '2024-12-31'
GROUP BY
    product_name
ORDER BY
    total_revenue DESC;
```

Рис. 3.13 Фрагмент вибірки з найпопулярнішими продуктами за рік, обсягами продажів і загальним доходом.

2. Пошук аптек із найвищими продажами в конкретному регіоні:

```sql
SELECT
    pharmacy_name,
    SUM(total_price) AS total_revenue
FROM
    sales
JOIN
    pharmacies ON sales.pharmacy_id = pharmacies.pharmacy_id
WHERE
    region = 'Київ'
GROUP BY
    pharmacy_name
ORDER BY
    total_revenue DESC
LIMIT 10;
```

Рис. 3.14 Фрагмент вибірки із списком 10 аптек у Києві з найвищими доходами.

3. Перевірка наявності дублюючих записів у таблиці продажів:

```sql
SELECT
    sale_id,
    COUNT(*) AS duplicate_count
FROM
    sales
GROUP BY
    sale_id
HAVING
    COUNT(*) > 1;
```

Рис. 3.15 Фрагмент вибірки ідентифікаторів продажів із кількістю повторень, які потребують перевірки.

4. Виявлення найпопулярніших дистриб'юторів за обсягом поставок:

```sql
SELECT
    distributor_name,
    SUM(quantity_supplied) AS total_quantity
FROM
    supplies
JOIN
    distributors ON supplies.distributor_id = distributors.distributor_id
GROUP BY
    distributor_name
ORDER BY
    total_quantity DESC;

```

Рис. 3.16 Фрагмент вибірки списку дистриб'юторів і кількість продукції, яку вони поставили.

5. Пошук даних про продукт із низьким рівнем продажів:

```sql
SELECT
    product_name,
    SUM(quantity_sold) AS total_quantity
FROM
    sales
WHERE
    quantity_sold < 50
GROUP BY
    product_name
ORDER BY
    total_quantity ASC;
```

Рис. 3.17 Фрагент вибірки продуктів з низькими продажами, які можуть потребувати додаткових маркетингових зусиль.

6. Аналіз сезонності продажів:

```sql
SELECT
    EXTRACT(MONTH FROM sale_date) AS sale_month,
    SUM(total_price) AS total_revenue
FROM
    sales
GROUP BY
    sale_month
ORDER BY
    sale_month;
```

Рис. 3.18 Фрагмент вибірки загальних доходів за кожен місяць, що дозволяє визначити пік продажів у році.

3.3 Структура програмного забезпечення – Опис структури програмного забезпечення.

Структура програмного забезпечення для інформаційної системи збору, систематизації та аналізу даних про продажі фармацевтичної продукції повинна бути модульною та гнучкою, щоб забезпечити легке розширення та підтримку. Нижче наведено основні компоненти структури програмного забезпечення, які

включають різні рівні та модулі системи.

Архітектура системи. Архітектура програмного забезпечення базується на сервісно-орієнтованій архітектурі (SOA), де кожен окремий сервіс має свою трирівневу архітектуру (*N-Tier Architecture*), яка складається з наступних рівнів:

1. *Рівень презентації (Presentation Layer)*
2. *Рівень бізнес-логіки (Business Logic Layer)*
3. *Рівень даних (Data Layer)*

Рівень презентації (Presentation Layer)

Цей рівень відповідає за взаємодію з користувачами системи. Він включає користувацькі інтерфейси, через які користувачі можуть отримувати доступ до функціональності системи.

Основні компоненти рівня презентації:

API: Інтерфейс програмування додатків, який забезпечує взаємодію з іншими системами та додатками.

Рівень бізнес-логіки (Business Logic Layer)

Цей рівень відповідає за обробку та аналіз даних. Він містить основні алгоритми та логіку, необхідну для функціонування системи.

Рівень даних (Data Layer)

Цей рівень відповідає за зберігання та управління даними, використовуючи виключно реляційну базу даних PostgreSQL. Він включає базу даних та інструменти для роботи з нею.

1. *Користувацькі інтерфейси:*
   - Веб-додаток (HTML, CSS, JavaScript)
   - API (RESTful)
2. *Сервіси:*
   - Сервіс збору даних (Java)
   - Сервіс очищення та нормалізації даних (Java)
   - Аналітичний сервіс (Java)

- Сервіс аутентифікації та авторизації (OAuth 2.0, JWT)
- Сервіс шифрування
- Проксі веб-сервер (Java, JavaScript)

3. *База даних:*
   - Реляційна СУБД (PostgreSQL)
4. *Інструменти для управління базою даних:*
   - Адміністрування бази даних (pgAdmin)
   - Резервне копіювання та відновлення даних (cron jobs, shell scripts)
5. *Інструменти для інтеграції:*
   - Інтерфейси для обміну даними (SOAP, RESTful API)
   - Інтеграція з ERP та CRM системами (SAP, Salesforce)
6. *Ціни:*
   - Модуль управління цінами, який зберігає та обробляє інформацію про ціни на продукцію, знижки та спеціальні пропозиції.
7. *Платформа для розгортання:*
   - Додаток буде розгорнутий на веб-сервері Apache Tomcat, який забезпечує надійну та стабільну роботу Java-додатків.

Технічні аспекти

Для розробки та впровадження програмного забезпечення будуть використовуватися сучасні технології та інструменти, що забезпечують високу продуктивність, надійність та безпеку системи.

Основні технології:

- Мови програмування: Java, JavaScript
- Фреймворки: Spring Boot, React
- Бази даних: PostgreSQL
- Інструменти контейнеризації: Docker, Kubernetes
- Системи управління версіями: Git (GitLab)
- Інструменти CI/CD: Jenkins, GitLab CI/CD

- Веб-сервер: Apache Tomcat

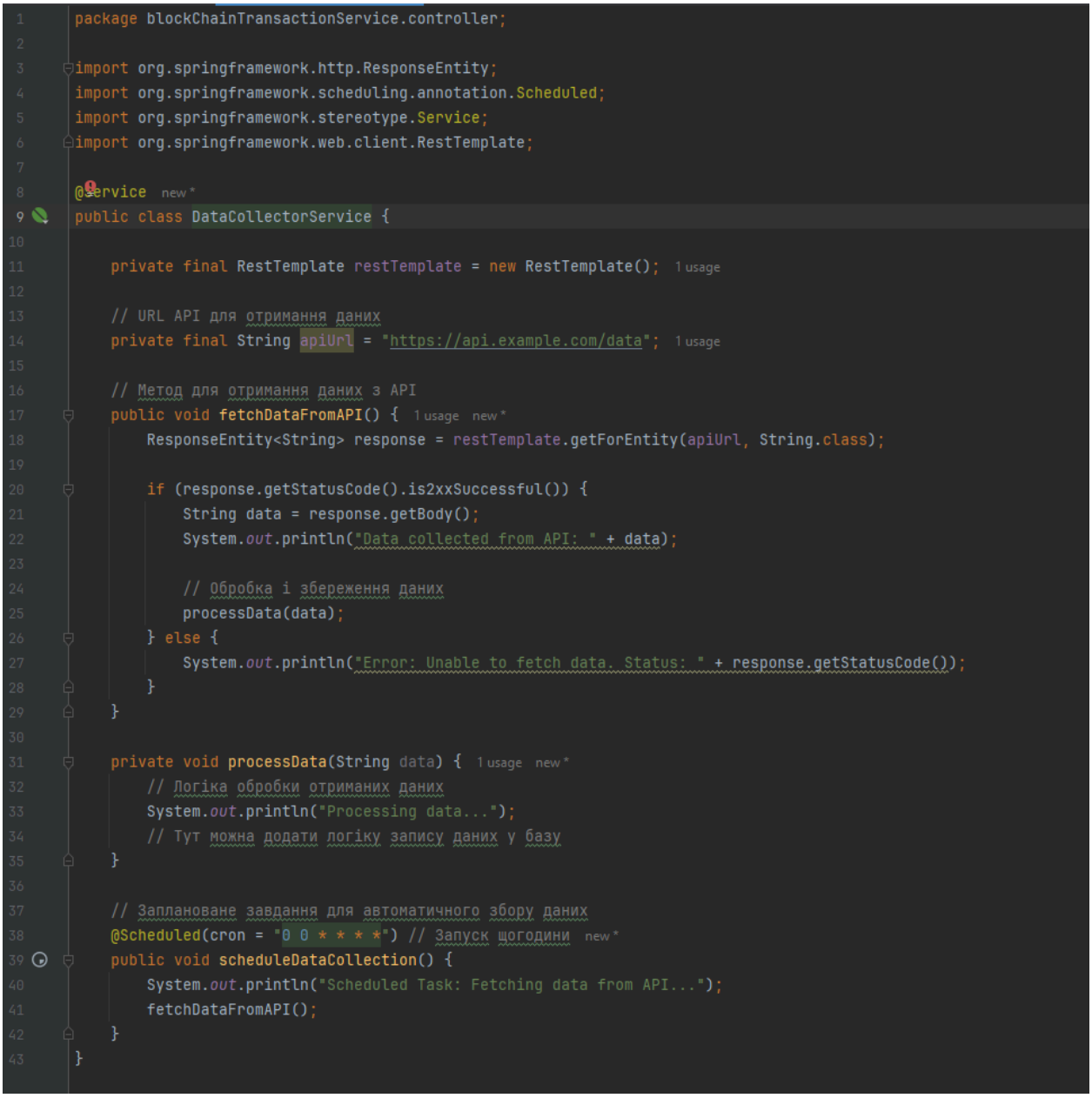
Структура програмного забезпечення для інформаційної системи збору, систематизації та аналізу даних про продажі фармацевтичної продукції включає трирівневу архітектуру, що складається з рівня презентації, рівня бізнес-логіки та рівня даних. Модульна та гнучка структура забезпечує легке розширення та підтримку системи, а також інтеграцію з іншими інформаційними системами. Використання сучасних технологій та інструментів дозволить забезпечити високу продуктивність, надійність та безпеку системи.

3.4 Розробка алгоритмів та інтерфейсів – Опис алгоритмів та інтерфейсів програмного забезпечення.

Алгоритми програмного забезпечення

Розробка алгоритмів для інформаційної системи включає створення чітких і ефективних алгоритмів для збору, очищення, систематизації, аналізу даних, а також управління безпекою і доступом. Основні алгоритми, які будуть реалізовані в системі, включають:

1. Алгоритм збору даних
    - Автоматизований збір даних:
        - Використання API для отримання даних з різних джерел (аптеки, дистриб'ютори, виробники).
        - Заплановані завдання (cron jobs) для регулярного збору даних.



```java
package blockChainTransactionService.controller;

import org.springframework.http.ResponseEntity;
import org.springframework.scheduling.annotation.Scheduled;
import org.springframework.stereotype.Service;
import org.springframework.web.client.RestTemplate;

@Service  new *
public class DataCollectorService {

    private final RestTemplate restTemplate = new RestTemplate();  1 usage

    // URL API для отримання даних
    private final String apiUrl = "https://api.example.com/data";  1 usage

    // Метод для отримання даних з API
    public void fetchDataFromAPI() {  1 usage  new *
        ResponseEntity<String> response = restTemplate.getForEntity(apiUrl, String.class);

        if (response.getStatusCode().is2xxSuccessful()) {
            String data = response.getBody();
            System.out.println("Data collected from API: " + data);

            // Обробка і збереження даних
            processData(data);
        } else {
            System.out.println("Error: Unable to fetch data. Status: " + response.getStatusCode());
        }
    }

    private void processData(String data) {  1 usage  new *
        // Логіка обробки отриманих даних
        System.out.println("Processing data...");
        // Тут можна додати логіку запису даних у базу
    }

    // Заплановане завдання для автоматичного збору даних
    @Scheduled(cron = "0 0 * * * *") // Запуск щогодини  new *
    public void scheduleDataCollection() {
        System.out.println("Scheduled Task: Fetching data from API...");
        fetchDataFromAPI();
    }
}
```

Рис. 4.1 Фрагмент коду по автоматизованому збору даних в Java (Автоматизований збір)

- Ручний збір даних:
    - Веб-форми для введення даних вручну, якщо автоматизований збір неможливий.

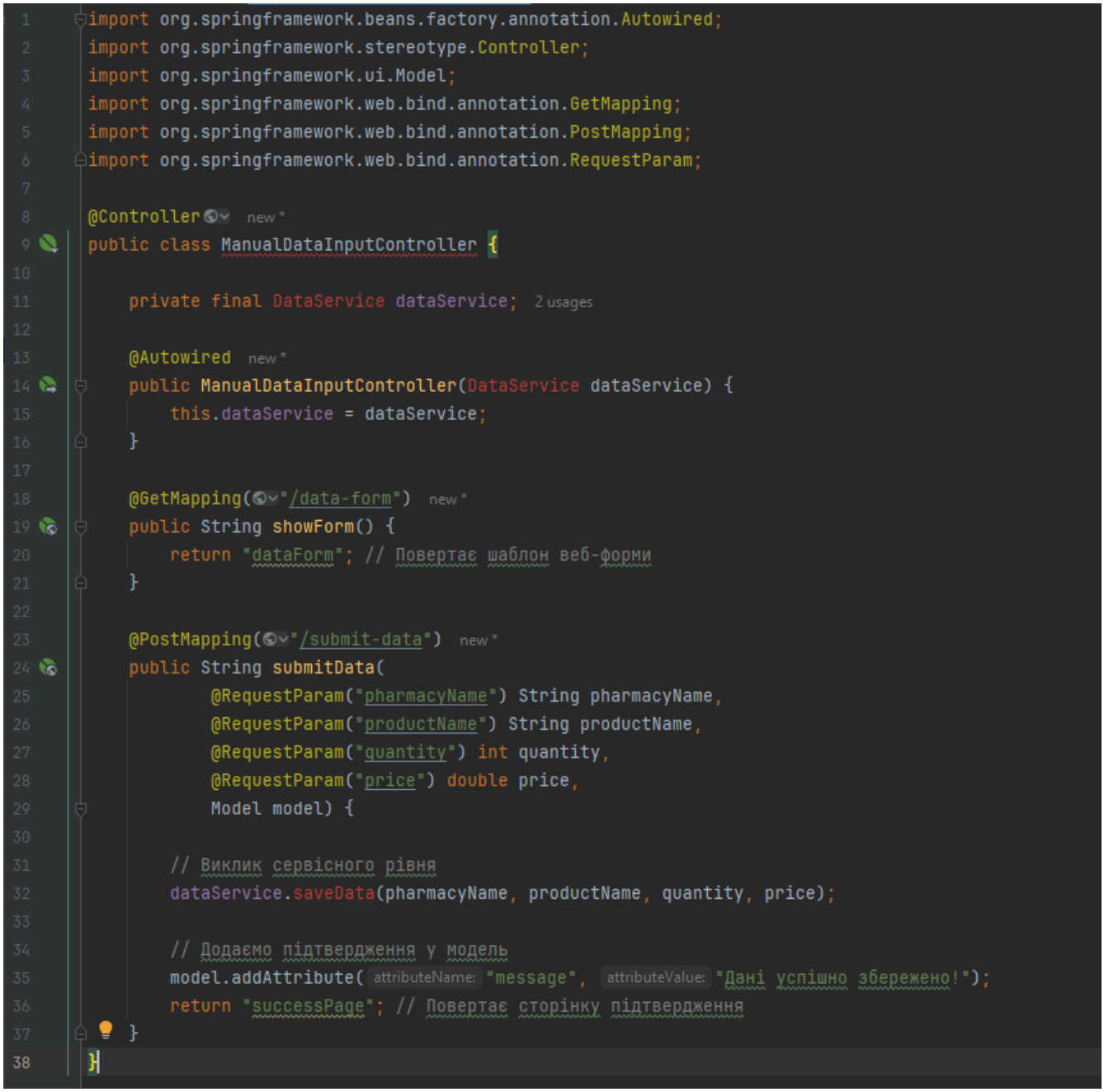


Рис. 4.2 Фрагмент коду по ручному збору даних на презентаційному рівні java застосунку

2. Алгоритм очищення та нормалізації даних
    - Виявлення та усунення дублікатів:
        - Використання хеш-функцій для виявлення дублікатів.
        - Об'єднання дублікатів на основі ключових атрибутів (наприклад,

ProductID).

- Нормалізація даних:
    - Перетворення даних у стандартний формат для забезпечення узгодженості.
    - Валідація даних на відповідність визначеним схемам та правилам.

3. Алгоритм систематизації даних
    - Класифікація даних:
        - Використання категорій для систематизації даних про продажі (наприклад, за продуктами, аптеками, виробниками).
    - Агрегація даних:
        - Обчислення загальних показників (наприклад, загальний обсяг продажів, середня ціна) для звітів.
4. Аналітичні алгоритми
    - Регресійний аналіз:
        - Визначення залежностей між різними змінними, такими як ціни, реклама, сезонність.
    - Кластерний аналіз :
        - Групування даних для виявлення схожих патернів та трендів (додаток Б) [14].
    - Прогнозування:
        - Використання моделей часових рядів для прогнозування майбутніх обсягів продажів.
5. Алгоритм безпеки та контролю доступу
    - Аутентифікація користувачів:
        - Використання OAuth 2.0 для забезпечення безпечної аутентифікації.
        - Генерація та валідація токенів JWT для сесій користувачів.

- Авторизація користувачів:
    - Визначення ролей та прав доступу для різних категорій користувачів.
    - Контроль доступу до різних функцій та даних системи на основі ролей користувачів.
- Шифрування даних:
    - Використання шифрування для захисту конфіденційних даних як в транзиті, так і в стані спокою.

Інтерфейси користувачів

Інтерфейси користувачів (UI) розробляються для забезпечення зручного доступу до функціональності системи (додаток Г). Основні інтерфейси включають:

1. Веб-інтерфейс
    - Інтерфейс введення даних:
        - Форми для ручного введення даних про продажі, продукти, аптеки та виробників.
    - Інтерфейс перегляду даних:
        - Таблиці та списки для відображення даних про продажі, звіти, продукти тощо.
    - Інтерфейс аналізу даних:
        - Графіки, діаграми та звіти для візуалізації результатів аналізу даних.
    - Інтерфейс управління користувачами:
        - Інструменти для створення, редагування та видалення користувачів, призначення ролей та прав доступу.
2. API (Application Programming Interface)
    - RESTful API:
        - Ендпоінти для автоматизованого збору, оновлення та видалення

даних.

- Ендпоінти для отримання аналітичних даних та звітів.

- Інтеграційні інтерфейси:
  - API для інтеграції з іншими системами, такими як ERP та CRM.

Приклади інтерфейсів

1. Веб-інтерфейс введення даних:
   - Форма введення продуктів:
     - Поля для введення назви продукту, ідентифікатора виробника, аптеки тощо.
2. Веб-інтерфейс перегляду даних:
   - Таблиця продажів:
     - Відображення списку продажів з можливістю сортування та фільтрації за різними атрибутами .
   - Таблиця продуктів:
     - Відображення списку продуктів з можливістю перегляду детальної інформації про кожен продукт.
3. Веб-інтерфейс аналізу даних:
   - Графіки продажів:
     - Лінійні графіки для відображення тенденцій продажів за певний період. (Рис. 4.2).
   - Діаграми продажів за категоріями:
     - Кругові діаграми для відображення розподілу продажів за категоріями продуктів або аптеками.

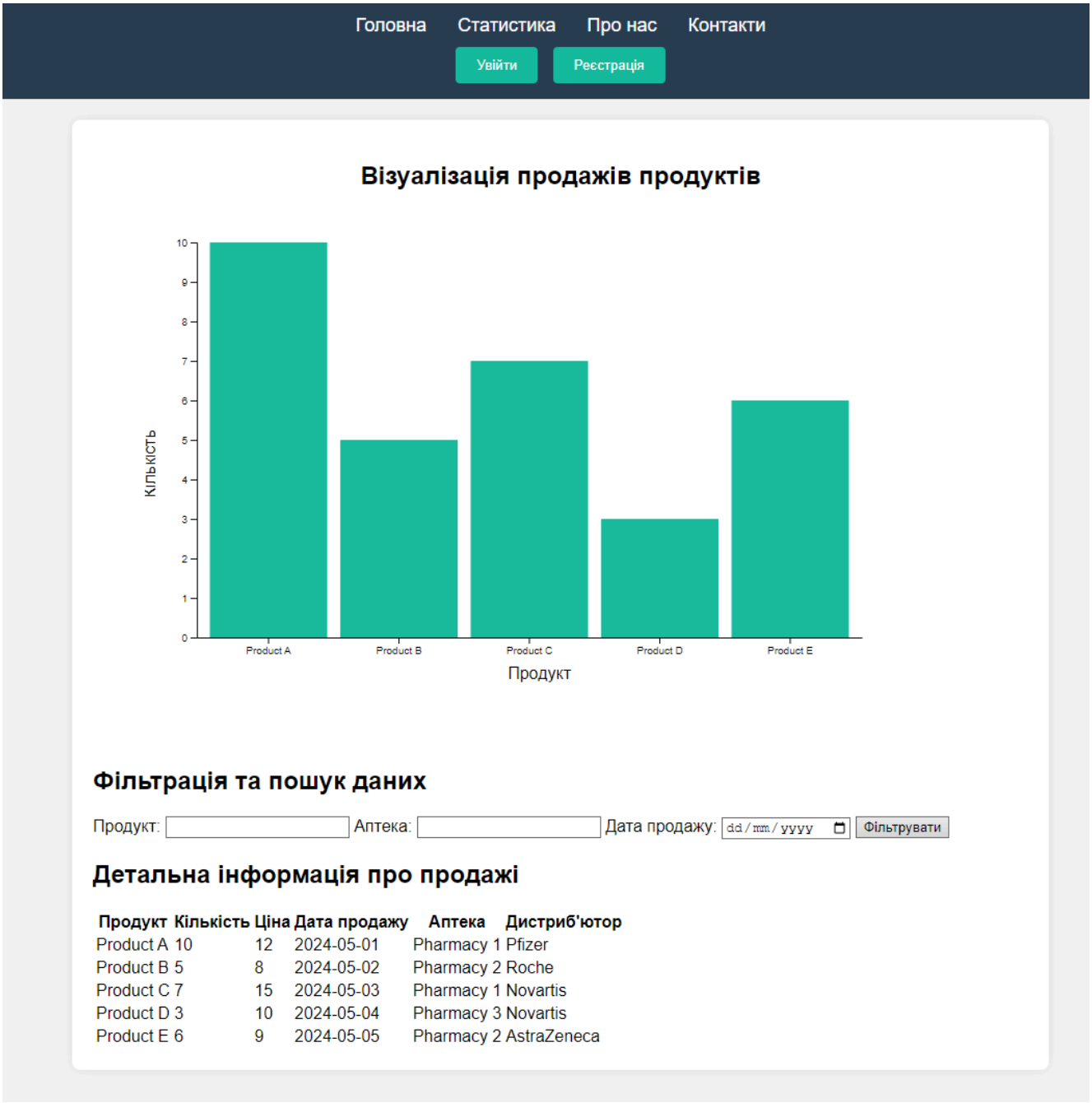


Рис. 4.2 Mockup форми візуалізації, фільтрації та пошуку даних продаж

РОЗДІЛ 4. ПЛАНУВАННЯ ЕЛЕМЕНТІВ УПРАВЛІННЯ ПРОЄКТОМ

4.1 Розробка ієрархічної структури управління проєктом та формування команди проєкту

Organizational Breakdown Structure (OBS) представляє детальний розподіл ролей та обов'язків у проєкті з розробки системи збору, систематизації та аналізу даних про продажі фармацевтичної продукції. OBS відображає, *хто саме* відповідає за кожне завдання у відповідності до WBS (Рис 4.1).

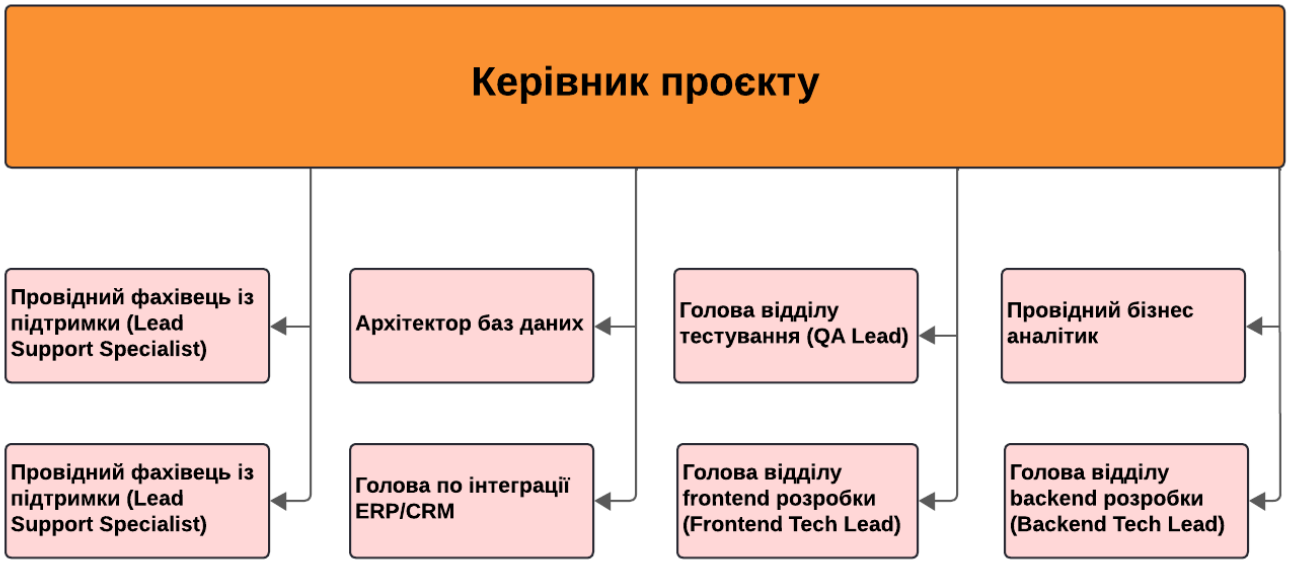


Рис. 4.1 OBS структура

- Керівник проєкту
  - Загальна координація проєкту та контроль всіх етапів.
- Провідний бізнес-аналітик
  - Збір даних про проблеми (1.1.1).
  - Визначення функціональних вимог (1.1.2).
  - Визначення нефункціональних вимог (1.1.3).
- Архітектор баз даних
  - Проєктування бази даних (1.2.1).

    - Створення ER-діаграми (1.2.3).
- Backend Tech Lead
    - Реалізація модулів збору даних (1.3.1).
    - Розробка модулів очищення та нормалізації даних (1.3.2).
    - Розробка аналітичного модуля (1.3.3).
- Frontend Tech Lead
    - Створення інтерфейсу збору даних (1.3.1).
    - Візуалізація аналітичних звітів (1.3.3).
- QA Lead
    - Юніт-тестування (1.4.1).
    - Інтеграційне тестування (1.4.2).
    - Тестування навантаження (1.4.3).
- ERP/CRM-інтегратор
    - Інтеграція системи з ERP/CRM (1.5.2).
- Lead Support Specialist
    - Навчання користувачів (1.5.1).
    - Налаштування системи для перших користувачів (1.6.1).
    - Оптимізація роботи системи на основі відгуків (1.6.3).
- Lead DevOps Enginer
    - Налаштування CI/CD процесів (1.5.3).
    - Моніторинг продуктивності системи (1.6.1).

Підсумок:

Детальна OBS структура відповідає WBS та розподіляє відповідальність між ролями й підзавданнями, забезпечуючи чіткий контроль над виконанням проєкту.

4.1.1. Склад команди проєкту

Склад команди проєкту формується відповідно до завдань та цілей, поставлених у рамках реалізації системи збору, систематизації та аналізу даних з продажу фармацевтичної продукції. Команда включає наступні ролі:

1. Керівник проєкту (Project Manager)
    - Відповідальність: Координація проєкту, управління ресурсами, ризиками, та дотриманням термінів.
    - Основні завдання:
        - Загальний контроль виконання робіт.
        - Управління комунікацією між командами.
        - Моніторинг прогресу та внесення коригувань.
2. Бізнес-аналітик (Business Analyst)
    - Відповідальність: Аналіз вимог, документування функціональних і нефункціональних характеристик.
    - Основні завдання:
        - Збір даних про поточні проблеми управління.
        - Визначення вимог до системи.
        - Формування технічного завдання.
3. Архітектор баз даних (Database Architect)
    - Відповідальність: Проєктування та оптимізація бази даних.
    - Основні завдання:
        - Створення структури бази даних.
        - Розробка ER-діаграми та налаштування взаємозв'язків.
4. Команда розробників
    - Backend-розробники
        - Відповідальність: Реалізація серверної логіки.
        - Завдання:
            - Створення модулів для збору, обробки та аналізу даних.

- Frontend-розробники
    - Відповідальність: Розробка інтерфейсу користувача.
    - Завдання:
        - Візуалізація даних та інтеграція із серверною частиною.

5. QA-тестувальники (QA Engineers)
    - Відповідальність: Перевірка якості та коректності роботи системи.
    - Основні завдання:
        - Юніт-тестування, інтеграційне та навантажувальне тестування.
6. ERP/CRM-інтегратор (Integrator)
    - Відповідальність: Інтеграція системи з існуючими ERP та CRM рішеннями.
    - Основні завдання:
        - Налаштування обміну даними з ERP/CRM.
7. Фахівці з навчання та підтримки (Support Specialists)
    - Відповідальність: Навчання користувачів та надання підтримки після впровадження.
    - Основні завдання:
        - Підготовка документації.
        - Оптимізація роботи системи за відгуками.
8. DevOps-інженер (DevOps Engineer)
    - Відповідальність: Забезпечення стабільності роботи системи та CI/CD процесів.
    - Основні завдання:
        - Налаштування інфраструктури та процесів деплоя.
        - Моніторинг та підтримка продуктивності системи.

4.2 Розробка календарного плану. Планування термінів проєкту

Календарний план – це інструмент управління проєктами, який відображає послідовність виконання завдань, їх тривалість та відповідальних осіб. Він допомагає координувати роботу команди, контролювати дотримання термінів і ефективно розподіляти ресурси. Основна мета календарного плану – забезпечити виконання проєкту у встановлені терміни шляхом чіткого визначення часових рамок для кожного етапу та завдання.

Основна мета календарного плану – забезпечити виконання проєкту у встановлені терміни шляхом чіткого визначення часових рамок для кожного етапу та завдання.

Таблиця 4.1

Календарний план

п/п	**Назва етапу проєкту**	**Планові терміни виконання**	**Відповідальні**
1	Вибір теми та формулювання проблемної області	01.09.2024 04.09.2024	Керівник проєкту, Бізнес-аналітик
2	Аналіз літератури та інформаційних джерел	05.09.2024 10.09.2024	Бізнес-аналітик
3	Постановка задач дослідження	11.09.2024 15.09.2024	Бізнес-аналітик, Керівник
4	Розробка концептуальної моделі системи	16.09.2024 27.09.2024	Архітектор баз даних

Продовження табл. 4.1

5	Розробка математичних моделей	28.09.2024 12.10.2024	Бізнес-аналітик, Архітектор
6	Проєктування бази даних	13.10.2024 24.10.2024	Архітектор баз даних
7	Розробка програмного забезпечення (Backend)	25.10.2024 05.11.2024	Backend-розробники
8	Розробка інтерфейсу користувача (Frontend)	06.11.2024 10.11.2024	Frontend-розробники
9	Тестування функціональних модулів	11.11.2024 15.11.2024	QA-тестувальники
10	Інтеграція з ERP/CRM	16.11.2024 20.11.2024	ERP/CRM-інтегратор
11	Налаштування та впровадження системи	21.11.2024 30.11.2024	DevOps-інженер, Керівник
12	Навчання користувачів	01.12.2024 05.12.2024	Фахівці з навчання
13	Тестування у виробничому середовищі	06.12.2024 09.12.2024	QA-тестувальники
14	Фінальна оптимізація системи	10.12.2024 12.12.2024	Керівник проєкту, Архітектор
15	Оформлення звітної документації	13.12.2024	Керівник проєкту

Календарний план є ключовим інструментом для контролю термінів і координації робіт у проєкті. Завдяки ньому можна відстежувати прогрес,

оптимізувати робочі процеси та забезпечити своєчасне завершення всіх етапів проєкту.

4.3 Віхи

Віхи проєкту — це ключові точки контролю, які визначають завершення важливих етапів або досягнення критичних результатів у межах життєвого циклу проєкту. Вони допомагають оцінити прогрес та забезпечити своєчасне виконання завдань (рис 4.3)

Основні віхи IT-проєкту:

1. Ініціація проєкту
    - Дата завершення: 15.09.2024
    - Результат: Паспорт проєкту, сформована команда, узгоджені цілі.
2. Планування проєкту
    - Дата завершення: 27.09.2024
    - Результат: Декомпозиція WBS, календарний план, графік ресурсів.
3. Розробка концептуальної моделі
    - Дата завершення: 12.10.2024
    - Результат: Концептуальна та логічна модель системи.
4. Розробка програмного забезпечення
    - Дата завершення: 10.11.2024
    - Результат: Реалізовані модулі збору, аналізу даних, інтерфейси.
5. Тестування системи
    - Дата завершення: 22.11.2024
    - Результат: Перевірка функціональності та усунення помилок.
6. Впровадження системи
    - Дата завершення: 09.12.2024

- Результат: Система розгорнута у виробничому середовищі.

7. Завершення проєкту
    - Дата завершення: 12.12.2024
    - Результат: Оформлення звітів, навчання користувачів, фіналізація документації.

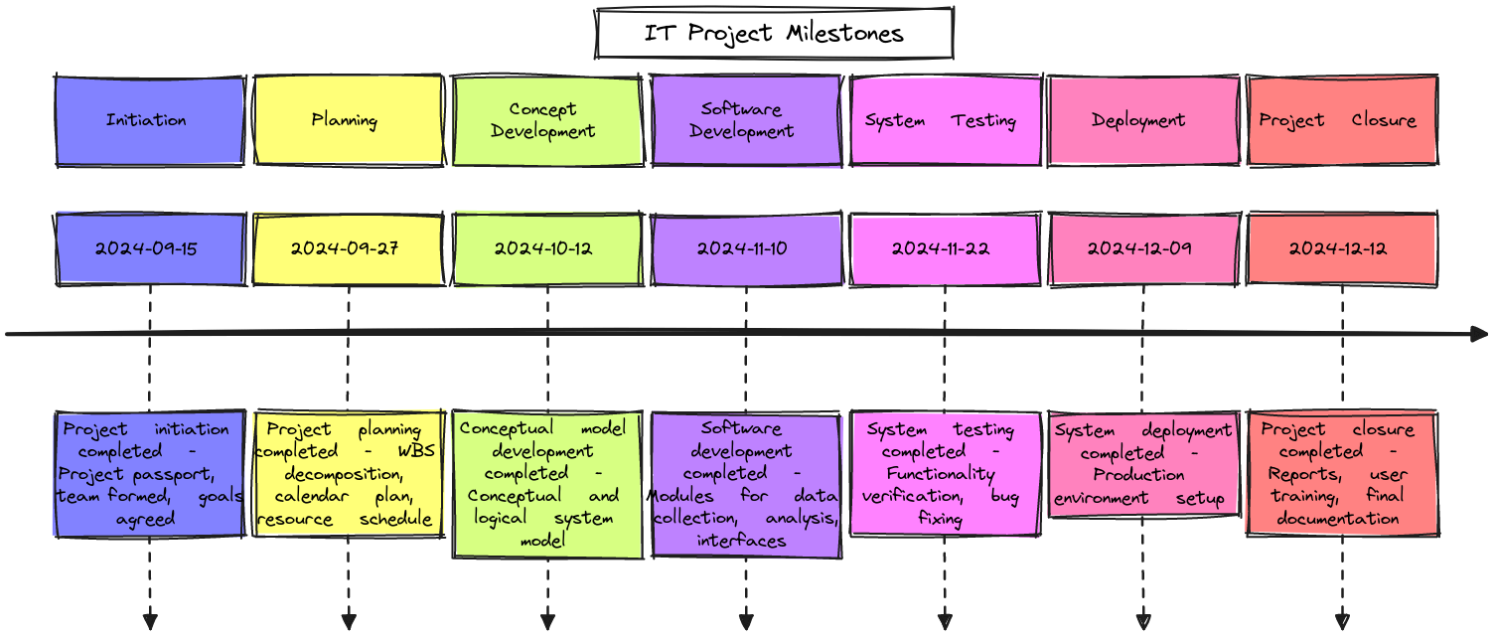


Рис. 4.3 Віхи

4.4 Визначення та планування ресурсів. Ресурсні конфлікти

Визначення та планування ресурсів є одним із критично важливих завдань у процесі управління проєктом. Воно передбачає визначення необхідних людських, матеріальних, фінансових та технологічних ресурсів для виконання завдань проєкту у визначені терміни.

1. Визначення ресурсів

Основні ресурси для реалізації проєкту системи збору та аналізу даних включають:

1. Людські ресурси
    - Керівник проєкту: загальне керівництво та координація.
    - Бізнес-аналітик: аналіз вимог, постановка задач.
    - Архітектор баз даних: проєктування та оптимізація бази даних.
    - Backend-розробники: реалізація серверної логіки.
    - Frontend-розробники: створення інтерфейсу користувача.
    - QA-тестувальники: забезпечення якості системи.
    - ERP/CRM-інтегратор: інтеграція із зовнішніми системами.
    - DevOps-інженер: налаштування CI/CD та моніторинг системи.
    - Фахівці з підтримки: навчання персоналу та підтримка системи.
2. Технологічні ресурси
    - Програмне забезпечення: Java, Spring Boot, React, PostgreSQL, Docker.
    - Інструменти для управління проєктами: Jira, Trello, MS Project.
    - Інфраструктура: сервери для розгортання системи, хмарні сервіси (AWS/Azure).
3. Фінансові ресурси
    - Бюджет проєкту, що включає оплату праці, закупівлю ліцензій та обладнання, витрати на навчання та підтримку.

2. Планування ресурсів

Планування ресурсів здійснюється на основі календарного плану робіт (4.3) та WBS структури. Основні кроки:

1. Ідентифікація завдань: визначення всіх робіт, що потребують ресурсів.
2. Оцінка ресурсів для кожного завдання:
    - Людські ресурси: кількість годин роботи фахівців.
    - Технологічні ресурси: необхідні інструменти та технології.

- Фінансові ресурси: розрахунок витрат на кожен етап.

3. Розподіл ресурсів за завданнями: створення графіку використання ресурсів.
4. Оптимізація ресурсів: балансування завантаження команди для уникнення простоїв.

3. Ресурсні конфлікти

Ресурсні конфлікти виникають, коли декілька завдань одночасно потребують однакових ресурсів. Основні причини ресурсних конфліктів:

- Перевантаження ключових фахівців.
- Наявність критичних завдань із однаковими термінами виконання.
- Нестача фінансових або матеріальних ресурсів у визначений період.

Методи вирішення ресурсних конфліктів:

1. Пріоритизація завдань: визначення критичних завдань та зміщення менш важливих.
2. Перепланування робіт: зміна графіку для рівномірного розподілу ресурсів.
3. Залучення додаткових ресурсів: розширення команди або оренда додаткових інфраструктурних потужностей.
4. Оптимізація завдань: зменшення обсягу робіт шляхом перегляду вимог або автоматизації.

Ефективне планування ресурсів дозволяє уникнути простоїв, забезпечити своєчасне виконання завдань та мінімізувати ризик конфліктів. Використання сучасних інструментів управління проєктами та гнучких підходів допомагає збалансувати навантаження на команду та підвищити продуктивність проєкту.

4.5 Визначення вартості проєкту та базового графіка вартості

Визначення вартості проєкту є важливим етапом планування, що дозволяє оцінити бюджет, необхідний для реалізації завдань проєкту, та визначити основні

витрати. Базовий графік вартості є основою для контролю витрат і зіставлення фактичних результатів із запланованими показниками.

1. Визначення вартості проєкту

Процес оцінки вартості передбачає аналіз ресурсів, часу та робіт, необхідних для кожного етапу проєкту. Вартість проєкту розподілена за такими категоріями:

Таблиця 4.2

Таблиця вартості проєкту

Категорія витрат	Опис	Орієнтовна сума (грн)
Людські ресурси	Оплата праці команди проєкту (розробники, аналітики, тестувальники, DevOps тощо).	2 500 000
Технологічні ресурси	Ліцензії на програмне забезпечення, хостинг, хмарні сервіси (AWS, Azure).	800 000
Матеріальні ресурси	Закупівля технічного обладнання (сервери, ПК).	500 000
Інтеграція та тестування	Витрати на інтеграцію з ERP/CRM, тестування.	600 000
Навчання та підтримка користувачів	Розробка документації, навчання персоналу.	300 000
Ризики та непередбачувані витрати	Резерв на покриття ризиків.	300 000
Загальна вартість проєкту		5 000 000

2. Базовий графік вартості

Базовий графік вартості відображає розподіл витрат по етапах життєвого циклу проєкту. Він є основою для контролю бюджету.

Таблиця 4.3

Таблиця розподілу вартості на етапах життєвого циклу прокту

Етап проєкту	Період виконання	Вартість (грн)
Ініціація проєкту	01.09.2024 15.09.2024	300 000
Планування проєкту	16.09.2024 27.09.2024	200 000
Розробка концептуальної моделі	28.09.2024 12.10.2024	400 000
Проєктування бази даних	13.10.2024 24.10.2024	600 000
Розробка програмного забезпечення	25.10.2024 10.11.2024	1 500 000
Тестування системи	11.11.2024 22.11.2024	600 000
Впровадження та інтеграція	23.11.2024 30.11.2024	800 000
Навчання користувачів	01.12.2024 05.12.2024	300 000
Фінальна оптимізація та підтримка	06.12.2024 12.12.2024	300 000

Закінчення табл. 4.3

Загальна вартість		5 000 000

Висновок:

Вартість проєкту визначена з урахуванням потреб у ресурсах, етапів робіт та можливих ризиків. Базовий графік вартості є інструментом для моніторингу фінансової дисципліни проєкту та забезпечення ефективного використання бюджету.

4.6 Аналіз ризиків проєкту. Розробка протиризикових заходів

Аналіз ризиків проєкту є важливим етапом управління проєктом, що дозволяє ідентифікувати можливі загрози, оцінити їх вплив та розробити заходи для мінімізації або усунення негативних наслідків.

1. Ідентифікація ризиків

Під час аналізу ризиків було виділено ключові загрози, які можуть виникнути у процесі реалізації проєкту зі створення та впровадження системи збору, систематизації та аналізу даних з продажу фармацевтичної продукції:

Таблиця 4.4

Таблиця ідентифікації ризиків прокту

№	Категорія ризику	Опис ризику	Вплив
1	Технічний ризик	Проблеми з інтеграцією системи з ERP/CRM	Середній
2	Операційний ризик	Низька продуктивність системи під час обробки великих обсягів даних	Високий
3	Людський фактор	Брак досвіду чи навичок у членів команди	Високий

Закінчення табл 4.4

4	Фінансовий ризик	Перевищення бюджету через неочікувані витрати	Середній
5	Організаційний ризик	Неефективне управління ресурсами та термінами	Високий
6	Безпека даних	Загроза витоку чи втрати конфіденційної інформації	Високий
7	Регуляторний ризик	Відповідність вимогам законодавства та стандартам безпеки	Середній

2. Оцінка ризиків

Для кожного ризику було проведено оцінку за двома критеріями:

- Ймовірність виникнення (висока, середня, низька).
- Вплив на проєкт (критичний, високий, середній, низький).

Таблиця 4.5

Таблиця оцінки ризиків проєкту

Ризик	Ймовірність	Вплив	Рівень ризику
Технічний ризик	Середня	Середній	Помірний
Операційний ризик	Висока	Високий	Критичний
Людський фактор	Середня	Високий	Високий

Закінчення табл 4.5

Фінансовий ризик	Середня	Середній	Помірний
Організаційний ризик	Середня	Високий	Високий
Безпека даних	Висока	Високий	Критичний
Регуляторний ризик	Низька	Середній	Низький

3. Протиризикові заходи

На основі оцінки ризиків розроблено заходи для їх мінімізації або усунення:

Таблиця 4.6

Таблиця проти ризикових заходів

Категорія ризику	Протиризикові заходи
Технічний ризик	- Використання стандартизованих API для інтеграції. - Тестування інтеграцій на ранніх етапах розробки.
Операційний ризик	- Оптимізація запитів до бази даних. - Використання хмарних рішень для масштабування системи.
Людський фактор	- Організація додаткових тренінгів для команди. - Залучення зовнішніх фахівців у критичних етапах.
Фінансовий ризик	- Резервування 10% бюджету на непередбачувані витрати. - Регулярний контроль за витратами.

Закінчення табл 4.6

Організаційний ризик	- Використання систем управління проєктами (Jira, Trello). - Регулярні зустрічі для моніторингу прогресу.
Безпека даних	- Впровадження шифрування даних (AES, RSA). - Розмежування доступу на основі ролей користувачів (RBAC).
Регуляторний ризик	- Перевірка відповідності проєкту регуляторним стандартам. - Консультації з юристами на початкових етапах.

Аналіз ризиків дозволив ідентифікувати потенційні загрози та розробити заходи для їх усунення. Впровадження протиризикових стратегій допоможе знизити ймовірність виникнення критичних ситуацій та забезпечити успішну реалізацію проєкту.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було досягнуто мети дослідження, а саме розроблено методи та моделі управління проєктом створення та впровадження системи збору, систематизації та аналізу даних з продажу фармацевтичної продукції. В результаті проведених досліджень та реалізованих рішень отримано такі основні результати:

1. Проведено аналіз методів оцінки впливів оточення на IT-проєкти (SWOT-аналіз, PESTEL-аналіз, аналіз зацікавлених сторін, п’яти сил Портера, LCA, CSF). Визначено, що для успішного управління проєктом доцільно застосовувати комплексний підхід до оцінки оточення, оскільки кожен метод забезпечує розгляд проєкту з різних кутів зору та доповнює один одного.
2. Сформульовано проблемну область проєкту, яка полягає у неефективному управлінні даними про продажі фармацевтичної продукції через фрагментацію джерел, низьку якість та відсутність автоматизації обробки даних. Виявлено ключові проблеми, включно зі складністю інтеграції з існуючими ERP та CRM системами, недоліками в прогнозуванні попиту, недостатністю аналітичної підтримки та необхідністю забезпечення інформаційної безпеки.
3. Сформульовано наукову новизну та інноваційність запропонованого проєкту.
4. Проведено огляд літературних та інформаційних джерел з теми дослідження, що дало змогу визначити актуальні підходи та інструменти для вирішення виявлених проблем, зокрема використання сучасних ІКТ та методів аналітики даних (регресійний аналіз, кластерний аналіз).
5. Сформульовано технічне завдання на розробку інформаційної системи для автоматизованого збору, систематизації та аналізу даних про продажі фармацевтичної продукції. Визначено функціональні та нефункціональні

вимоги, обґрунтовано використання реляційної СУБД PostgreSQL, а також проєктування масштабованої та надійної архітектури із застосуванням Spring Boot, React, Docker та Kubernetes.

6. Розроблено концептуальну модель інформаційної системи та формалізовано математичні моделі для аналізу даних. Запропоновані моделі лінійної регресії та кластерного аналізу дозволили автоматизувати процес прогнозування продажів, оптимізації запасів та сегментації ринку.
7. Створено концептуальну та логічну моделі бази даних проєкту з урахуванням нормалізації та вимог до безпеки. Запропоновано механізми контролю доступу, шифрування, логування дій користувачів та резервного копіювання, що забезпечує надійність, цілісність та безпеку даних.
8. Розроблено програмне забезпечення з використанням сервісно-орієнтованої архітектури та трирівневої структури (Presentation, Business Logic, Data Layer). Описано алгоритми збору, очищення, нормалізації даних, а також реалізовано аналітичні алгоритми для регресійного та кластерного аналізу. Створено інтерфейси користувача для введення, перегляду та аналізу даних, а також RESTful API для інтеграції з ERP та CRM системами.
9. Виконано планування проєкту, включаючи розробку паспорту проєкту, декомпозицію задач за методом WBS, побудову календарного плану, визначення ресурсів та оцінку вартості. Проведено аналіз ризиків та розроблено протиризикові заходи, що підвищують шанси на успішну реалізацію проєкту.
10. У результаті впровадження створеної інформаційної системи очікується підвищення ефективності управлінських рішень у фармацевтичних компаніях. Завдяки автоматизації процесів збору та аналізу даних скорочується час і ресурси на обробку інформації, підвищується точність прогнозів попиту, оптимізуються запаси та логістика, що сприяє підвищенню конкурентоспроможності та прибутковості бізнесу.

Таким чином, у кваліфікаційній роботі розв’язано поставлені завдання: від аналізу проблемної області та обґрунтування доцільності проєкту – до формалізації математичних моделей, проєктування інформаційного та програмного забезпечення, а також детального планування проєкту. Отримані результати можуть бути використані як основа для подальшого вдосконалення системи та розширення її функціональних можливостей, впровадження штучного інтелекту та інтеграції з новими джерелами даних.

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. F. Hoffmann-La Roche. (2020). Personalized healthcare through big data analytics. Journal of Pharmaceutical Research, 12(3), 156-170.
2. Thrive Bioscience. (2021). Machine learning in cell culture management. Cellular Biology and Biotechnology, 8(5), 102-114.
3. Pharmaceutical Technology. (2022). Innovations in big data usage in the pharmaceutical industry. Pharmaceutical Technology Journal, 10(4), 203-218.
4. BMC Health Services Research. (2021). The role of ICT in enhancing pharmaceutical supply chains. BMC Health Services Research, 21(12), 321-334.
5. Porter, M. E. (1979). How competitive forces shape strategy. Harvard Business Review, 57(2), 137-145.
6. SWOT Analysis. (n.d.). In Business Dictionary.
7. PESTEL Analysis. (n.d.). In Business Dictionary.
8. LaValle, S., Lesser, E., Shockley, R., Hopkins, M. S., & Kruschwitz, N. (2011). Big data, analytics and the path from insights to value. MIT Sloan Management Review, 52(2), 21-32.
9. McAfee, A., & Brynjolfsson, E. (2012). Big data: The management revolution. Harvard Business Review, 90(10), 60-68.
10. Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., & Hung Byers, A. (2011). Big data: The next frontier for innovation, competition, and productivity. McKinsey Global Institute.
11. Davenport, T. H., & Harris, J. G. (2007). Competing on analytics: The new science of winning. Harvard Business School Press.
12. Provost, F., & Fawcett, T. (2013). Data science for business: What you need to know about data mining and data-analytic thinking. O'Reilly Media, Inc.
13. Chen, H., Chiang, R. H. L., & Storey, V. C. (2012). Business intelligence and analytics: From big data to big impact. MIS Quarterly, 36(4), 1165-1188.
14. Kudyba, S., & Hoptroff, R. (2001). Data mining and business intelligence: A

guide to productivity. Information Systems Management, 18(4), 56-59.

15. Watson, H. J., & Wixom, B. H. (2007). The current state of business intelligence. Computer, 40(9), 96-99.
16. The Global Use of ICT in Healthcare (2019). Global Health Review, 5(2), 45-62.
17. International Data Corporation (IDC). (2020). Worldwide big data and business analytics forecast, 2019-2023.
18. World Health Organization (WHO). (2021). Global strategy on digital health 2020-2025. WHO Publication.
19. Forbes Insights. (2019). Data-driven and digital: 6 strategies for data-driven digital transformation.
20. Accenture. (2020). The big data revolution in healthcare.
21. Russom, P. (2011). Big Data Analytics. *TDWI Best Practices Report*. [Electronic resource]: https://tdwi.org/research
22. Agrawal, D., Das, S., & El Abbadi, A. (2012). Big Data and Cloud Computing: Current State and Future Opportunities. *Proceedings of the 14th International Conference on Extending Database Technology*. [Electronic resource]: https://dl.acm.org/doi/10.1145/2247596.2247645
23. Patel, P., & Savani, G. (2020). The Role of AI and ML in Pharmaceutical Industry. *Future Internet*, 12(9), 157. DOI: 10.3390/fi12090157
24. Han, J., Kamber, M., & Pei, J. (2011). *Data Mining: Concepts and Techniques*. Morgan Kaufmann.
25. Taylor, J. (2022). Leveraging Big Data to Transform the Pharmaceutical Supply Chain. *Healthcare Journal of Analytics*, 15(3), 58-72.
26. Mertens, W., & Faisal, S. (2021). Blockchain in Pharmaceutical Supply Chain: A Review. *Journal of Digital Innovation*, 5(2), 81-96.
27. Chen, M., Mao, S., & Liu, Y. (2014). Big Data: A Survey. *Mobile Networks and Applications*, 19(2), 171-209. DOI: 10.1007/s11036-013-0489-0
28. Kambatla, K., Kollias, G., Kumar, V., & Grama, A. (2014). Trends in Big Data

Analytics. *Journal of Parallel and Distributed Computing*, 74(7), 2561-2573.

29. World Economic Forum. (2021). Digital Transformation in Healthcare: Unlocking Big Data. [Electronic resource]: https://www.weforum.org/reports
30. Chaudhuri, S., Dayal, U., & Narasayya, V. (2011). An Overview of Business Intelligence Technology. *Communications of the ACM*, 54(8), 88-98.
31. Google Cloud. (2022). Machine Learning for Healthcare Analytics. [Electronic resource]: https://cloud.google.com/solutions/healthcare
32. KPMG. (2019). The Future of Healthcare: Data-Driven Solutions. [Electronic resource]: https://home.kpmg/xx/en/home/industries/healthcare
33. IBM. (2020). How AI is Revolutionizing the Pharmaceutical Industry. [Electronic resource]: https://www.ibm.com/industries/life-sciences
34. Redman, T. C. (2018). The Impact of Poor Data Quality in the Pharmaceutical Industry. *Harvard Business Review*. [Electronic resource]: https://hbr.org/2018/09
35. Brynjolfsson, E., & McAfee, A. (2014). *The Second Machine Age: Work, Progress, and Prosperity in a Time of Brilliant Technologies*. W. W. Norton & Company.
36. Databricks. (2022). Real-Time Data Analytics in Pharma. [Electronic resource]: https://databricks.com/solutions
37. Snowflake. (2021). Improving Pharma Supply Chain Analytics. [Electronic resource]: https://www.snowflake.com/healthcare
38. The Economist Intelligence Unit. (2020). Transforming Healthcare Through Data. [Electronic resource]: https://www.eiu.com/healthcare
39. Accenture. (2019). AI in Pharma: Improving Patient Outcomes. [Electronic resource]: https://www.accenture.com
40. Merck & Co. (2020). Utilizing Data Analytics for Supply Chain Optimization. *Merck Insights*. [Electronic resource]: https://www.merck.com/insights
41. United Nations. (2021). Digital Health and the SDGs. [Electronic resource]: https://sdgs.un.org

42. Harvard Health. (2022). Big Data in Medicine: Opportunities and Challenges. [Electronic resource]: https://www.health.harvard.edu
43. Boulton, C. (2020). Why Big Data Matters in Pharmaceuticals. *CIO Journal*. [Electronic resource]: https://www.cio.com/article
44. Singhal, S. (2022). Machine Learning for Predictive Analytics in Pharma. *Healthcare Data Review*, 14(7), 101-118.
45. Liu, Y., & Zhang, X. (2019). Real-Time Data Processing for Healthcare Systems. *Journal of Health Informatics*, 27(2), 32-41.

ДОДАТКИ

Додаток А

Підготовки даних для кластерного аналізу в JAVA

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;

public class SalesData {
    public int saleId;
    public java.sql.Date saleDate;
    public int productId;
    public int quantity;
    public double price;

    public SalesData(int saleId, java.sql.Date saleDate, int productId, int quantity, double price) {
        this.saleId = saleId;
        this.saleDate = saleDate;
        this.productId = productId;
        this.quantity = quantity;
        this.price = price;
    }
}

public class DataPreparation {
    public static List<SalesData> fetchData() {
        List<SalesData> salesDataList = new ArrayList<>();
        try {
            Connection connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/pharmacy_sales_db", "username", "password");
            Statement statement = connection.createStatement();
            ResultSet resultSet = statement.executeQuery("SELECT SaleID, SaleDate, ProductID, Quantity, Price
FROM Sale");

            while (resultSet.next()) {
                int saleId = resultSet.getInt("SaleID");
                java.sql.Date saleDate = resultSet.getDate("SaleDate");
                int productId = resultSet.getInt("ProductID");
                int quantity = resultSet.getInt("Quantity");
                double price = resultSet.getDouble("Price");
                salesDataList.add(new SalesData(saleId, saleDate, productId, quantity, price));
            }
            connection.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
        return salesDataList;
    }
}
```

Додаток Б

Використання алгоритму k - середніх для кластерного аналізу в JAVA

```
import org.apache.commons.math3.ml.clustering.Cluster;
import org.apache.commons.math3.ml.clustering.Clusterable;
import org.apache.commons.math3.ml.clustering.KMeansPlusPlusClusterer;
import org.apache.commons.math3.ml.clustering.DoublePoint;
import java.util.ArrayList;
import java.util.List;


public class KMeansClustering {
    public static void main(String[] args) {
        List<SalesData> salesDataList = DataPreparation.fetchData();

        // Підготовка даних для кластеризації
        List<Clusterable> points = new ArrayList<>();
        for (SalesData data : salesDataList) {
            points.add(new DoublePoint(new double[]{data.quantity, data.price}));
        }

        // Виконання кластеризації
        int k = 4; // Приклад, вибираємо 4 кластери
        KMeansPlusPlusClusterer<Clusterable> clusterer = new KMeansPlusPlusClusterer<>(k);
        List<Cluster<Clusterable>> clusterResults = clusterer.cluster(points);

        // Виведення результатів кластеризації
        for (int i = 0; i < clusterResults.size(); i++) {
            System.out.println("Cluster " + i + ":");
            for (Clusterable point : clusterResults.get(i).getPoints()) {
                double[] pointData = point.getPoint();
                System.out.println("Quantity: " + pointData[0] + ", Price: " + pointData[1]);
            }
        }
    }
}
```

Додаток В

Використання симетричного AES шифрування на сервісному рівні серверної архітектури в JAVA

```java
package com.example.demo.service;

import org.springframework.stereotype.Service;

import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.GCMParameterSpec;
import java.security.SecureRandom;
import java.util.Base64;

@Service
public class EncryptionService implements IEncryptionService {
    private static final String ALGORITHM = "AES";
    private static final int KEY_SIZE = 128;
    private static final int T_LEN = 128;
    private SecretKey key;
    private byte[] iv;

    public EncryptionService() throws Exception {
        KeyGenerator keyGen = KeyGenerator.getInstance(ALGORITHM);
        keyGen.init(KEY_SIZE);
        key = keyGen.generateKey();
        iv = new byte[12];
        SecureRandom random = new SecureRandom();
        random.nextBytes(iv);
    }

    @Override
    public String encrypt(String data) throws Exception {
        Cipher cipher = Cipher.getInstance("AES/GCM/NoPadding");
        GCMParameterSpec spec = new GCMParameterSpec(T_LEN, iv);
        cipher.init(Cipher.ENCRYPT_MODE, key, spec);
        byte[] encrypted = cipher.doFinal(data.getBytes());
        return Base64.getEncoder().encodeToString(encrypted);
    }
    @Override
    public String decrypt(String encryptedData) throws Exception {
        Cipher cipher = Cipher.getInstance("AES/GCM/NoPadding");
        GCMParameterSpec spec = new GCMParameterSpec(T_LEN, iv);
        cipher.init(Cipher.DECRYPT_MODE, key, spec);
        byte[] decodedData = Base64.getDecoder().decode(encryptedData);
        byte[] decrypted = cipher.doFinal(decodedData);
        return new String(decrypted);
    }
}
```

Додаток Г

Фрагменти коду по візуалізації, фільтрації та пошуку даних продаж

index.html > html

```html
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Візуалізація продажів продуктів</title>
    <!-- Підключення CSS -->
    <link rel="stylesheet" href="style.css">
    <!-- Підключення Chart.js -->
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <!-- Навігація -->
    <nav>
        <a href="#">Головна</a>
        <a href="#">Статистика</a>
        <a href="#">Про нас</a>
        <a href="#">Контакти</a>
        <button class="btn">Увійти</button>
        <button class="btn">Реєстрація</button>
    </nav>

    <!-- Основний блок -->
    <div class="container">
        <h2>Візуалізація продажів продуктів</h2>
        <canvas id="salesChart"></canvas>

        <!-- Фільтрація -->
        <div class="filter-container">
            <label>Продукт: <input type="text" id="productFilter"></label>
            <label>Аптека: <input type="text" id="pharmacyFilter"></label>
            <label>Дата продажу: <input type="date" id="dateFilter"></label>
            <button class="btn" onclick="filterData()">Фільтрувати</button>
        </div>

        <!-- Таблиця даних -->
        <h3>Детальна інформація про продажі</h3>
        <table>
            <thead>
                <tr>
                    <th>Продукт</th>
                    <th>Кількість</th>
                    <th>Ціна</th>
                    <th>Дата продажу</th>
                    <th>Аптека</th>
                    <th>Дистриб'ютор</th>
                </tr>
            </thead>
            <tbody id="salesData">
                <tr><td>Product A</td><td>10</td><td>12</td><td>2024-05-01</td><td>Pharmacy 1</td><td>Pfizer</td></tr>
                <tr><td>Product B</td><td>8</td><td>8</td><td>2024-05-02</td><td>Pharmacy 2</td><td>Roche</td></tr>
                <tr><td>Product C</td><td>7</td><td>15</td><td>2024-05-03</td><td>Pharmacy 3</td><td>Novartis</td></tr>
                <tr><td>Product D</td><td>5</td><td>20</td><td>2024-05-04</td><td>Pharmacy 3</td><td>Novartis</td></tr>
                <tr><td>Product E</td><td>9</td><td>9</td><td>2024-05-05</td><td>Pharmacy 2</td><td>AstraZeneca</td></tr>
            </tbody>
        </table>
    </div>

    <!-- Підключення JavaScript -->
    <script src="script.js"></script>
</body>
</html>
```

Рис. Г.1 Фрагмент HTML по візуалізації, фільтрації та пошуку даних продаж

```css
body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
  background-color: #f9f9f9;
}

nav {
  background-color: #003366;
  padding: 10px;
  display: flex;
  justify-content: space-around;
  color: white;
}

nav a {
  color: white;
  text-decoration: none;
  margin: 0 10px;
}

.btn {
  background-color: #1abc9c;
  color: white;
  border: none;
  padding: 8px 15px;
  cursor: pointer;
  border-radius: 5px;
}

.container {
  margin: 20px;
  padding: 20px;
  background: white;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
  border-radius: 8px;
}

h2 {
  text-align: center;
}

table {
  width: 100%;
  border-collapse: collapse;
}

th, td {
  padding: 8px;
  border: 1px solid #ddd;
  text-align: center;
}

th {
  background-color: #f2f2f2;
}

.filter-container {
  display: flex;
  justify-content: space-between;
  margin-bottom: 20px;
}

input, button {
  padding: 5px;
}
```

Рис. Г.2 Фрагмент CSS по візуалізації, фільтрації та пошуку даних продаж

```
JS sctipt.js > ...
// Дані для графіку
const chartData = {
    labels: ['Product A', 'Product B', 'Product C', 'Product D', 'Product E'],
    datasets: [{
        label: 'Кількість',
        data: [10, 8, 7, 5, 9], // Дані для графіку
        backgroundColor: 'teal'
    }]
};

// Створення графіку
const salesChart = new Chart(document.getElementById('salesChart'), {
    type: 'bar',
    data: chartData,
    options: {
        responsive: true,
        scales: {
            y: {
                beginAtZero: true
            }
        }
    }
});

// Фільтрація таблиці
function filterData() {
    const product = document.getElementById('productFilter').value.toLowerCase();
    const pharmacy = document.getElementById('pharmacyFilter').value.toLowerCase();
    const date = document.getElementById('dateFilter').value;

    const rows = document.querySelectorAll('#salesData tr');
    rows.forEach(row => {
        const cells = row.getElementsByTagName('td');
        const productName = cells[0].innerText.toLowerCase();
        const pharmacyName = cells[4].innerText.toLowerCase();
        const saleDate = cells[3].innerText;

        row.style.display =
            (product === '' || productName.includes(product)) &&
            (pharmacy === '' || pharmacyName.includes(pharmacy)) &&
            (date === '' || saleDate === date) ? '' : 'none';
    });
}
```

Рис. Г.3 Фрагмент JavaScript по візуалізації, фільтрації та пошуку даних продаж

Додаток Д

Використання симетричного AES шифрування на сервісному рівні серверної архітектури в JAVA

```
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.*;
import java.util.stream.Collectors;

public class DataCleaner {

    // Клас для представлення продуктів
    static class Product {
        int productID;
        String productName;
        String manufacturerName;
        String pharmacyName;

        public Product(int productID, String productName, String manufacturerName,
String pharmacyName) {
            this.productID = productID;
            this.productName = productName.trim().toUpperCase(); // Нормалізація назв
            this.manufacturerName = manufacturerName.trim().toUpperCase();
            this.pharmacyName = pharmacyName.trim().toUpperCase();
        }

        // Метод для генерації хешу для виявлення дублікатів
        public String generateHash() {
            try {
                String data = productID + productName + manufacturerName + pharmacyName;
                MessageDigest md = MessageDigest.getInstance("SHA-256");
                byte[] hashBytes = md.digest(data.getBytes());
                StringBuilder hash = new StringBuilder();
                for (byte b : hashBytes) {
                    hash.append(String.format("%02x", b));
                }
                return hash.toString();
            } catch (NoSuchAlgorithmException e) {
                throw new RuntimeException("Error generating hash", e);
            }
        }

        @Override
        public String toString() {
            return "Product{" +
                    "productID=" + productID +
                    ", productName='" + productName + '\'' +
                    ", manufacturerName='" + manufacturerName + '\'' +
```

```
                    ", pharmacyName='" + pharmacyName + '\'' +
                    '}';
        }
    }

    // Метод для виявлення та усунення дублікатів
    public static List<Product> removeDuplicates(List<Product> productList) {
        Map<String, Product> uniqueProducts = new HashMap<>();
        for (Product product : productList) {
            String hash = product.generateHash();
            uniqueProducts.putIfAbsent(hash, product); // Якщо хеш уже існує, дубль
пропускається
        }
        return new ArrayList<>(uniqueProducts.values());
    }

    // Метод для валідації даних
    public static boolean validateProduct(Product product) {
        return product.productID > 0 && product.productName != null &&
!product.productName.isEmpty();
    }

    public static void main(String[] args) {
        // Приклад даних
        List<Product> products = Arrays.asList(
                new Product(1, " Product A ", "Pfizer", "Pharmacy 1"),
                new Product(1, "Product A", "Pfizer", "Pharmacy 1"), // Дублікат
                new Product(2, "Product B", " Roche ", "Pharmacy 2"),
                new Product(3, "Product C", "Novartis", "Pharmacy 3"),
                new Product(3, "Product C", "Novartis", "Pharmacy 3"), // Дублікат
                new Product(4, "Product D", "Novartis", "Pharmacy 3"),
                new Product(-1, "Invalid Product", "Unknown", "Unknown") // Некоректний
запис
        );

        System.out.println("=== Вхідні дані ===");
        products.forEach(System.out::println);

        // Крок 1: Нормалізація та валідація
        List<Product> validProducts = products.stream()
                .filter(DataCleaner::validateProduct) // Фільтруємо некоректні дані
                .collect(Collectors.toList());

        System.out.println("\n=== Після валідації та нормалізації ===");
        validProducts.forEach(System.out::println);

        // Крок 2: Видалення дублікатів
        List<Product> uniqueProducts = removeDuplicates(validProducts);
```

```
        System.out.println("\n=== Унікальні дані ===");
        uniqueProducts.forEach(System.out::println);
    }
}
```