Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Системотехніки
(повна назва)

**КВАЛІФІКАЦІЙНА РОБОТА**
**Пояснювальна записка**

рівень вищої освіти другий (магістерський)

Розробка методів CRM на базі генерації асоціативних правил та їх використання в інформаційній системі бібліотеки
(тема)

Виконав:
студент 2 курсу, групи *ІТПм–24–1*
*Забийворота М.А.*
(група, прізвище, ініціали)
Спеціальність *122 Комп'ютерні науки*
(код і назва спеціальності)
Освітня програма *Інформаційні технології проектування*
(повна назва освітньої програми)
Керівник доцент *Тітов С.В.*
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ *проф. Гребеннік І.В.*
(підпис) (прізвище, ініціали)

2025 р

Харківський національний університет радіоелектроніки

Факультет *Комп'ютерних наук*
Кафедра *Системотехніки*
Рівень вищої освіти *другий (магістерський)*
Спеціальність *122 Комп'ютерні науки*
(код і повна назва)
Тип програми *освітньо-професійна*
(освітньо-професійна або освітньо-наукова)
Освітня програма *Інформаційні технології проектування*
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____»_____2025 р.

**ЗАВДАННЯ**
НА КВАЛІФІКАЦІЙНУ РОБОТУ
студентові Забийворота Михайло Андрійович
(прізвище, ім'я, по батькові)

1. Тема роботи: Розробка методів CRM на базі генерації асоціативних правил та їх використання в інформаційній системі бібліотеки
затверджена наказом по університету від "24" листопада 2025 р. № 1058Ст

2. Термін подання студентом роботи до екзаменаційної комісії 16.12.2025
3. Вихідні дані до роботи розробити методи CRM на базі генерації асоціативних правил в інформаційній системі бібліотеки
4. Перелік питань, що потрібно опрацювати в роботі *1 Аналіз предметної області, 1.1 Аналіз предметної області, яка визначає діяльність бібліотеки, 1.2 Аналіз існуючих систем, 1.3 Визначення сфери застосування системи, 1.4 Постановка задачі 2 Дослідження асоціативних алгоритмів, 2.1 Загальний огляд 2.2 Дослідження Apriori, 2.3 Дослідження FP-Growth 2.4 Дослідження Eclat 3 Проектування системи 3.1 Розробка функціональних вимог до системи 3.2 Розробка моделі потоків даних системи, 3.3 Опис архітектури розробленої системи 3.4 Розробка діаграми варіантів використання системи, 3.5 Розробка діаграми послідовності, 3.6 Розробка ієрархії сторінок 4. Розробка системи, 4.1 Розробка логічної та фізичної моделі, 4.2 Створення бази даних на платформі СУБД, 4.3 Розробка системних вимог до системи, 4.4 Розробка уявлень та sql запитів серверної частини системи, 4.5 Розробка інтерфейсу для функцій інформаційної системи, 4.6 Реалізація FP-Growth в інформаційній системі.*

**КАЛЕНДАРНИЙ ПЛАН**

| № | Назва етапів роботи | Терміни виконання етапів | Примітка |
|---|---|---|---|
| 1 | Аналіз предметної області, яка визначає діяльність бібліотеки | 08.10.25 – 11.10.25 | виконано |
| 2 | Аналіз існуючих систем | 12.10.25 – 15.10.25 | виконано |
| 3 | Визначення сфери застосування системи | 16.10.25 – 17.10.25 | виконано |
| 4 | Постановка задачі | 18.10.25 – 19.10.25 | виконано |
| 5 | Дослідження асоціативних алгоритмів | 20.10.25 – 23.10.25 | виконано |
| 6 | Загальний огляд | 24.10.25 – 25.10.25 | виконано |
| 7 | Дослідження Apriori | 26.10.25 – 27.10.25 | виконано |
| 8 | Дослідження FP-Growth | 28.10.25 – 29.10.25 | виконано |
| 9 | Дослідження Eclat | 30.10.25 – 31.10.25 | виконано |
| 10 | Проектування системи | 01.11.25 – 05.11.25 | виконано |
| 11 | Розробка функціональних вимог до системи | 06.11.25 – 08.11.25 | виконано |
| 12 | Розробка моделі потоків даних системи | 09.11.25 – 10.11.25 | виконано |
| 13 | Опис архітектури розробленої системи | 11.11.25 – 12.11.25 | виконано |
| 14 | Розробка діаграми варіантів використання системи | 13.11.25 – 15.11.25 | виконано |
| 15 | Розробка діаграми послідовності | 16.11.25 – 18.11.25 | виконано |
| 16 | Розробка ієрархії сторінок | 19.11.25 – 20.11.25 | виконано |
| 17 | Розробка системи | 21.11.25 – 05.12.25 | виконано |
| 18 | Розробка логічної та фізичної моделі | 21.11.25 – 05.12.25 | виконано |
| 19 | Створення бази даних на платформі СУБД | 21.11.25 – 05.12.25 | виконано |
| 20 | Розробка системних вимог до системи | 21.11.25 – 05.12.25 | виконано |
| 21 | Розробка уявлень та sql запитів серверної частини системи | 21.11.25 – 05.12.25 | виконано |
| 22 | Розробка інтерфейсу для функцій інформаційної системи | 06.12.25 – 10.12.25 | виконано |
| 23 | Реалізація FP-Growth в інформаційній системі | 06.12.25 – 11.12.25 | виконано |
| 24 | Оформлення пояснювальної записки | 12.12.25 – 12.12.25 | виконано |
| 25 | Подання роботи до ЕК | 16.12.2025 | виконано |

Дата видачі завдання 08 жовтня 2025 р.

Студент _____ Забийворота М.А
(підпис) (посада, прізвище, ініціали)

Керівник роботи _____ Доц. Тітов С.В
(підпис) (посада, прізвище, ініціали)

# РЕФЕРАТ

Пояснювальна записка містить: 69 сторінок, 51 рисунків, 1 таблиця, 2 1 джерело.

E-Business, IDEF0, IDEF1X, DATA FLOW DIAGRAM, API, МОДЕЛЬ ДАНИХ, БАЗА ДАНИХ, АСОЦІАТИВНІ ПРАВИЛА, APRIORI, ECLAT, FPGROWTH, DATA MINING

Об’єктом досліджень кваліфікаційної роботи є процес взаємодії з читачами інформаційної системи бібліотеки.

Предметом досліджень кваліфікаційної роботи є методи аналізу даних та алгоритми генерації асоціативних правил для вдосконалення функціональності CRM-системи бібліотеки.

Мета досліджень: розробка та впровадження методу формування рекомендацій на основі асоціативних алгоритмів у CRM-компонент інформаційної системи бібліотеки.

Методи дослідження – системний підхід, методи структурного аналізу і моделювання реляційних баз даних, методи пошуку асоціативних правил та порівняння їх ефективності.

Підсумком роботи є розроблений та інтегрований в інформаційну систему бібліотеки програмний модуль для генерації асоціативних правил. Отримані результати дозволяють автоматизувати процес виявлення прихованих закономірностей у поведінці користувачів та формувати рекомендації.

# ABSTRACT

The report on the certification work contains: 69 pages, 51 figures, 1 table, 21 references.

E-Business, IDEF0, IDEF1X, DATA FLOW DIAGRAM, API, МОДЕЛЬ ДАНИХ, БАЗА ДАНИХ, АСОЦІАТИВНІ ПРАВИЛА, APRIORI, ECLAT, FPGROWTH, DATA MINING

The object of research in the qualification work is the process of interaction with the readers of the library information system.

The subject of the qualification work research is data analysis methods and algorithms for generating association rules to improve the functionality of the library's CRM system.

The purpose of the research is the development and implementation of a recommendation generation method based on associative algorithms in the CRM component of the library information system.

Research methods include a systematic approach, methods of structural analysis and modeling of relational databases, methods for searching association rules, and comparing their effectiveness.

The result of the work is a software module for generating association rules, designed and integrated into the library information system. The obtained results allow for automating the process of detecting hidden patterns in user behavior and generating recommendations.

# ЗМІСТ

# ВСТУП

У світі, де доступ до інформації стає все більш безпосереднім та невід'ємним від нашого щоденного життя, роль бібліотек стає набагато важливішою. Старі добрі книжки та періодичні видання залишаються невід'ємною частиною нашої культури, проте разом із тим з'являються нові вимоги та потреби. Сьогоднішній користувач бібліотеки шукає не лише книги, а й доступ до електронних ресурсів, аудіокниг, відеоматеріалів та онлайн-курсів.

У такому контексті з'являється необхідність у модернізації бібліотечного середовища та у створенні ефективних інструментів для зручного доступу до різноманітної інформації. І саме тут на сцену виходить концепція е-бібліотек.

Запровадження е-бібліотек відкриває нові можливості для користувачів, дозволяючи зручно та швидко знаходити потрібні матеріали, незалежно від їх формату та виду. Вони стають не лише місцем зберігання книг, а й центром обміну знаннями та інформацією, де кожен може знайти щось корисне для себе.

З розвитком електронних бібліотек та онлайн сервісів зростає потреба у системах, які можуть враховувати індивідуальні інтереси читачів, прогнозувати їхні інформаційні потреби та підвищувати рівень задоволеності від взаємодії з системою.

Проблема полягає в тому, що більшість бібліотечних систем обмежуються лише функціями обліку та доступу, не використовуючи зібрані дані для покращення якості обслуговування. Як запропонувати читачеві таку книгу, яка його зацікавить?

Для досягнення цього застосовуються методи інтелектуального аналізу, зокрема алгоритми генерації асоціативних правил.

Метою мого дослідження є аналіз існуючих рекомендаційних алгоритмів

заснованих на пошуку асоціативних правил, вибір найкращого для предметної області бібліотеки, проектування й створення ефективної е-бібліотеки, яка відповідає потребам сучасного користувача та сприяє розвитку читацької культури та доступу до знань.

За допомогою імплементації асоціативних алгоритмів в інформаційну систему я прагну створити зручне та інноваційне середовище для всіх шукачів знань, де кожен зможе знайти відповідь на свої запитання та відкрити для себе нові горизонти знань і відкриттів.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

## 1.1 Аналіз предметної області, яка визначає діяльність бібліотеки

Бібліотека це установа, яка зберігає та надає доступ до різноманітних друкованих інформаційних ресурсів, таких як книги, журнали, періодичні видання і багато іншого. Предметна область бібліотеки включає в себе різні типи інформації та матеріалів, що можуть бути корисними для великої кількості користувачів.

Замовником бібліотеки може бути організація або установа, яка фінансує та визначає вимоги до створення або покращення доступу до різноманітних інформаційних ресурсів для власних користувачів. Наприклад, школа чи громадська організація.

Для провадження діяльності бібліотеки ведеться паперовий облік книжок які зберігаються та поступають до неї. Облік ведеться за каталогами для того, щоб було зручно шукати ці книги. Про кожну книгу зберігаються такі дані: назва; автор; рік видання; мова; жанр; тип обкладинки; видавництво; ISBN; кількість сторінок; кількість екземплярів.

Також ведеться облік читачів. Їх особиста інформація бронювання книжок. Про кожного користувача зберігається така інформація: ПІБ; Email; номер телефону; історія бронювання книжок.

Web-застосунок для бібліотеки зможе виконувати функцію зберігання та надання доступу до списку доступних книг, журналів, газет тощо. Дасть змогу позичати матеріали.

Система має автоматично обробляти запити користувачів, виконувати резервації та надавати інші послуги, що полегшує взаємодію з читачами.

Бібліотека веде свою діяльність за такими нормативними документами:

- закон України «Про бібліотеки та бібліотечну справу»[1];

- наказ Міністерства культури України «Про затвердження типових правил користування бібліотеками в Україні»[2];
- Закон України «Про авторське право і суміжні права».

Бібліотека має виконувати такі задачі:

- надання інформації про наявні книги;
- облік книжок та читачів;
- бронювання книжок.

Необхідно розробити інформаційну систему для того, щоб позбавитися паперового обліку та автоматизувати процеси та бізнес-функції бібліотеки.

Додатково, актуальним є впровадження CRM, що дозволяють аналізувати поведінку користувачів і будувати ефективні стратегії взаємодії з ними. Така підсистема може використовувати методи інтелектуального аналізу даних, зокрема пошук асоціативних правил, для виявлення прихованих закономірностей у читацьких уподобаннях. Це дає змогу створювати рекомендації.

## 1.2 Аналіз існуючих систем

Аналіз систем які дозволяють автоматизувати роботу бібліотеки може допомогти визначити їхні недоліки та виправити їх у своїй. А також виявити їх переваги та впровадити в свою розробку.

Для прикладу розглянемо електронну бібліотеку «Koha». Рисунок з логотипом «Koha» зображено на рисунку 1.1. Ця платформа надає можливість для ведення систематизованого обліку всіх видань.

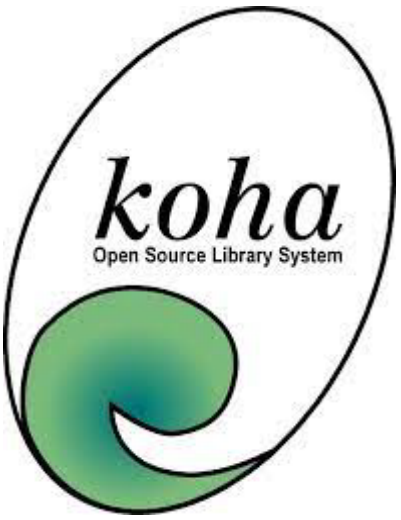



Рисунок 1.1 – Логотип koha

Вона дозволяє каталогізувати книги. Дана бібліотека має такий функціонал:

- каталогізація книг. Система дає можливість поповнювати бібліотечний фонд, додаючи інформацію про нові надходження;
- облік, тобто відстеження видачі та повернення книг. Це одна з найважливіших функцій в інформаційній системі бібліотеки. Вона дозволяє контролювати хто і коли взяв книгу;
- пошук книг за автором чи назвою. Це зручний інструмент, який допомагає користувачам знаходити потрібні їм книги;
- доступ через веб-інтерфейс. Можливість замовлення онлайн. Завдяки цьому користувач має доступ до бібліотеки 24/7, адже не потрібно йти до бібліотеки особисто. Веб-інтерфейс зображено на рисунку 1.1
- реєстрація та авторизація. Дозволяє користувачам створити обліковий запис у системі.

Проте ця платформа має суттєвий недолік. Стандартний функціонал «Koha» та більшості подібних систем орієнтований тільки на облік та пошук, а не на аналіз прихованих закономірностей, завдяки накопиченим даним. Весь масив даних про транзакції ніяк не використовується для покращення користувацького досвіду.

Інтеграція в систему рекомендаційних функцій, що базуються на поведінці користувачів, допоможе читачам обирати нові книги для подальшого читання, які з високою ймовірністю відповідатимуть їхнім реальним інтересам.

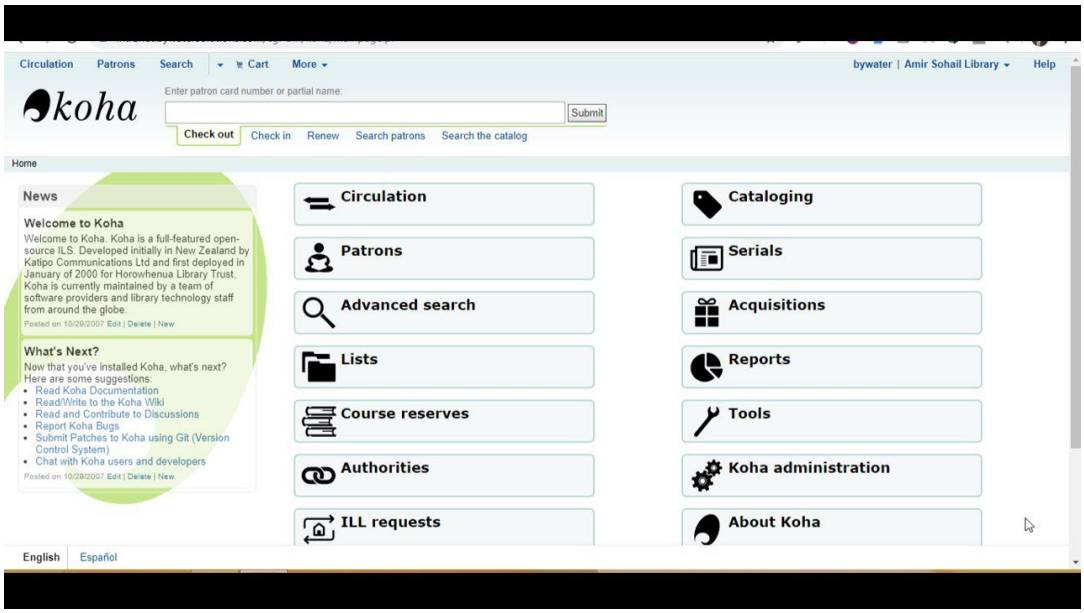



Рисунок 1.2 – Інтерфейс «Koha»

## 1.3 Визначення сфери застосування розроблювальної системи

Для інформаційної системи бібліотеки обрано систему E-Business. Це система завдяки якій можна вести справи та взаємодіяти з клієнтами через інтернет.

Проте, на відміну від E-Commerce, в цій системі може не бути комерційної складової.

Два головних компоненти інформаційної системи – це база даних MySql, де зберігається інформація, та інтерфейс, що дозволяє користувачам з нею працювати.

В інформаційній системі бібліотеки є три типи користувачів.

Незареєстрований користувач - це особа, яка ще не створила обліковий запис у системі бібліотеки. Такий користувач має обмежений доступ до функціоналу системи, він має зареєструватися перед для того, щоб

користуватися бібліотекою.

Зареєстрований користувач - це особа, яка успішно пройшла процес реєстрації у системі бібліотеки. Він має персональний обліковий запис, може взаємодіяти з розширеним функціоналом, таким як бронювання книг та перегляд каталогу книг, отримання персоналізованих рекомендацій.

Адміністратор - це користувач бібліотеки з особливими повноваженнями, які дозволяють йому управляти системою бібліотеки. Тобто він має більше прав ніж звичайний користувач. Адміністратор може додавати та видаляти книги, керувати обліком користувачів, перевіряти звіти та виконувати інші адміністративні функції.

До бізнес-функцій системи входить:

1) Реєстрація та авторизація користувача. Бізнес-процес включає реєстрацію нових користувачів у системі та їх подальшу авторизацію. Реєстрація передбачає збір необхідної інформації для створення облікового запису, в той час як авторизація дозволяє вже зареєстрованим користувачам отримувати доступ до персоналізованого облікового запису;

2) забезпечення функції роботи з каталогом книг:

а) перегляд каталогу книг. Користувачеві надається можливість переглядати всі наявні книги у системі, ознайомлюючись з їх загальною інформацією;

б) пошук книг за параметром. Користувач може використовувати параметри пошуку (наприклад, мова) для точного вибору книги в каталозі або використати функцію «випадкова книга»;

в) перегляд інформації про бронювання книг. Можливість користувача переглядати статус своїх бронювання включаючи деталі та терміни;

г) бронювання. Користувач може здійснити бронювання книги для попереднього відведення її для себе на певний термін. Для цього йому потрібно вказати дату на яку він хоче забронювати книгу.

ґ) отримання рекомендацій. Отримання персональних рекомендацій за

допомогою алгоритму, який аналізує базу даних та виявляє закономірності у виборі літератури.

3) адміністрування інформаційної системи бібліотеки:

а) облік читачів. Включає в себе перегляд всіх зареєстрованих в системі читачів та їх ролей;

б) облік книг. Система ведення обліку книг, їх кількості, наявності та звітність щодо руху книг в бібліотеці, включаючи видачу та повернення;

г) адміністратор може додати нову книгу, видалити книгу або редагувати вже наявні книги в бібліотеці.

Автоматизація дозволяє ефективно керувати колекцією, обліком книг та підтримкою бази даних, що спрощує рутинні завдання.

Онлайн інтерфейс забезпечує користувачам легкий доступ до різноманітних ресурсів, забезпечуючи високий рівень зручності та швидкості.

Віртуальні системи бібліотеки можуть автоматично обробляти запитання користувачів, виконувати резервації та надавати інші послуги, що полегшує взаємодію з читачами.

Отже, до головних задач ІС бібліотеки входить:

- Збільшення доступності інформації. Онлайн бібліотеки дозволяють користувачам отримувати доступ до різноманітної інформації.
- Автоматизація обробки запитів та резервацій. Використання автоматизованих систем сприяє ефективній обробці великого обсягу запитів та резервацій, що полегшує роботу бібліотеки та дозволяє швидко задовольняти потреби користувачів.

## 1.4 Постановка задачі

Через стрімкий розвиток Web-технологій зросли технічні можливості для реалізації будь-якого проекту. З’явилась можливість полегшення доступу до книжкового фонду та автоматизація процесу обслуговування користувачів.

Головною метою роботи є підвищення ефективності взаємодії з читачами шляхом розробки та імплементації в інформаційну систему бібліотеки програмного модуля генерації асоціативних правил.

Для досягнення цієї мети необхідно вирішити наступні конкретні задачі:

- Дослідити існуючи методи та алгоритми Data Mining для пошуку асоціативних правил, обґрунтувати вибір алгоритму для предметної області бібліотеки.
- Розробити програмний модуль, що реалізує обраний алгоритм.

# 2 ДОСЛІДЖЕННЯ АСОЦІАТИВНИХ АЛГОРИТМІВ

## 2.1 Загальний огляд

Людська свідомість природньо налаштована на пошук зв’язків між різними спостереженнями та ідеями, що ґрунтуються на асоціаціях. Проте, ми часто припускаємося помилки, інтуїтивно плутаючи просту асоціацію з глибоким причинно-наслідковим зв’язком. У машинному навчанні, зокрема в навчанні асоціативних правил, підхід значно суворіший. Спершу розглянемо, що це таке і чому воно не обов’язково означає наявність причинності.

Коли ми заходимо до супермаркету, є причина, чому товар «А» розташовано поруч з товаром «В» та іншими товарами «до А». Таке розміщення товарів – результат аналізу покупок минулих клієнтів. Його мета – зробити пошук потрібних товарів зручним і водночас спонукати купити більше, ніж планував відвідувач. Він міг зайти тільки за товаром «А», але, побачивши «В», підсвідомо відчув зв'язок і придбав його.

Навчання асоціативних правил – це метод машинного навчання, який базується на правилах та застосовується для виявлення прихованих взаємозв’язків між змінними у великих баз даних. Ключова мета – ідентифікувати сильні, статистично обґрунтовані патерни (зв’язки) між об’єктами, оцінюючи їх за допомогою спеціальних метрик «цікавості».

Асоціативні правила є одним з ключових методів інтелектуального аналізу даних, що дозволяє знаходити ці закономірності. Вони мають інтуїтивно зрозумілий вигляд: «Якщо {A}, то {B}». У контексті бібліотеки це означає: якщо читач взяв книгу (або набір книг) A, то з високою ймовірністю він також візьме книгу (або набір книг) B. Наприклад, аналіз може виявити правило: {Читачі, які беруть «Дюну»} → {часто також беруть «Гіперіон»}.

Але чи означає це що покупець завжди купуватиме чіпси, якщо він придбав колу. Або читач "Дюни" зацікавиться "Гіперіоном".

Приклади асоціацій, як-от інтерес читача до серії "Дюна" та потенційне зацікавлення "Гіперіоном", демонструють ключовий аспект методу: навчання асоціативних правил не встановлює причинно-наслідковий зв'язок.

Метод має на меті виключно кількісну оцінку частоти спільної появи двох або більше елементів у наборі даних. Наприклад, спільна купівля товару «А» та товару «В» не означає, що купівля «А» спричинила купівлю «В».

Крім того, метод асоціативних правил не враховує послідовність або порядок дій, розглядаючи всі елементи транзакції як єдину, невпорядковану множину. Ця особливість ускладнює інтерпретацію переваг клієнта та визначення першопричини.

Оскільки не всі виявлені закономірності мають однакову силу, критично важливим є вміння розрізняти випадкові збіги та статистично значущі асоціації.

Для оцінки сили та значущості виявлених правил застосовуються три основні метрики.[3]

**Підтримка (Support):** Ця метрика є оцінкою частоти появи певного набору елементів, що виникають разом у транзакціях.

Наприклад, при аналізі даних щодо дозвілля мешканців міста співпадіння ситуацій, коли {Погана погода} та {Люди йдуть у кіно}, може мати значно більшу підтримку, ніж коли {Погана погода} та {Люди йдуть на пікнік}. Кількість випадків, коли люди обирають закриті приміщення під час дощу, безумовно перевищує кількість разів, коли вони обирають активності на відкритому повітрі. Крім того, {Погана погода} майже завжди супроводжується {Збільшенням трафіку}, тоді як {Гарна погода} — {Прогулянками в парку}. Іншими словами, набір подій {Погана погода, Кіно} має значно більшу підтримку, ніж набір {Погана погода, Пікнік}.

Якщо відомо, що після певного порогу (наприклад, коли починається сильна злива) люди стабільно поводяться певним чином (скасовують плани на

вулиці), то можна вважати це значущою підтримкою. Однак якщо рівень підтримки низький (наприклад, лише кілька людей пішли в кіно під час дощу), то й сила асоціації буде слабкою — не можна зробити надійних висновків у такій ситуації.

$$\text{support} = P(A \cap B) = \frac{(\text{number of transactions containing } A \text{ and } B)}{(\text{total number of transactions})}$$

Рисунок 2.1 – Формула підтримки

**Довіра (Confidence):** Ця метрика вимірює умовну ймовірність того, що наслідок «Y» з'явиться у транзакції, якщо у ній вже присутній антецедент X. Іншими словами, вона показує, наскільки достовірно правило «Якщо X, то Y».

Ця метрика вказує наскільки впевнено можна стверджувати, що певне правило (погана погода) веде до іншого (похід в кіно). Іноді це значення може суперечити інтуїтивному розумінню. Якщо кількість спостережень загалом невелика, але спостерігається значне перекриття між подіями (наприклад, ми бачили лише 10 походів у кіно, і 8 з них були під час дощу), рівень довіри може бути дуже високим. Хоча інтуїтивно ми розуміємо, що такий зв’язок може бути хибним (можливо, люди просто чекали на прем'єру фільму). Наприклад, якщо дано {Прогулянка в парку}, то який рівень довіри для {Гарна погода}? (Він буде високим, але не 100%). Саме тому потрібна ще одна міра, щоб не порівнювати очевидні речі з неочевидними.

$$\text{conf}(X \Rightarrow Y) = P(Y|X) = \frac{\text{supp}(X \cap Y)}{\text{supp}(X)} = \frac{\text{number of transactions containing } X \text{ and } Y}{\text{number of transactions containing } X}$$

Рисунок 2.2 – Формула довіри

**Підйом (Lift):** Підйом є найважливішою метрикою, що визначає силу асоціації та визначає наскільки підтримка одного набору залежить від іншого.

Він показує який підйом має довіра до {Піти в кіно} у контексті наявності {Погана погода} у тому ж наборі. Аналізуючи, наскільки змінюється ймовірність того, що {Людина піде в кіно}, якщо відомо, що {Надворі погана погода}, порівняно з випадком, коли ми нічого не знаємо про погоду (тобто порівняно із середньою ймовірністю походу в кіно у будь-який день).

$$\text{lift}(X \Rightarrow Y) = \frac{\text{supp}(X \cup Y)}{\text{supp}(X) \times \text{supp}(Y)}$$

Рисунок 2.3 – Формула підйому

Коли наявність одного елемента підвищує ймовірність появи іншого в наборі, підйом буде більшим за 1. Це свідчить про сильну асоціацію, часту спільну появу та більшу надійність правила {X} → {Y}. Саме ця метрика допомагає у випадках, подібних до розміщення товарів у супермаркеті, про що ми говорили раніше — вона дозволяє визначити, які товари дійсно пов’язані між собою і розташування яких поруч має сенс.

Таким чином, для виявлення справді цінних закономірностей необхідно використовувати всі три метрики в комплексі. Підтримка відсіює випадкові, поодинокі збіги. Довіра показує умовну ймовірність правила. Але саме Підйом є вирішальним індикатором, що відокремлює справжню, неочевидну асоціацію від простої одночасної популярності двох незалежних елементів. Загальний вплив метрик на пошук асоціативних правил зображено на рисунку 2.4

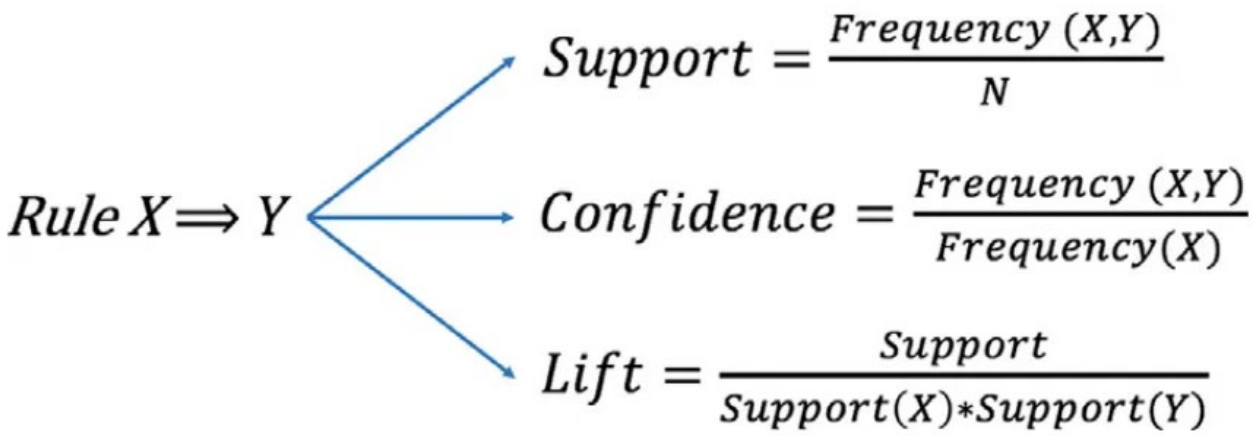



Рисунок 2.4 Вплив підтримки, довіри та підйому на пошук асоціативних правил

При розробці інформаційної системи бібліотеки, особливо її рекомендаційного модуля, важливо оцінити доцільність обраного методу. Застосування аналізу асоціативних правил має низку суттєвих переваг порівняно з іншими підходами (наприклад, колаборативною фільтрацією), але також має певні обмеження.

Переваги:

1. Простота та інтерпретованість. Правила є зрозумілими для людини. Їх легко пояснити та інтерпретувати.
2. Не вимагає персональних даних. Для побудови правил не потрібні оцінки чи історія транзакцій конкретного користувача, лише дані про видачі книг.
3. Ефективність на великих даних. Існують добре відомі та оптимізовані алгоритми для знаходження правил у баз даних. Наприклад, Apriori.

Недоліки:

1. Холодний старт. Алгоритм не може давати рекомендації для нових книг, оскільки для них ще нема правил про спільне взяття з іншими книгами.
2. Не враховує послідовність. Асоціативні правила не беруть до уваги порядок, у якому читач брав книги, що може бути важливим для серій

книг або певних тем.

3. Велика кількість правил. Може генеруватися велика кількість правил, більшість з яких є тривіальними.
4. Не враховує контекст. Наприклад, сезонність інтересу до певних тем або мету читання (розвага,  навчання).

## 2.2 Дослідження Apriori

Apriori — це класичний і один із найвідоміших алгоритмів для пошуку асоціативних правил у базах даних транзакцій. Його головна мета — знайти набори елементів, що часто трапляються. Наприклад, у контексті бібліотеки — виявити книги, які читачі часто беруть разом[4].

Саме на основі цих наборів згодом будуються рекомендаційні функції формату «З цією книгою також читають...», що є прямим застосуванням CRM-стратегій для покращення сервісу.

На перший погляд, задача пошуку наборів здається простою. Однак, якщо в бібліотеці є, наприклад, 1000 унікальних книг, кількість можливих комбінацій є дуже великою що робить повний перебір абсолютно неможливим з точки зору обчислень.

Алгоритм Apriori вирішує цю проблему за допомогою евристичного принципу, який також відомий як властивість анти-монотонності підтримки.

Принцип стверджує: якщо набір елементів зустрічається часто, то всі його підмножини також гарантовано зустрічаються часто.

З цього випливає набагато важливіше зворотне правило, яке і лежить в основі ефективності алгоритму: якщо набір елементів зустрічається рідко, то всі його підмножини (будь-які набори, що його містять) також гарантовано будуть рідкими.

Це припущення дозволяє алгоритму "відсікати" величезну кількість потенційних наборів-кандидатів, навіть не перевіряючи їх.

Приклад: Якщо ми вже знаємо, що набір {Книга А, Книга Б} є нечастим (тобто його підтримка нижча за встановлений нами мінімальний поріг), нам немає жодного сенсу перевіряти підтримку для наборів {Книга А, Книга Б, Книга В} або {Книга А, Книга Б, Книга Г, Книга Д}. Вони апріорі будуть нечастими і можуть бути відкинуті.

Інше формулювання цього ж правила: зі зростанням розміру набору елементів, його підтримка або зменшується, або залишається такою ж.

Візуалізація роботи цього алгоритму зображено на рисунку 2.5

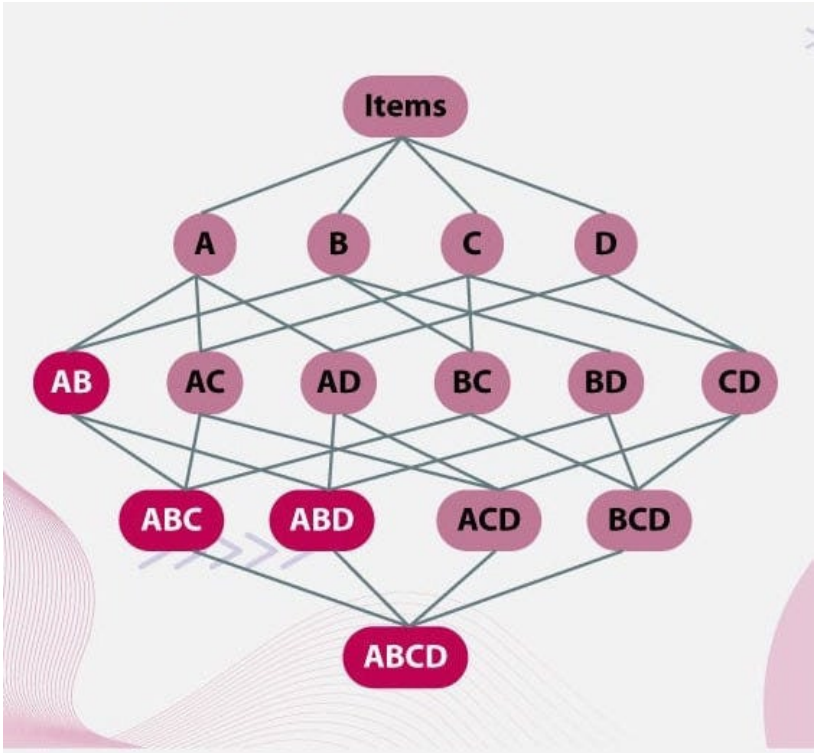



Рисунок 2.5 – Відсіювання апріорі нечастих кандидатів

Алгоритм працює ітераціями та складається з таких кроків:

1) Встановлення порогу (Min_Support) Перш за все, ми самі визначаємо мінімальну підтримку — наприклад, "вважати набір частим, якщо він зустрічається хоча б у 5% транзакцій. Крім того, потрібно побудувати бінарну таблицю транзакцій.

2)Генерація наборів, які трапляються часто. Алгоритм один раз сканує

всю базу даних і рахує підтримку для кожного окремого елемента. Всі елементи, підтримка яких нижча за поріг, відкидаються. Ті, що залишилися, формують набір елементів.

3) Генерація наборів з двох елементів. Алгоритм бере набір з одних елементів і генерує з нього кандидатів у набори з двох, об'єднуючи кожен елемент з кожним (наприклад, з {А}, {Б}, {В} він створить кандидатів {А, Б}, {А, В}, {Б, В}). Алгоритм вдруге сканує базу даних. Ті кандидати, що долають поріг, стають новими наборами елементів.

4) Генерація наборів з трьох елементів. На цьому етапі працює відсіювання Apriori. Для кандидата він перевіряє підмножини та якщо хоч одна є нечастою то він відкидає повністю всю множину. Цей шаг повторюється допоки алгоритм може знайти нові набори(для чотирьох, п’яти елементів і т.д.)

5)Генерація правил. Після того, як всі набори знайдені, алгоритм генерує з них асоціативні правила. Наприклад, з частого набору {Хліб, Масло} він генерує два правила-кандидати: {Хліб} → {Масло} та {Масло} → {Хліб}. Потім для кожного правила розраховується довіра і відкидаються ті, що не проходять її мінімальний поріг.

Візуалізація алгоритму у форматі блок-схеми зображено на рисунку 2.6

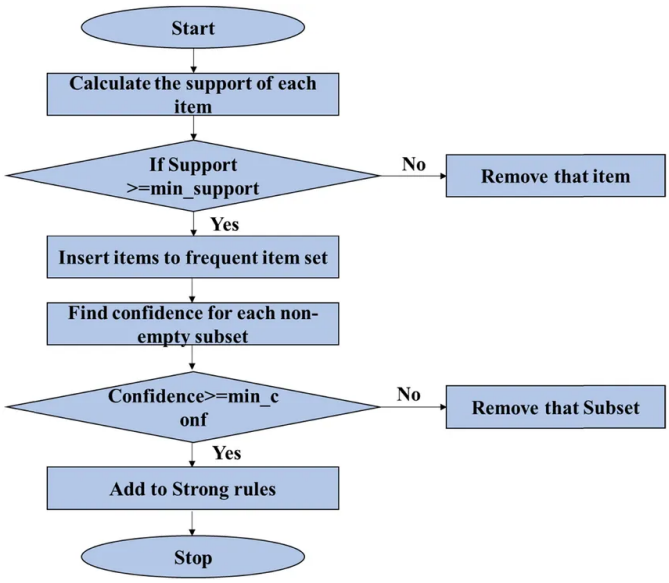



Рисунок 2.6 – Алгоритм Apriori

## 2.3 Дослідження FP-Growth

Алгоритм FP-Grow був розроблений як пряме вдосконалення та альтернатива Apriori[5]. Його створили, щоб вирішити дві найболючіші проблеми Apriori:

Продуктивність: Уникнути багаторазового і дорогого сканування всієї бази даних для кожного рівня k-наборів. Лише два повних проходи по базі даних.

Генерація кандидатів: Уникнути створення мільярдів потенційних наборів-кандидатів, які потім доводиться перевіряти.

Основна ідея FP-Growth — уникати генерації кандидатів взагалі. Замість цього, він стискає всю базу даних транзакцій у компактну деревоподібну структуру в пам'яті — FP-Tree (Дерево частих патернів). Подальший аналіз та майнінг відбувається вже з цим деревом, без жодних звернень до початкової бази даних.

Весь алгоритм можна умовно поділити на два великі етапи: Побудова дерева та майнінг наборів з нього.

Алгоритм складається з таких етапів:

1)Попередня обробка та фільтрація елементів. Алгоритм перший раз проходить по базі даних транзакцій та видаляє всі елементи, чия підтримка нижча за мінімальний поріг. Залишені елементи впорядковуються у порядку спадання підтримки. Наприклад: D (0.8) → C (0.7) → B (0.6) → E (0.4)

2)Побудова FP-дерева. Відбувається другий прохід по базі транзакцій, під час якого відкидаємо рідкісні елементи, яких немає в списку, побудованому на минулому етапі та сортуємо решту у порядку спадання. Вставляємо цей впорядкований список у дерево як шлях. Кожен вузол дерева має мітку елемента та лічильник частоти. Якщо шлях транзакції співпадає з існуючою

гілкою – лічильники збільшуються. Якщо частина шляху відсутня – створюються нові вузли. Таким чином усі транзакції накладаються одна на одну, утворюючи компактну структуру, де спільні префікси об'єднані. На цьому етапі можливі два варіанти структури: єдине дерево з умовним коренем або ліс дерев, коли деякі транзакції не мають спільного префікса. На практиці, зазвичай, використовують один загальний корінь.

На цьому етапі база даних стискається у пам’яті до мінімального необхідного вигляду. Подальші кроки виконуються без звернень до бази даних.

3)Пошук паттернів через умовні дерева. Знайдення наборів відбувається знизу догори. Починаємо з листа, тобто з елемента у якого найменша підтримка. Визначаємо всі шляхи що ведуть до цього елемента, формуємо умовні паттерни і рахуємо для них нові значення підтримки. Будуємо умовне FP-дерево. Набори, знайдені в умовному дереві, об’єднуються з елементом, який аналізується.

Елемент з меншою підтримкою появляється як наслідок комбінацій більш частих елементів, тому розбір ведеться від листя до кореня.

Наприклад, для найменш частого елемента E шляхи можуть виглядати так: DCBE(0.2), DCE(0.1), CBE(0.1). Якщо шляхи не задовольняють мінімальну підтримку та не виконують мінімальну довіру, то вони не проходять.

Після обробки елемента всі вузли з ним видаляються. Завдяки цьому робота з наступним елементом прискорюється.

4)Генерація асоціативних правил. Після третього кроку отримуємо список елементів, які зустрічаються часто разом Тепер алгоритм перетворює їх на правила вигляду «Якщо {A}, то {B}».

Якщо довіра правила перевищує заданий мінімальний поріг, правило зберігається як значуще. Якщо ні — відкидається. Правила з високою довірою стають результатом роботи системи і передаються користувачеві (або в модуль рекомендацій).

## 2.4 Дослідження Eclat

Eclat це алгоритм схожий на Apriori[6]. Основна різниця полягає у способі зберігання та пошуку даних. Apriori використовує горизонтальний формат, в якому кожна транзакція – це рядок. Також він багаторазово сканує базу даних, тобто використовує пошук в ширину.

Eclat використовує вертикальний формат, де кожен елемент пов'язаний зі списком ідентифікаторів транзакцій. Він застосовує пошук у глибину. Це потребує значно менше сканувань та використання пам’яті. Завдяки цьому можна сказати, що Eclat, як і FP-Growth, є покращеною версією Apriori, оскільки забезпечує кращу масштабованість і вищу обчислювальну ефективність.

Алгоритм Eclat базується на взаємодії наборів транзакцій для обчислення підтримки наборів елементів, уникаючи генерації підмножин, які, скоріше за все, не зустрічаються у даних.

Алгоритм складається з таких кроків:

1)Створення списку транзакцій для кожного окремого предмета. Тобто це список транзакцій, в яких зустрічається предмет. Наприклад, хліб – { T1, T4, T5, T7, T8, T9 }, кола – {T2, T4}.

2)Алгоритм продовжує роботу, рекурсивно поєднуючи списки. Підтримка набору елементів визначається перетином відповідних наборів. Наприклад, {хліб, кола} – {T4}

3)Рекурсивний виклик та генерація більших наборів. Алгоритм продовжує роботу, комбінуючи пари наборів та обчислюючи їхню підтримку через перетин. Рекурсія триває, доки не можна згенерувати нові набори.

4)Зупинка, коли більше не можна більше знайти наборів. Алгоритм зупиняється, коли жодна комбінація не відповідає мінімальній підтримці.

Проте у алгоритму э недоліки. Це високе споживання пам’яті, довга обробка великих транзакцій та низька ефективність у розріджених наборах

даних, адже перетини дають дуже малі результати.

## 2.5 Порівняння алгоритмів на тестових даних

Для проведення експерименту обрано та встановлено бібліотеку SPMF (Sequential Pattern Mining Framework) з графічним інтерфейсом. Ця бібліотека реалізує більшість відомих алгоритмів для пошуку асоціативних правил[7].

Завдяки графічному інтерфейсу можна не реалізовувати кожен алгоритм, а швидко перевірити роботу кожного з них на тестових даних і вже після аналізу результатів обрати один з них для впровадження в систему.

За допомогою використання моделі штучного інтелекту Gemini, я згенерував синтетичний набір даних, що імітує реальні транзакції книжкової бібліотеки. Це десять тисяч транзакцій бронювання книг з асортименту зі ста книг.

Штучні дані імітують реальні транзакцій, які могли б зробити користувачі бібліотеки. Основна частина транзакцій містить від 2 до 5 книг. Проте частина з них містить лише 1 або більше 5. Графік з розподілом книг на транзакції наведено на рисунку 2.7

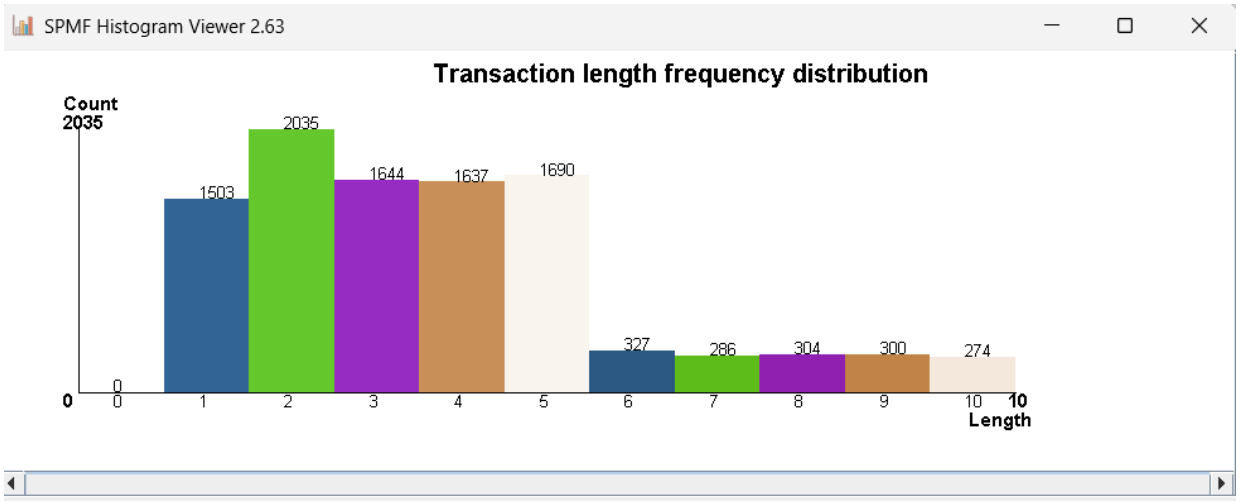



Рисунок 2.7 – Розподіл довжини транзакцій

У дані закладені приховані паттерни. Наприклад, бронювання серії книг «Гаррі Поттер» щоб перевірити здатність алгоритмів знаходити асоціативні правила.

Дані збережено у звичайному .txt файлі, де кожна транзакція — це набір ID книг, який починається в нового рядка.

В налаштуваннях для всіх алгоритмів обрані однакові параметри. Мінімальна підтримка – 0.01. Всі необов'язкові поля залишені порожніми. На рисунках 2.8-2.13 зображені налаштування та результати виконання алгоритмів.

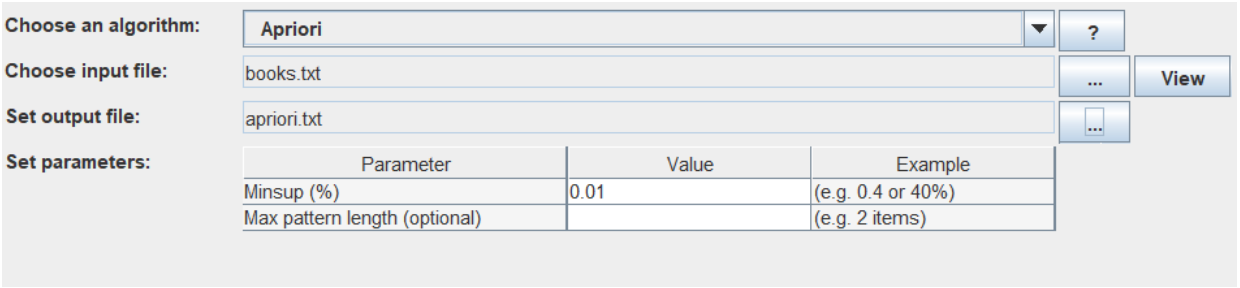



Рисунок 2.8 – Налаштування Apriori

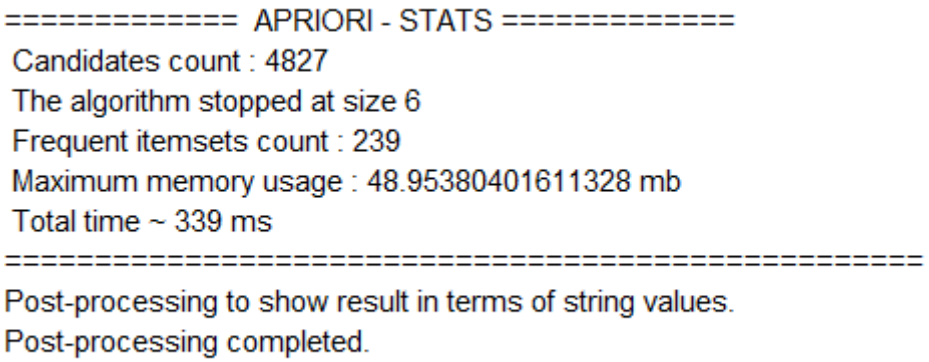



Рисунок 2.9 – Результат роботи Apriori

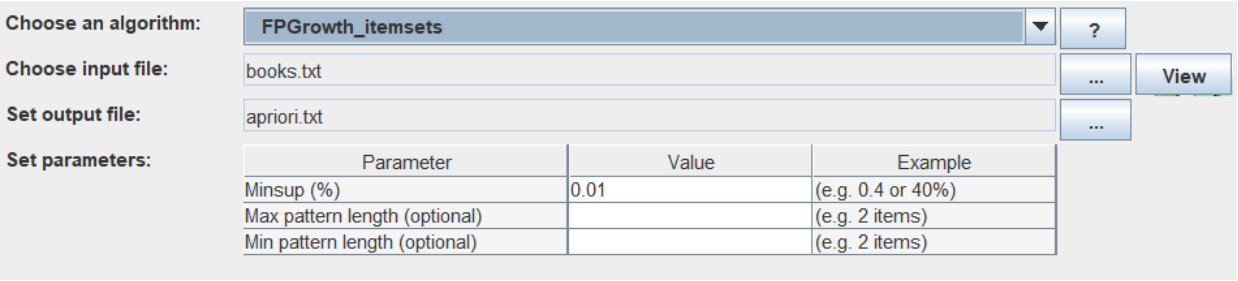



Рисунок 2.10 – Налаштування FP-Growth

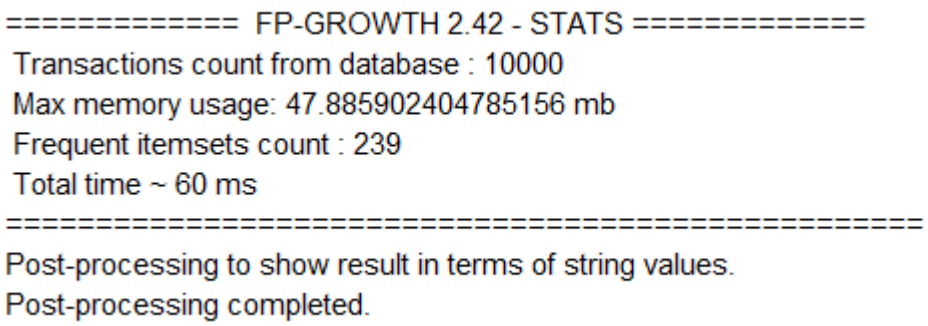



Рисунок 2.11 – Результат роботи FP-Growth

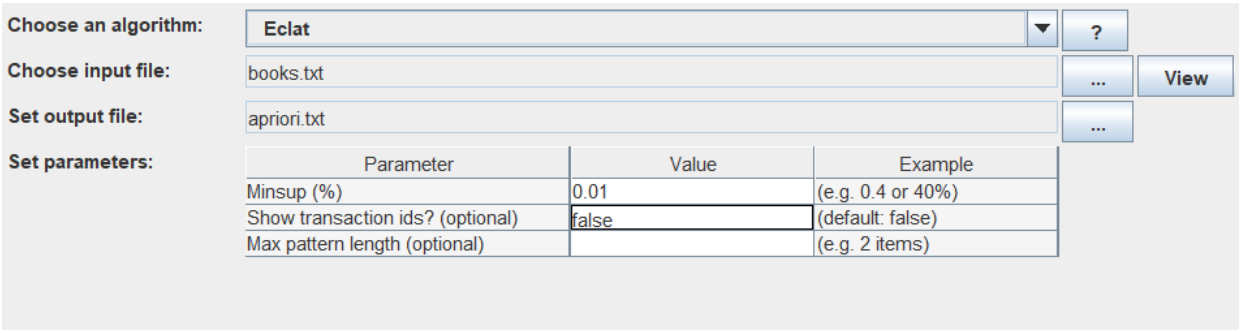



Рисунок 2.12 – Налаштування Eclat

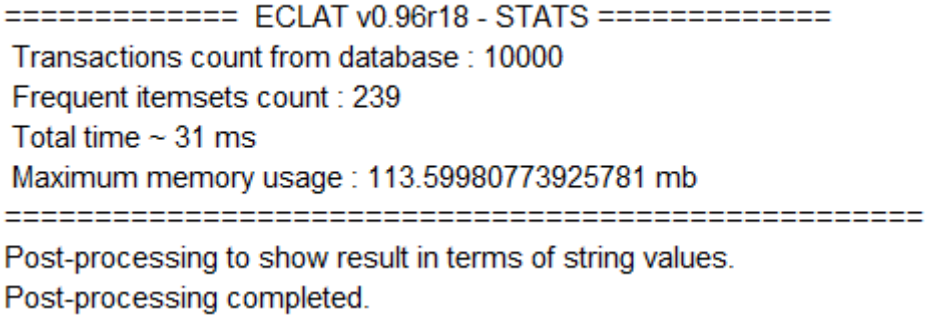



Рисунок 2.13 – Результат роботи Eclat

На основі отриманих даних побудовано порівняльні діаграми продуктивності. Діаграм зображено на рисунках 2.14-2.15

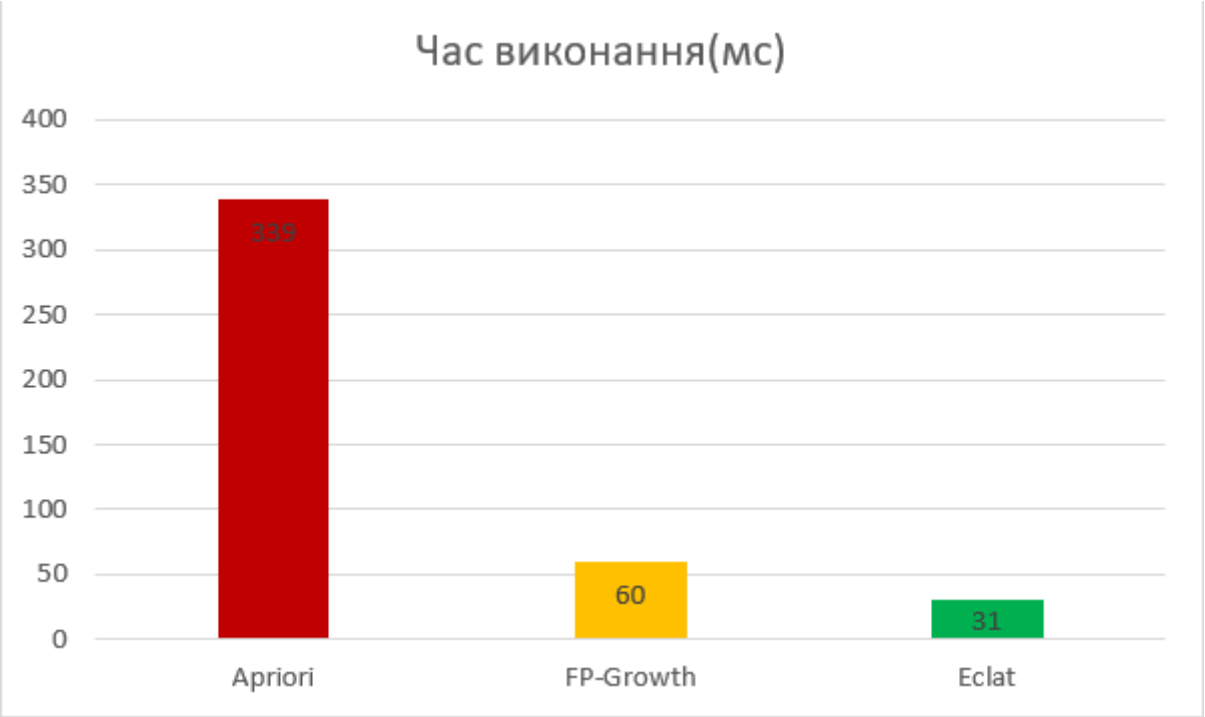



Рисунок 2.14 – Час виконання

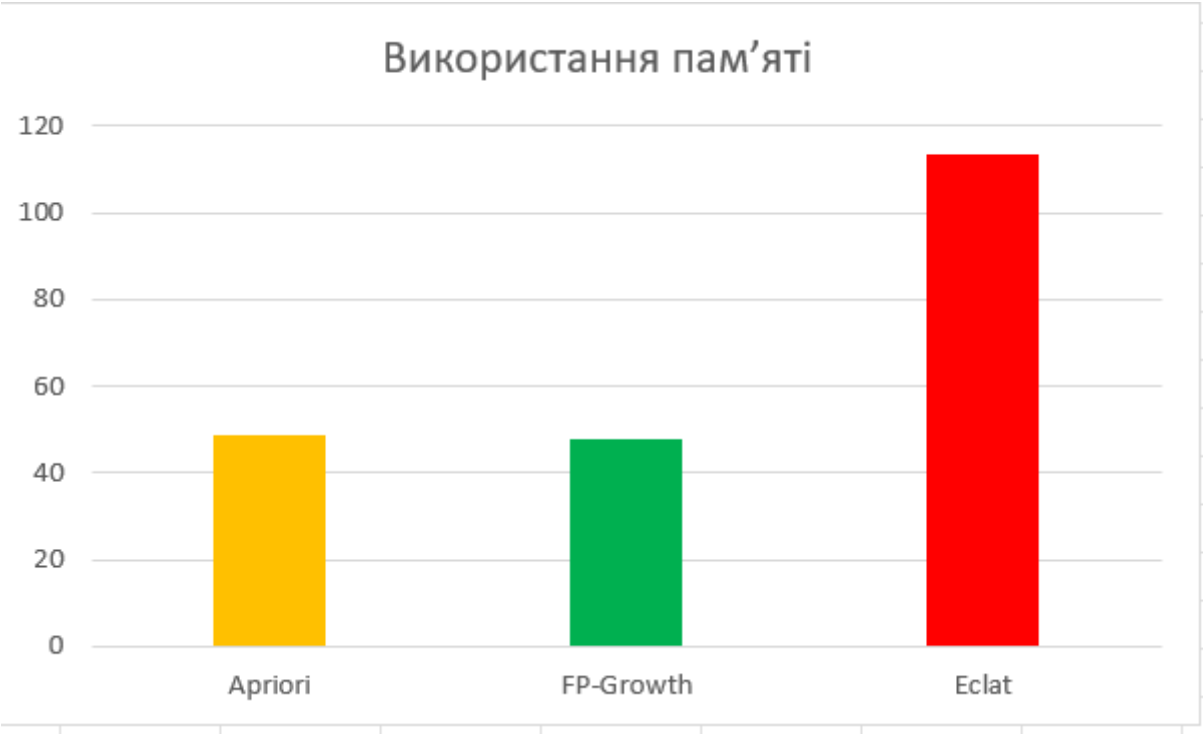


Рисунок 2.15 – Використання пам’яті

**Швидкість**: Eclat є беззаперечним лідером на цьому наборі даних, трохи випереджаючи FP-Growth. Apriori значно відстає через ітеративний підхід.

**Пам'ять**: FP-Growth є найбільш економним алгоритмом. Eclat споживає найбільше ресурсів RAM, оскільки завантажує структури перетинів у пам'ять.

Для впровадження в систему обрано алгоритм FP-Growth. Попри те, що Eclat продемонстрував кращий час опрацювання на вибірці з десяти тисяч транзакцій, швидкість FP-Growth є прийнятною. Натомість, стабільність FP-Growth щодо споживання пам'яті робить його більш надійним для роботи у майбутньому, коли база даних наповниться великою кількістю транзакцій.

# 3 ПРОЕКТУВАННЯ СИСТЕМИ

## 3.1 Розробка функціональних вимог до системи

Для визначення функціональних вимог ІС бібліотеки проведено функціональне моделювання системи, яке представлено у вигляді контекстної діаграми та її декомпозиції першого та другого рівня[8]. Діаграма наведена на рис. 3.1-3.5.

На рис.3.1 наведено головну бізнес-функцію ІС бібліотеки «Облік діяльності бібліотеки».

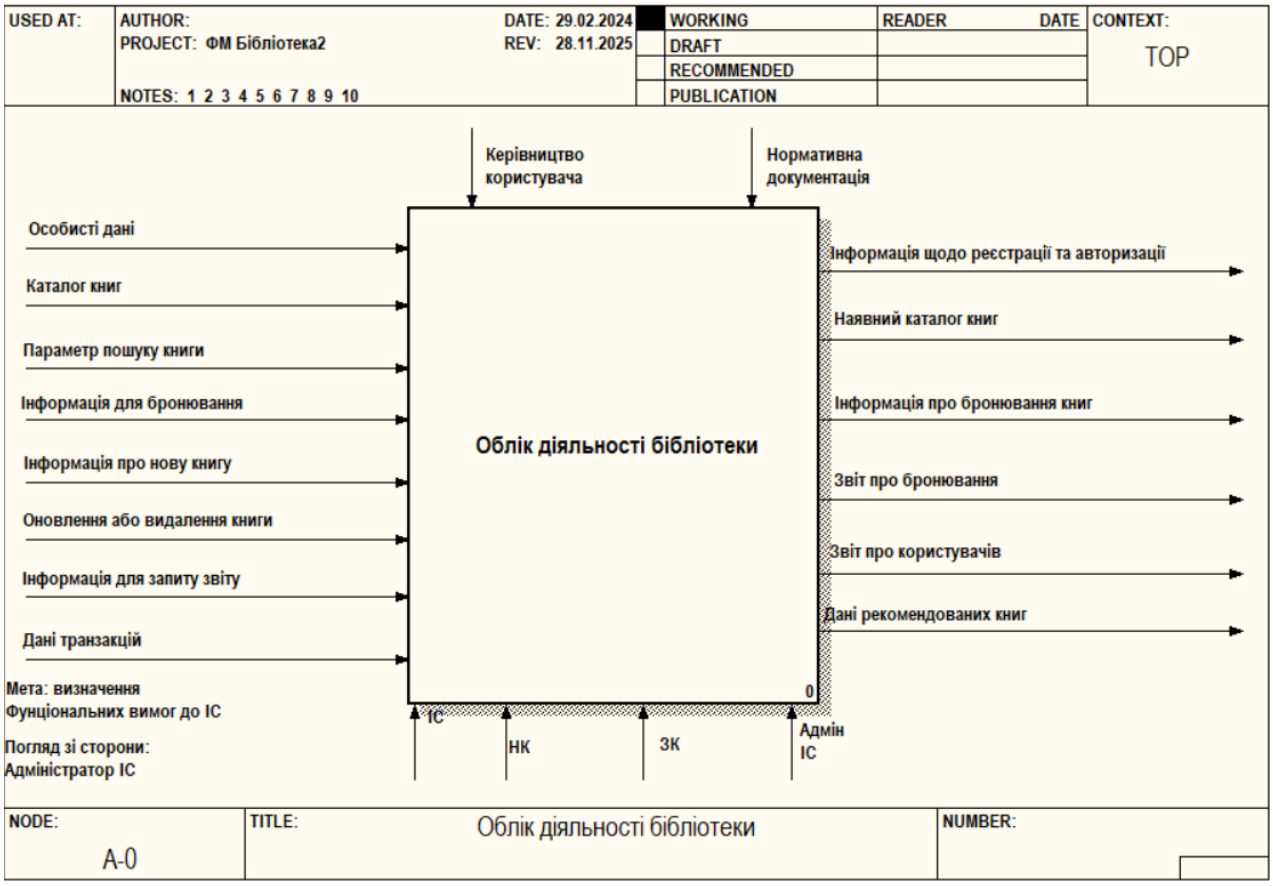



Рисунок 3.1- Контекстна діаграма «Облік діяльності бібліотеки»

На вхід головна бізнес-функція отримує:

- каталог книг. Це база даних або система зберігання інформації про доступні книги. Каталог може включати такі дані, як назва книги, автор, жанр, ISBN, кількість екземплярів, наявність тощо;
- параметри пошуку. Це дані, які користувач може ввести для фільтрації і пошуку конкретних книг у каталозі. Параметри можуть включати автора, назву книги, жанр, рік видання, наявність тощо;
- нова інформація про книгу. Це дані, які можуть бути додані або оновлені в каталозі після введення нової книги. Інформація може містити всі параметри книги, які дозволяють ідентифікувати та описати її;
- особисті дані. Це інформація, що ідентифікує користувача, який використовує систему каталогу книг. Може включати ім'я користувача, пароль або інші форми ідентифікації;
- інформація для бронювання. Це дані, які потрібні для здійснення бронювання користувачем;
- інформація про нову книгу. Це дані, які вводяться в систему при додаванні нової книги в каталог. Інформація може включати всі параметри книги, а також додаткові дані, такі як дата додавання, ідентифікатор книги в системі та інше;
- інформація для формування звіту. Це дані які вказують на те, який звіт хоче отримати адміністратор. Це може бути звіт про користувачів або про бронювання. Також він може фільтрувати інформацію в обох звітах. Наприклад, переглядати прострочені бронювання;
- дані транзакцій. Це інформація про всі бронювання, здійснені користувачами бібліотеки.

На вихід має такі дані:

- наявний каталог книг. Інформація про всі книги, які доступні в бібліотеці. Включає дані про назву, автора, жанр, наявність тощо;
- інформація щодо реєстрації та авторизації. Персональна інформація

про користувача, який зареєструвався в системі. Включає ім'я, прізвище, адресу електронної пошти, інші ідентифікаційні дані;

- інформація про бронювання книг. Деталі про те, які книги були заброньовані користувачем. Може включати назву книги, дату бронювання, термін видачі та інші відомості;
- звіт про бронювання. Детальний звіт про бронювання книг. З урахуванням хто забронював конкретну книгу і коли;
- звіт про читачів. Загальна інформація про читачів бібліотеки.
- дані рекомендованих книг. Це дані книг, які асоціативний алгоритм вирішив запропонувати користувачу.

Система розроблювалася та має використовуватися згідно з керівництвом користувача та нормативною документацією.

Головну бізнес-функцію можна декомпозувати на бізнес-процеси. На рисунку 3.2 наведено декомпозицію контекстної діаграми.

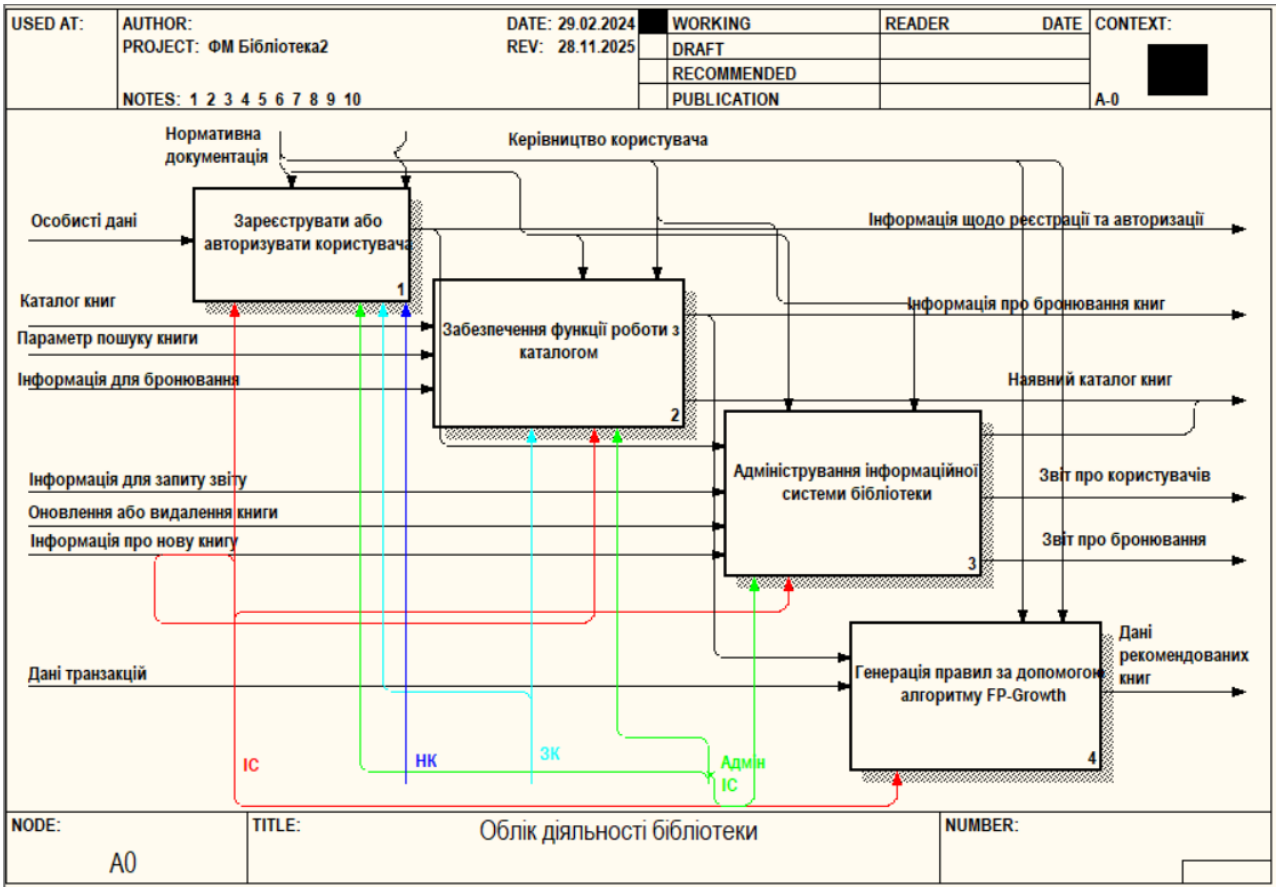


Рисунок 3.2 - Декомпозиція контекстної діаграми

а) Реєстрація та авторизація користувача . Бізнес-процес включає реєстрацію нових читачів. Реєстрація передбачає збір необхідної інформації для створення облікового запису, в той час як авторизація дозволяє вже зареєстрованим користувачам отримувати доступ до персоналізованого облікового запису. На рисунку 3.3 наведено декомпозицію реєстрації та авторизації;

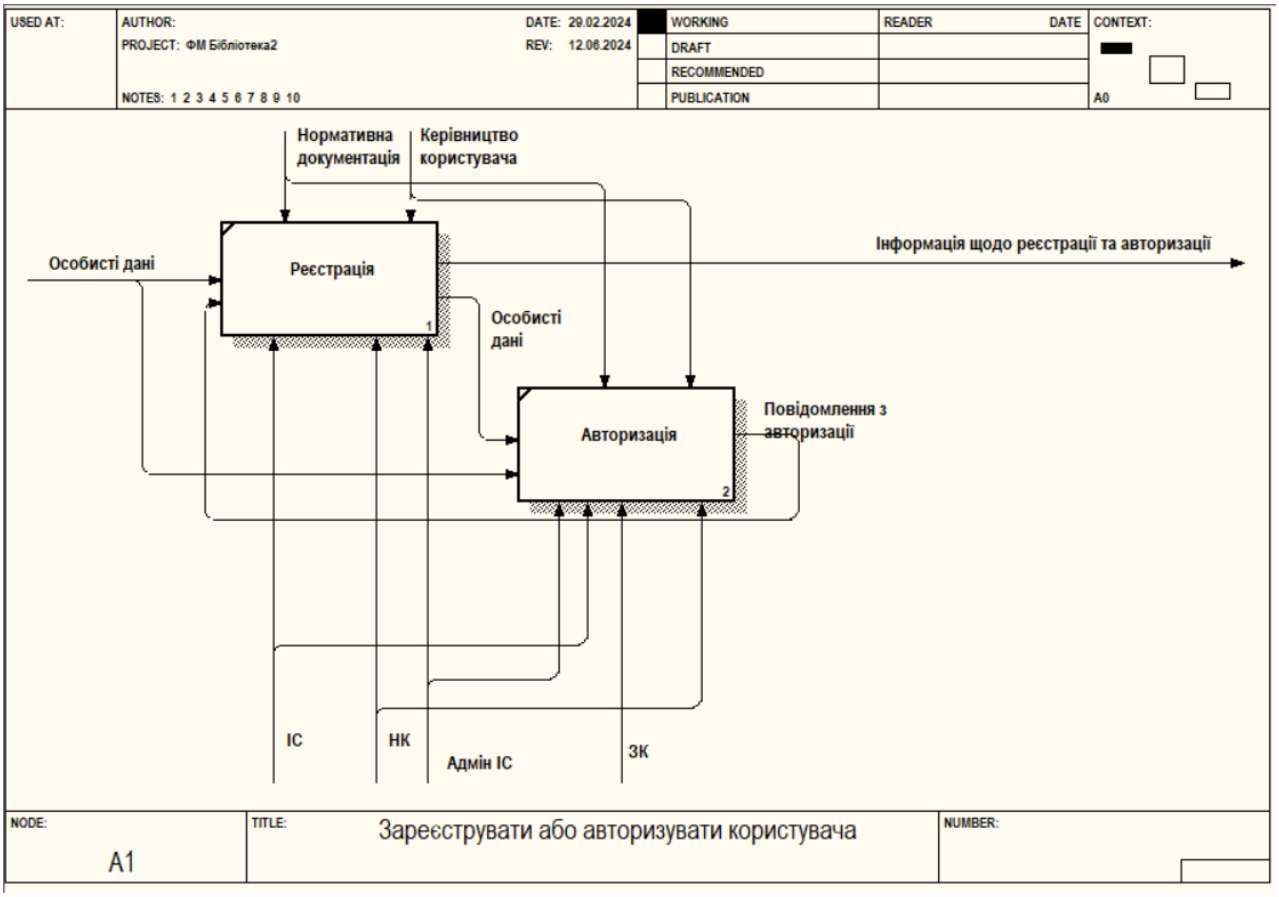



Рисунок 3.3 - Декомпозиція процесу «Реєстрація та авторизація»

б) забезпечення функції роботи з каталогом книг:

- перегляд каталогу книг. Користувачеві надається можливість переглядати всі наявні книги у системі, ознайомлюючись з їх загальною інформацією;

- пошук книг за параметром. Користувач може використовувати параметри пошуку (наприклад, автор, жанр, назва) для точного вибору книги в каталозі;
- інформація для бронювання книги. Користувач може здійснити бронювання книги для попереднього відведення її для себе на певний термін;
- бронювання книг. Можливість користувача бронювати книги а потім переглядати статус свого бронювання, включаючи такі деталі як статус, початок та закінчення бронювання.

На рисунку 3.4 подано декомпозицію процесу роботи з каталогом

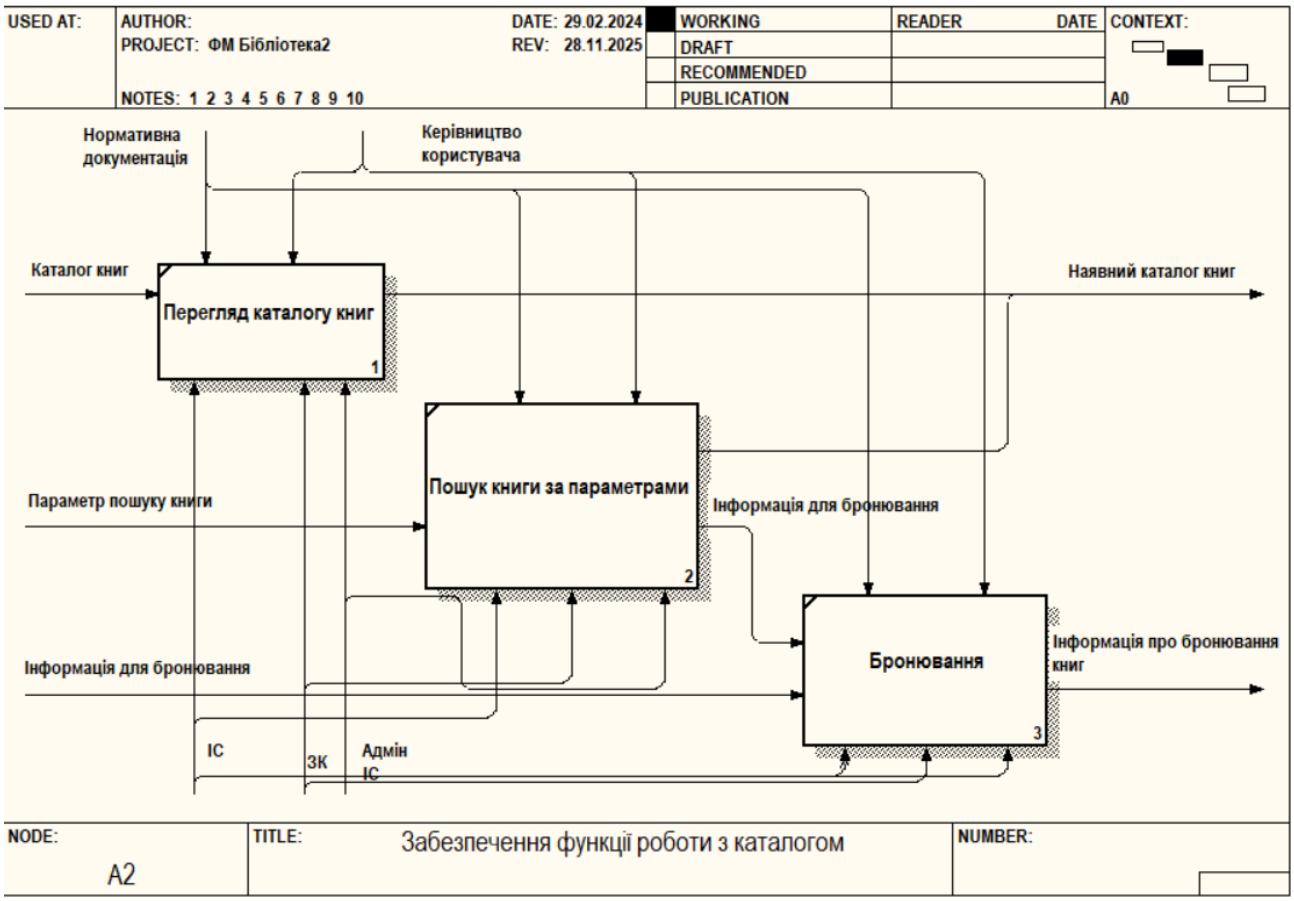



Рисунок 3.4 - Декомпозиція процесу «Забезпечення функції роботи з каталогом»

г) адміністрування інформаційної системи бібліотеки:

- облік читачів. Включає в себе реєстрацію та управління даними

читачів бібліотеки. А також можливість переглядати всіх зареєстрованих користувачів у системі;

- облік книг. Система ведення обліку книг, їх кількості, наявності та звітність щодо руху книг в бібліотеці, включаючи те, кому вона була видана та чи повернена книга назад.

- адміністратор може додавати та видаляти книги у бібліотеку або оновлювати дані про стару. А також переглядати звіти про читачів та бронювання.

На рисунку 3.5 подано декомпозицію процесу адміністрування інформаційної системи бібліотеки.

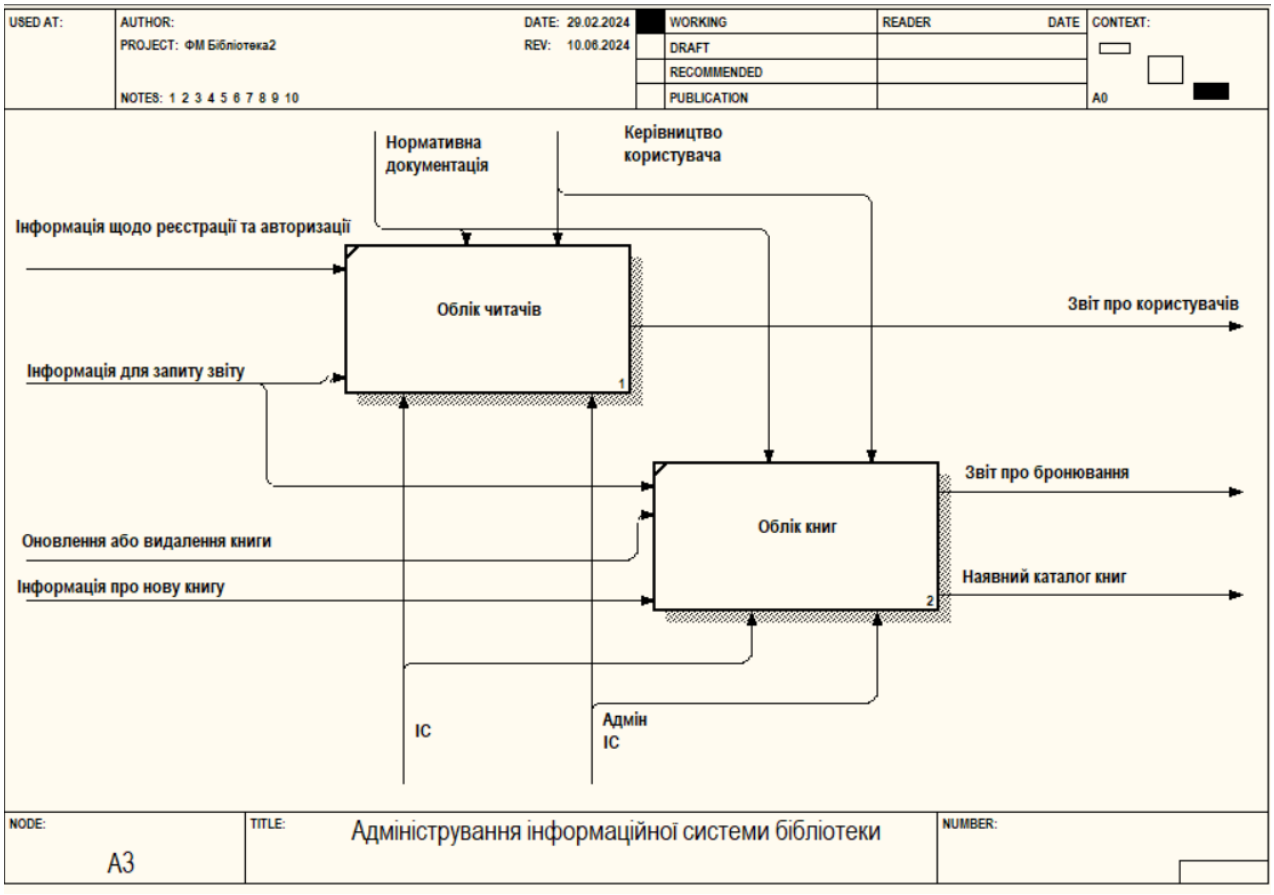



Рисунок 3.5 - Декомпозиція процесу «Адміністрування інформаційної системи бібліотеки»

г) генерація правил за допомогою алгоритму FP-Growth:

- аналіз транзакцій. Алгоритм оброблює всі транзакції в системі та знаходить пов’язані об’єкти;

- генерація рекомендацій. Після бронювання користувачу пропонуються книги схожі на ті, що присутні в його транзакції.

На рисунку 3.6 подано декомпозицію процесу генерація правил за допомогою алгоритму FP-Growth.

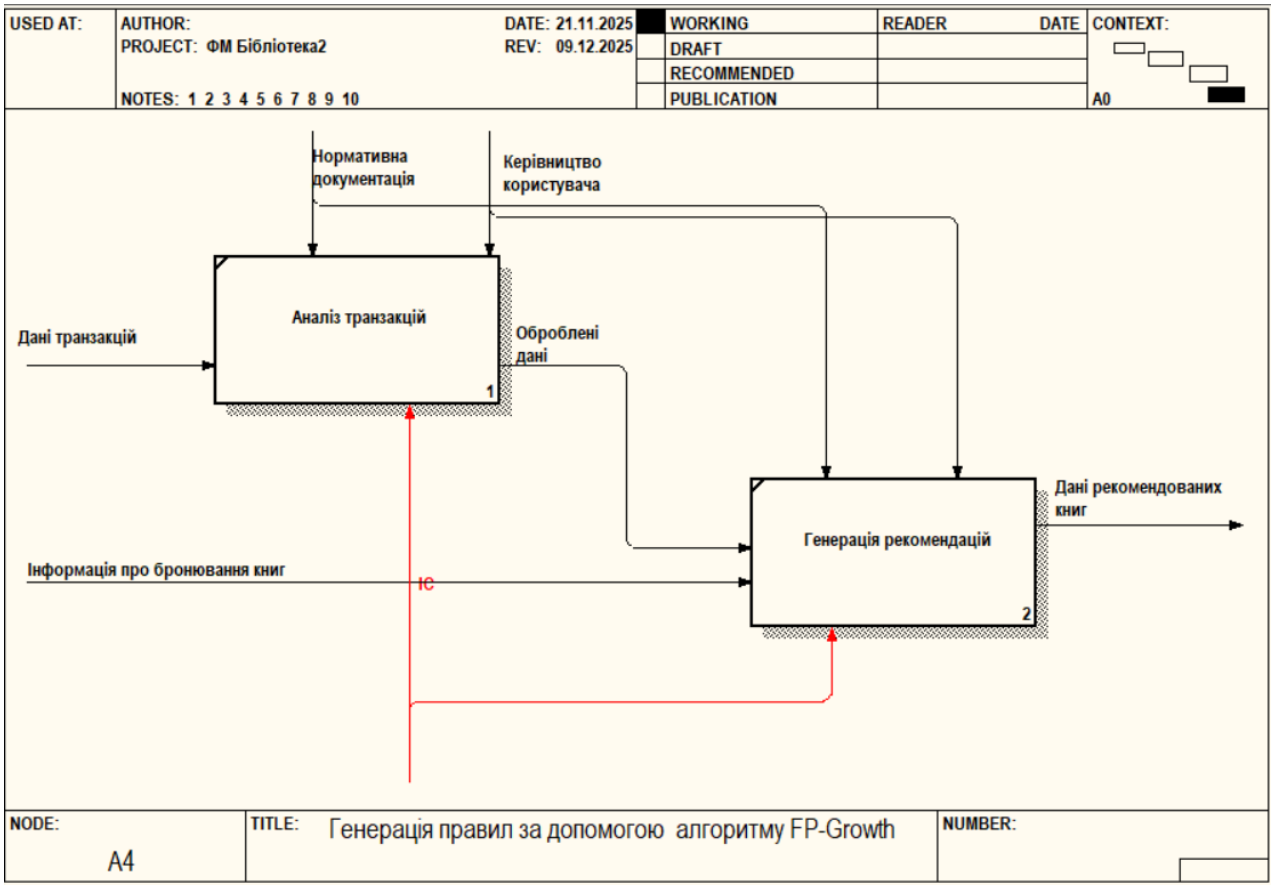



Рисунок 3.6 - Декомпозиція процесу «Генерація правил за допомогою алгоритму FP-Growth»

Діаграму дерева вузлів основного бізнес-процесу наведено на рисунку 3.7

Ця діаграма дозволить побачити ієрархію в моделі в зручному вигляді. Вона відображає всі процеси в інформаційній системі бібліотеки.

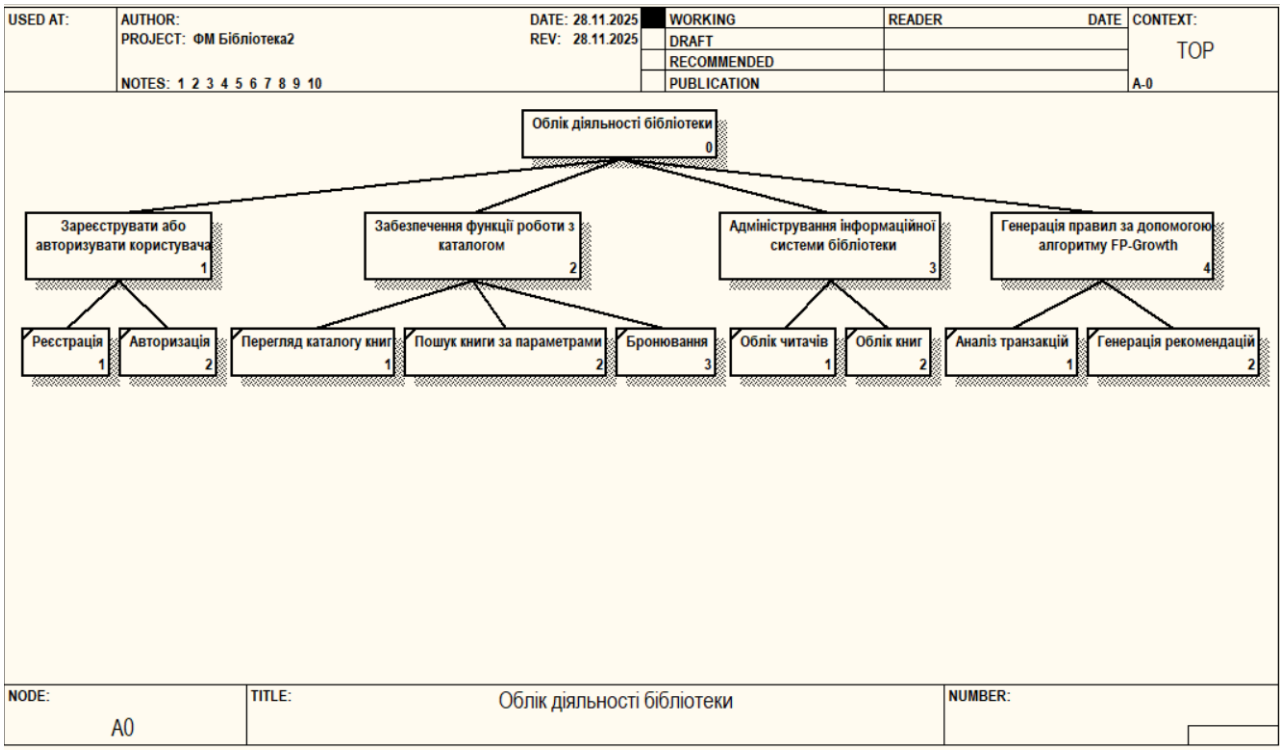



Рисунок 3.7 – Діаграма дерева вузлів

## 3.2 Розробка моделі потоків даних системи

Проектування моделей потоків даних є важливим етапом, оскільки дозволяє уточнити процеси і описати систему більш деталізовано[9].

В інформаційній системі бібліотеки передбачено три типи користувачів: зареєстрований, незареєстрований та адміністратор. До системи надходять різні потоки даних, які визначають процес взаємодії користувачів з інформаційною системою та різні сценарії її використання. Визначено такі потоки даних, що надходять до системи:

Особисті дані(Інформація про користувачів, такі як ім'я, прізвище, електронна пошта та інші персональні дані);

Параметри пошуку книги(Дані, що вказують на критерії пошуку книг, такі як мова);

Інформація для броні книги(Деталі, необхідні для здійснення бронювання книги, включаючи ідентифікатор книги та дату);

Інформація про нову книгу. Це дані, які вводяться в систему при

додаванні нової книги в каталог. Інформація може включати всі параметри книги, а також додаткові дані, такі як дата додавання, ідентифікатор книги в системі та інше;

Нова інформація про книгу. Це дані, які можуть бути додані або оновлені в каталозі після введення нової книги. Інформація може містити всі параметри книги, які дозволяють ідентифікувати її.

Інформація для видалення книги. Це дані про книгу яку потрібно видалити з каталогу.

Інформація для формування звіту. Це дані які потрібні для запиту до системи для формування звіту.

Потоки даних, що виходять з системи:

Інформація щодо реєстрація та авторизації(Дані, які підтверджують успішну реєстрацію або авторизацію користувача.);

Інформація про рекомендаційні книги.

Звіт про користувачів  (Звіт, який може включати список читачів, їх ролі та особисту інформацію);

Звіт про бронювання книг(Інформація про бронювання);

Отримані дані (Дані, що надходять від користувачів під час реєстрації, авторизації чи інших операцій);

Інформація для бронювання книги (інформація необхідна для того, щоб зарезервувати книгу для себе);

На рис. 3.8 наведено контексту діаграму потоків даних у системі бібліотеки.

Для отримання додаткової інформації щодо наявних потоків даних проведено декомпозицію контекстної діаграми і виділено основні потоки даних Декомпозиція контекстної діаграми подана на рис. 3.8

а) Реєстрація та авторизація читача. Бізнес-процес включає реєстрацію нових читачів у системі та їх подальшу авторизацію.

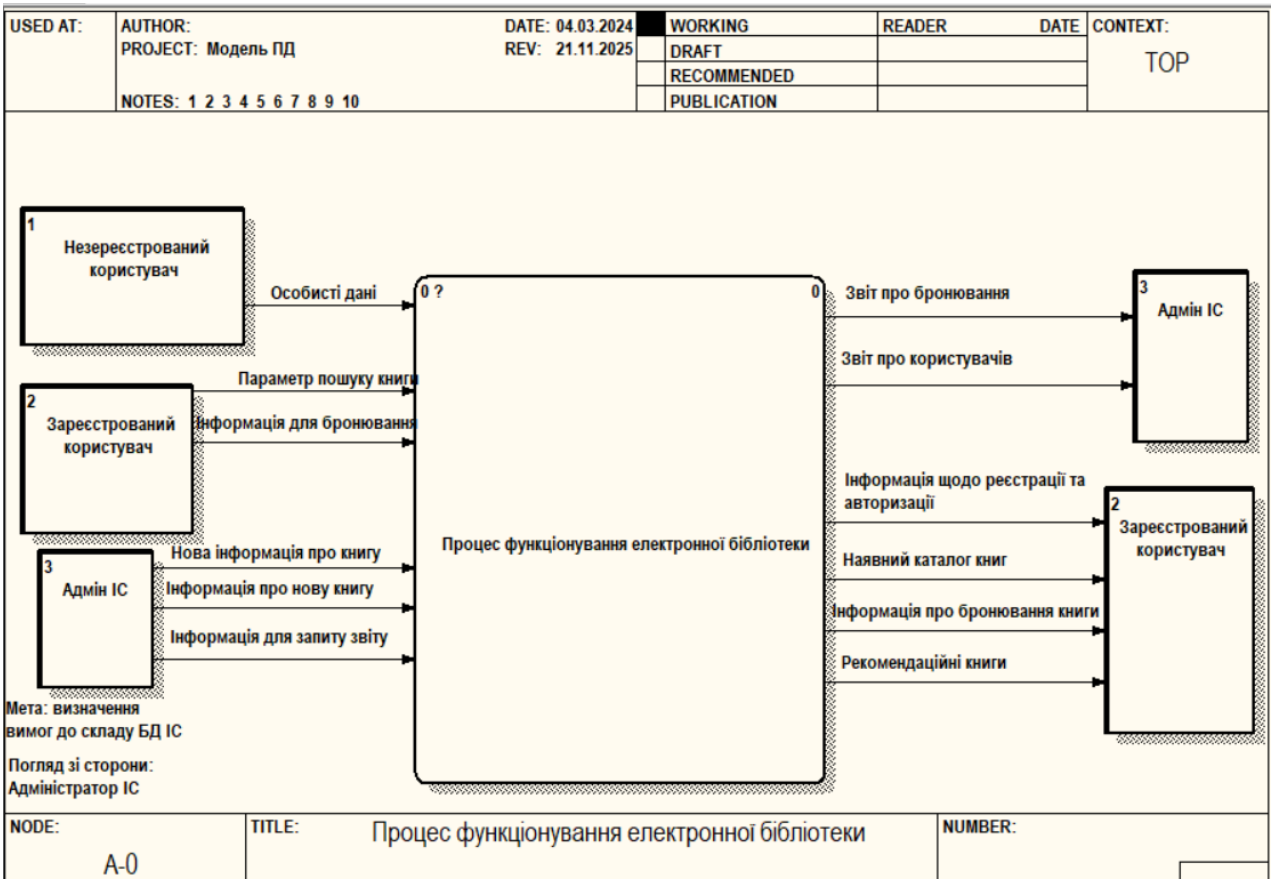



Рисунок 3.8 – Контекстна діаграма потоків даних

б) забезпечення функції роботи з каталогом книг:

перегляд каталогу книг. Користувачеві надається можливість переглядати всі наявні книги у системі, ознайомлюючись з їх загальною інформацією. Наприклад, ким та коли вона написана;

пошук книг за параметром. Користувач може використовувати параметри пошуку для точного вибору книги в каталозі;

бронювання книг. Користувач може здійснити бронювання книги для попереднього відведення її для себе на певний термін;

перегляд бронювання книг. Можливість користувача переглядати статус своїх бронювань, включаючи дату початку та кінця бронювання та іншу інформацію. Завдяки цьому він зможе не забути коли треба повернути книгу до бібліотеки;

г) адміністрування інформаційної системи бібліотеки:

облік про користувачів. Зберігання даних про реєстрацію та управління

даними читачів бібліотеки;

облік книг. Інформація про бронювання книг в бібліотеці. Включає в себе інформацію про те, хто взяв конкретну книгу, коли він її взяв, коли він має її повернути та статус його бронювання.

перегляд бронювання книг. Можливість користувача переглядати статус всіх бронювань, включаючи дату початку та кінця бронювання та іншу інформацію;

адміністратор може додати нову книгу у бібліотеку або змінити дані про вже існуючу. Крім того, він може видаляти книги або формувати звіти щодо бронювання.

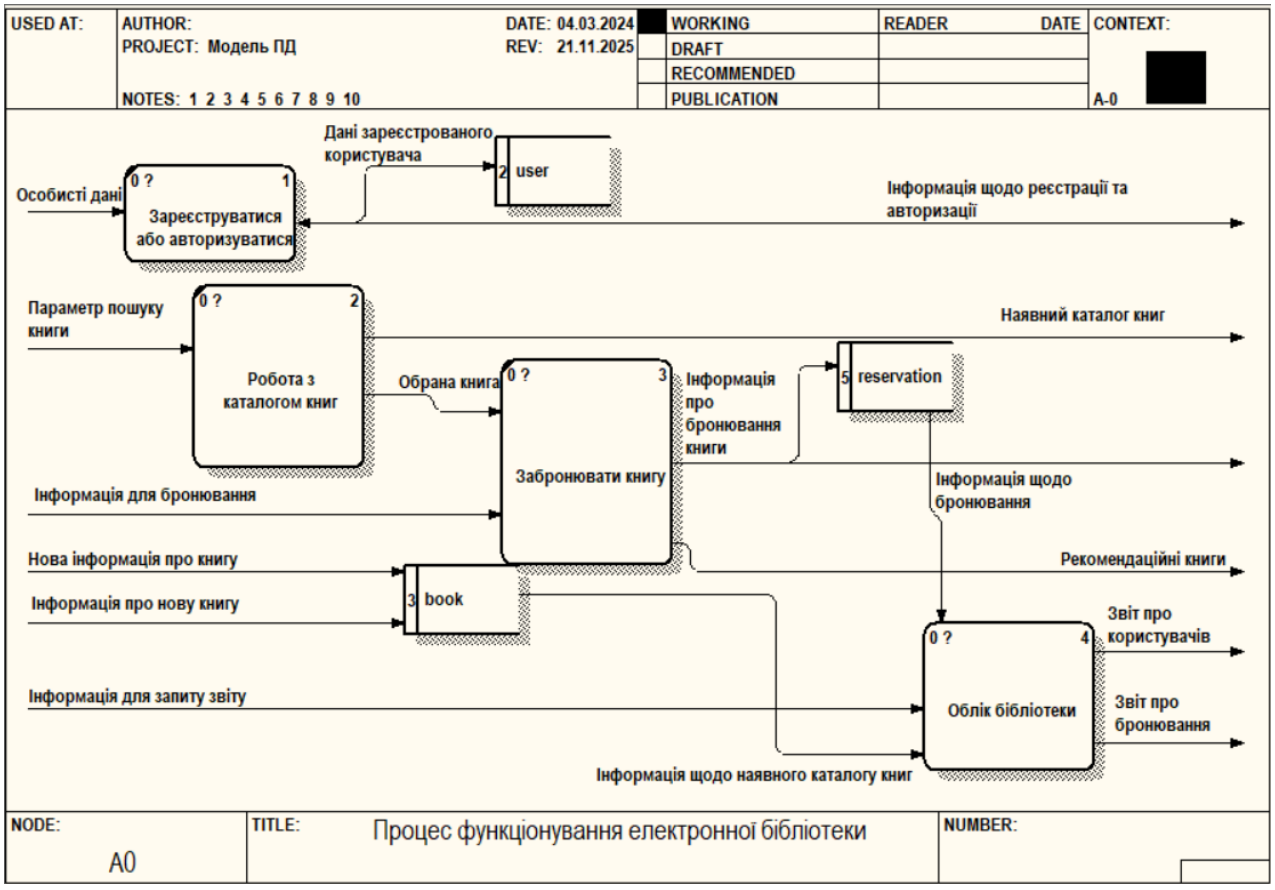



Рисунок 3.9 – Діаграма декомпозиції DFD

За моделюванням потоків даних визначені такі вимоги до бази даних :

Сутність користувач (User) -зберігає інформацію про користувачів та

адміністраторів. Тут можуть включатися дані, такі як ім'я, прізвище, по батькові, електронна пошта, інформація про роль (користувач, адміністратор), номер телефону, дата народження а також логін та пароль. Тобто всі персональні дані користувачів.

Сутність книга (Book) – зберігає інформацію про книги. Це сховище даних призначене для зберігання інформації про книги. Серед даних можуть бути назва книги, автор, жанр, опис, видавництво, мова та інші відомості, необхідні для ідентифікації та опису книг;

Сутність бронь (Reservation) – зберігає інформацію про резервації. Сховище Reservation відповідає за зберігання інформації про резервації книг читачами. Тут можуть бути включені дані про дату початку та кінеця резервації, статус, чи використана бронь та інші деталі, які визначають, яка книга зарезервована та кому;

Після декомпозиції обліку бібліотеки я отримав два процеси. Облік користувачів та облік книг в бібліотеці.

На рисунку 3.10 зображено декомпозицію потоків даних обліку книг в електронній бібліотеці.

На діаграмі декомпозиції обліку книг наведено такі потоки даних: інформація для бронювання книги, звіт про користувачів, персональні данні, створене бронювання, інформація для запиту звіту, інформація про книги, звіт про бронювання, інформація щодо наявного каталогу книг, інформація про книги.

Ця структура бази даних забезпечить зберігання та управління даними про користувачів, книжковий фонд та процес бронювання книг у бібліотеці. Вона не є надмірною та задовольняє всі потреби поставлені перед нею.

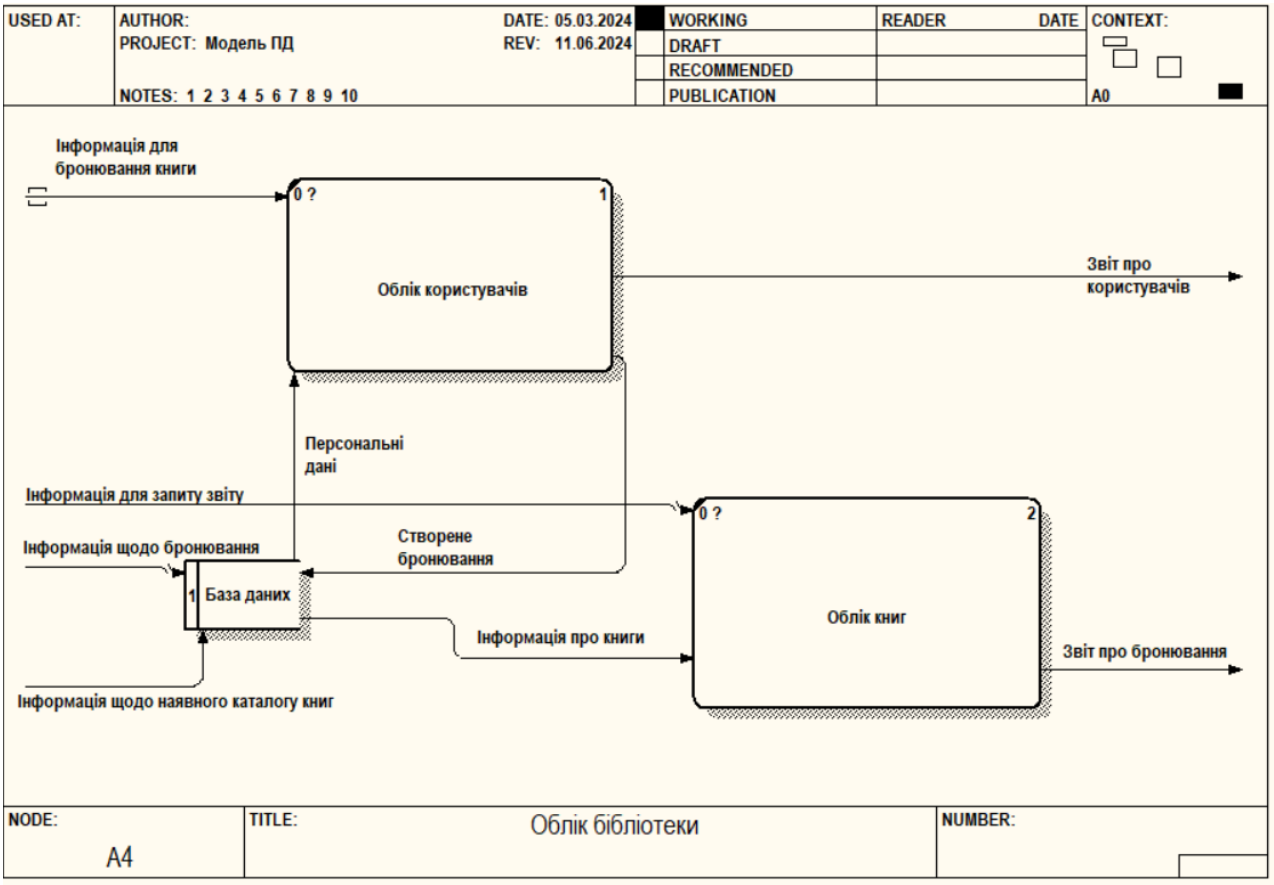



Рисунок 3.10 - Діаграма декомпозиції DFD обліку книг

## 3.3 Опис архітектури розробленої системи

Розроблені компоненти втілюють структуру з трьох етапів доступу до мережі Інтернет. Клієнтська, серверна та база даних.

У створенні клієнтської частини використовується мова JS та html. З метою розробки серверної частини вибрано мову Java та фреймворк Spring Boot. Для управління базою даних використовується система управління базами даних (СУБД) MySQL 8.3.0. На рисунку 3.11 наведена архітектура системи. Клієнт посилає запити серверу бібліотеки. MySQL обробляє запит та повертає результат або код помилки. Після цього клієнт відображає отримані дані.

Це сучасні технології, які добре підходять для розробки інформаційної системи бібліотеки.

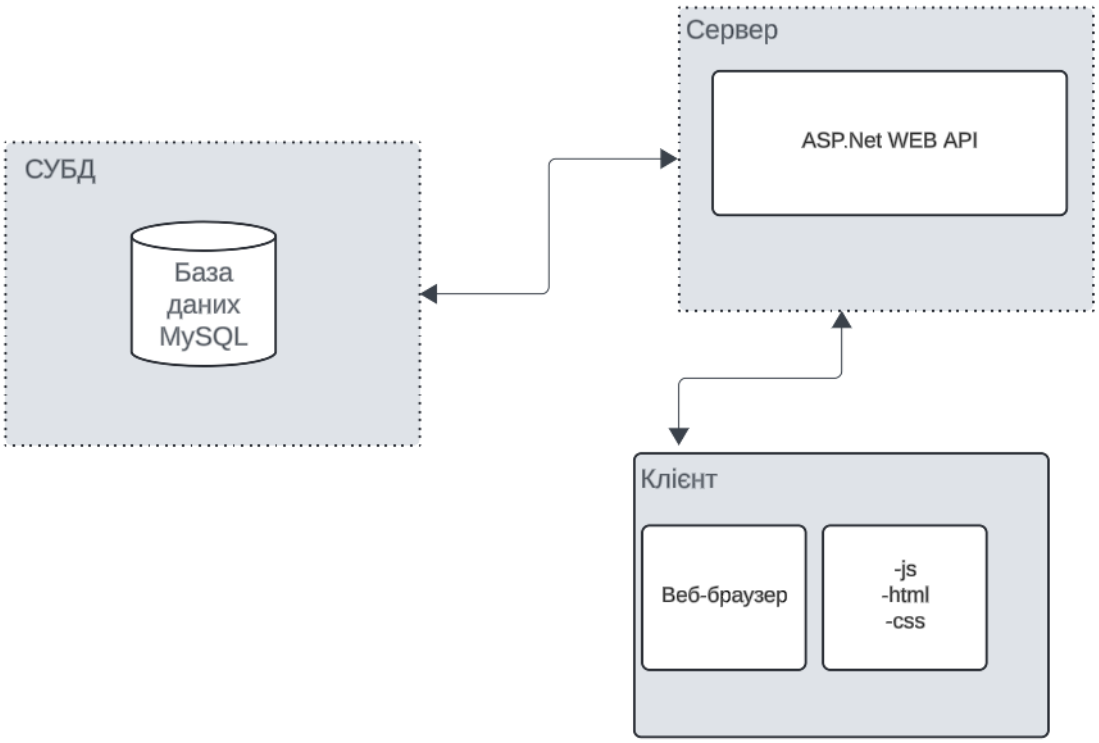



Рисунок 3.11 – Архітектура системи

## 3.4 Розробка діаграми варіантів використання системи

USE CASE (діаграма використання) відіграє важливу роль для моделювання функціональності оскільки вона наочно демонструє можливості системи та взаємодію користувачів певних ролей з системою електронної бібліотеки[10]. Результат моделювання наведено на рисунку 3.12

Також ця діаграма допомагає при розробці інтерфейсу користувача оскільки графічно зображує доступні дії всіх ролей та сприяє розуміння розробниками потреб всіх користувачів.

Крім того, вона сприяє комунікації між учасниками проекту, які не розуміють діаграму класів, оскільки для неї не потрібні додаткові знання. Вона показує можливості всіх користувачів без розкриття деталей та технічних подробиць.

На діаграмі ми бачимо, що незареєстрований користувач не має ніяких

прав крім реєстрації та авторизації. Тобто він навіть не може переглянути каталог книг.

Робота рекомендаційного алгоритму проявляє себе після бронювання книги, зареєстрованим користувачем. Після вдалого бронювання він має можливість переглянути список книг, які порекомендував йому алгоритм на основі його транзакції.

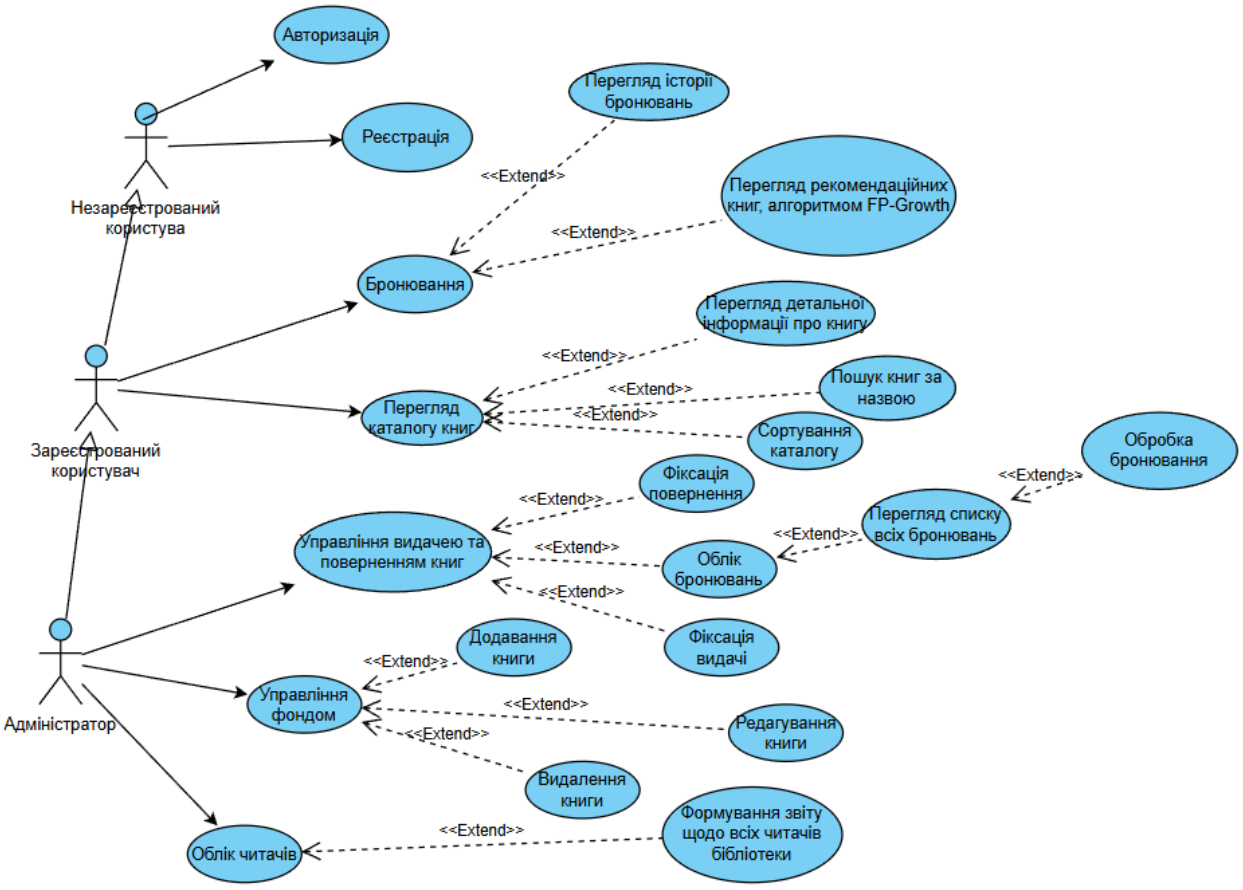



Рисунок 3.12 – UseCase діаграма

## 3.5 Розробка діаграми послідовності

Після USE CASE діаграми важливо розробити Sequence діаграму для методу «Бронювання книги»[11]. Це важливо для повноцінного аналізу та проектування системи електронної бібліотеки. Ця діаграма дозволяє зобразити послідовність функцій користувача з певною роллю для виконання конкретної

функції. Це дозволить повністю змоделювати певний процес від початку і до кінця.

На діаграмі послідовності будуть представлені всі об'єкти, які беруть участь у виконанні методу "Бронювання книги", такі як користувач, контролер книг, бронювання, тощо. Діаграма відображатиме впорядковану послідовність повідомлень, що відправляються між цими об'єктами під час процесу бронювання книги.

Діаграма послідовності дії «Бронювання книги» наведена на рисунку 3.13

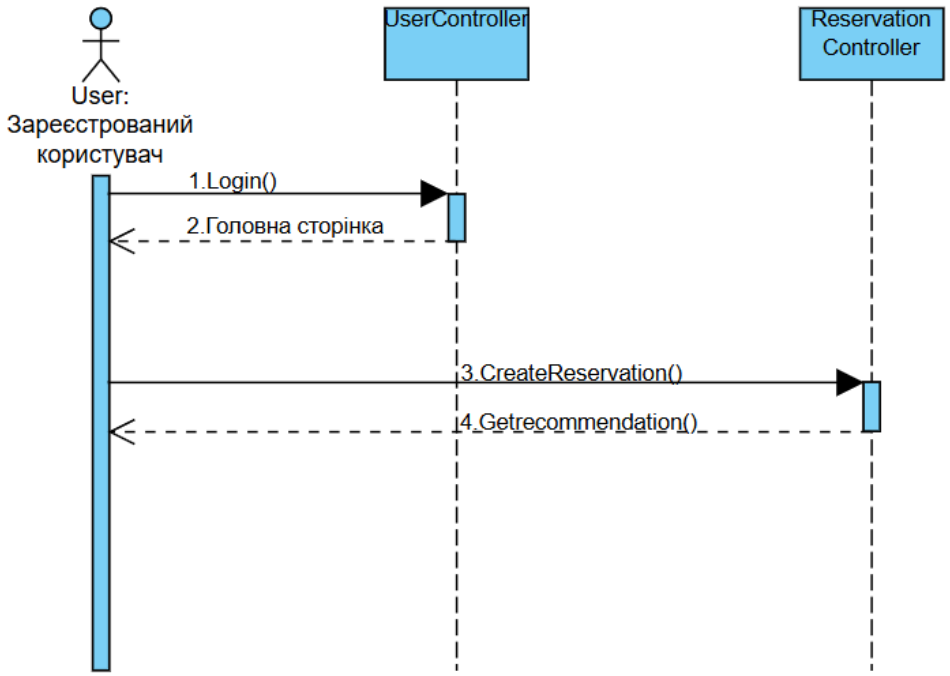



Рисунок 3.13 – Діаграма послідовності бронювання певної книги

## 3.6 Розробка ієрархії сторінок

Для відображення ієрархії сторінок розроблена модель веб-сайту для планування всіх можливих переходів між сторінками 3.14.


Рисунок 3.14 – Карта сайту

Вона допоможе правильно спроектувати зрозумілу логіку переходів між сторінками з урахуванням бізнес-функцій електронної бібліотеки. Це допоможе користувачам вільно знаходити необхідну інформацію та повноцінно використовувати всі можливості системи.

Ця модель відображає зв’язки між сторінками. Вона допоможе розробникам правильно зрозуміти вимоги користувачів та розробити зв’язки між сторінками так, щоб використання системи було інтуїтивно зрозумілим та зручним як для читачів, так і для адміністраторів. Це одна зі складових для забезпечення позитивного досвіду для всіх користувачів.

Ця діаграма дозолить виявити потенційні проблеми та складності в навігації по карті сайту з якими може стикнутися звичайний користувач або адміністратор. Якщо користувач не зможе використовувати систему через її складність, то він не буде повертатись до неї.

З головної сторінки можна перейти на будь-яку іншу. Зі сторінок каталогу та детальної інформації про книгу можна відразу перейти до бронювання, а після можна перейти до списку власних бронювань і переглянути статус.

Всього на ній зображено 8 сторінок але в дійсності їх більше, оскільки деякі з них об’єднані в спільні блоки. Це дозволяє спростити схему та

зосередитись на основних функціях та переходах між ними. За рахунок цього я забезпечив баланс між зрозумілістю та деталізацією моделі.

# 4 РОЗРОБКА СИСТЕМИ

## 4.1 Розробка логічної та фізичної моделі

Розробити логічну та фізичну модель БД[12]. У БД маємо зберігати інформацію, яка відповідає предметній області бібліотеки:

користувач - таблиця містить інформацію про користувачів. В ній записано інформацію про адміністраторів та звичайних користувачів;

роль користувача - таблиця відображає ролі, які доступні користувачам. Кожна роль може мати свій унікальний ідентифікатор і назву;

книга - таблиця зберігає інформацію про книги, доступні для бронювання або оформлення. Кожна книга може мати ідентифікатор, назву, автора, жанр та інші характеристики книги;

автор книги - таблиця містить інформацію про авторів книг. Кожен автор представлений у цій таблиці з унікальним ідентифікатором та інформацією про ім'я, прізвище та інші відомості;

інформація щодо бронювання книги - таблиця може зберігати дані про бронювання книг, такі як ідентифікатор бронювання, ідентифікатор користувача, ідентифікатор книги та іншу інформацію, пов'язану з бронюванням;

жанр книги - таблиця включає різні жанри книг. Кожен жанр може мати свій унікальний ідентифікатор і назву, що допомагає класифікувати книги за жанрами;

статус бронювання - таблиця відображає можливі стани бронювань книг. Кожен статус може мати ідентифікатор і назву, що дозволяє відстежувати стан кожного бронювання.

мова – таблиця відображає всі наявні мови в бібліотеці

видавництво – таблиця відображає видавництва, які випускають книг

обкладинка – таблиця відображає тип обкладинки(твердий, м'який)

рекомендаційні правила. Таблиця зберігає правила, знайдені алгоритмом пошуку асоціативних правил та їх довіру(confidence)

елементи правила. Таблиця зберігає книги, які є складовими того чи іншого правила. Зберігає як книги що є умовою і спричиняють рекомендації, так і ті, що є наслідком і рекомендуються.

Побудова логічної моделі здійснена в інструментальному засобі ERWin. Діаграму логічної моделі даних зображено на рисунку 3.1.

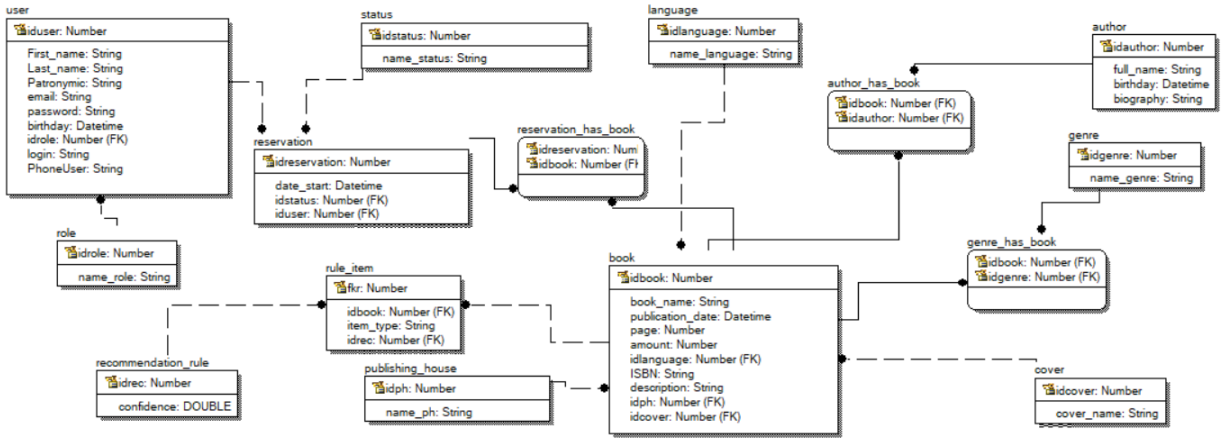



Рисунок 4.1 - Логічна модель даних

На логічній моделі наведено такі сутності та віповідні до них атрибути.

Сутність «role» - зберігає дані про те, які ролі можуть бути у користувача Сутність «user» - зберігає інформацію про користувачів

Сутність «book» - зберігає інформацію про книги

Сутність «genre» - зберігає інформацію про жанри, які можуть бути у книг Сутність «autor» - зберігає інформацію про авторів

Сутність «status» - зберігає інформацію про можливі статуси бронювання чи оформлення книг

Сутність «reservation» - зберігає інформацію про бронь книг

Сутність «publishing_house» - зберігає інформацію про видання

Сутність «language» - зберігає інформацію про мови

Сутність «cover» - зберігає інформацію про обкладинки книг

Сутність «recommendation_rule» - зберігає інформацію про асоціативні правила

Після розробки логічної моделі даних, потрібно розробити фізичну, для подальшого використання у СУБД MySQL[3].

На рисунку 4.2 зображено фізичку модель даних.

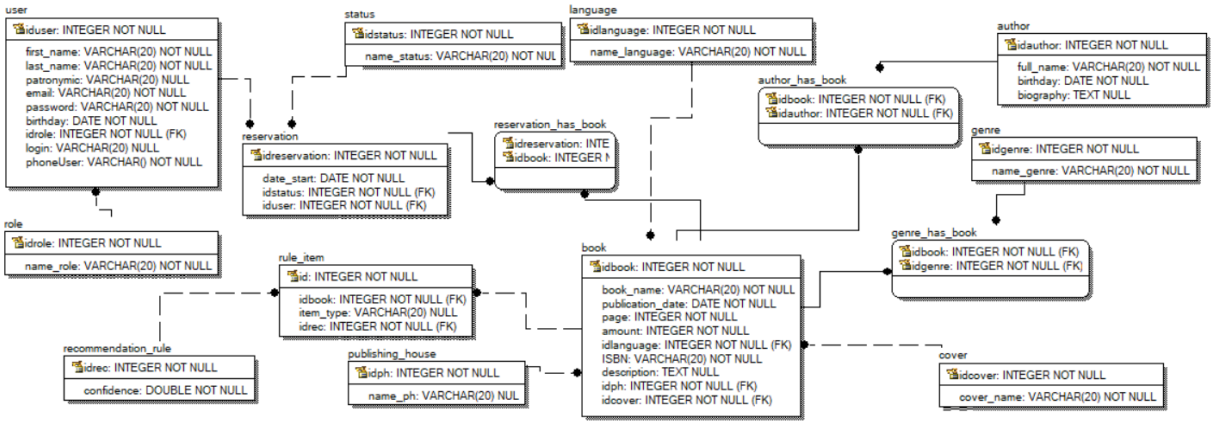



Рисунок 4.2 - Фізична модель даних

Сутність «role»:

атрибут «idrole» - ціле число. Це поле містить унікальний ідентифікатор для кожної ролі.

атрибут «name_role» - текстовий тип даних. Це поле містить назву ролі користувача в бібліотеці.

Сутність «user»:

атрибут «iduser» - ціле число. Це поле містить унікальний ідентифікатор для кожного користувача.

атрибут «first_name» - текстовий тип даних, це поле містить ім’я користувача

атрибут «last_name» - текстовий тип даних, це поле містить фамілію користувача

атрибут «patronymic» - текстовий тип даних, це поле містить по батькові користувача

атрибут «email» - текстовий тип даних. Це поле містить адресу електронної пошти користувача.

атрибут «login» - текстовий тип даних. Це поле містить логін користувача.

атрибут «password» - текстовий тип даних. Це поле містить пароль користувача.

атрибут «bithday» - дата Це поле містить дату народження користувача.

атрибут «role_idrole» - ціле число. Це поле містить ідентифікатор ролі користувача у бібліотеці.

атрибут «phoneuser» - текстовий тип даних. Це поле містить інформацію про телефон користувача;

Сутність «book»:

атрибут «idbook» - ціле число. Це поле містить унікальний ідентифікатор для кожної книги в бібліотеці.

атрибут «book_name» - текстовий тип даних. Це поле містить назву книги.

атрибут «publication_date» - дата Це поле містить дату публікації книги.

атрибут «ISBN» - текстовий тип. Це поле містить міжнародний стандартний номер книги

атрибут «page» - ціле число. Це поле містить кількість сторінок у книзі.

атрибут «amount» - ціле число. Це поле містить кількість доступних книг в бібліотеці.

атрибут «description» - текстовий тип даних. Це поле містить опис книги.

Сутність «genre»:

атрибут «idgenre» - ціле число. Це поле містить унікальний ідентифікатор для кожного жанру.

атрибут «name_genre»» - текстовий тип даних. Містить назву жанру.

Сутність «genre_has_book»:

атрибут «genre_idgenre» - ціле число. Це поле містить ідентифікатор

жанру, який пов'язаний з певною книгою.

атрибут «book_idbook» - ціле число. Це поле містить ідентифікатор книги, яка має певний жанр.

Сутність «author»:

атрибут «idauthor» - ціле число. Це поле містить унікальний ідентифікатор для кожного автора.

атрибут «full_name» - текстовий тип даних, це поле містить ім’я ПІБ автора

атрибут «biography» - текстовий тип даних. Це поле містить біографію автора.

Сутність «autor_has_book»:

атрибут «autor_idauthor» - ціле число. Це поле містить ідентифікатор автора, який пов'язаний з певною книгою.

атрибут «book_idbook» - ціле число. Це поле містить ідентифікатор книги, яка має певного автора.

Сутність «status»:

атрибут «idstatus» - ціле число. Це поле містить унікальний ідентифікатор для кожного статусу.

атрибут «name_status» - текстовий тип даних. Це поле містить назву статусу.

Сутність «reservation»:

атрибут «idreservation» - ціле число. Це поле містить унікальний ідентифікатор для кожного запису в таблиці резервацій.

атрибут «date_reserv» - дата Це поле містить дату початку бронювання.

атрибут «user_iduser» - ціле число. Це поле містить ідентифікатор користувача, який зробив резервацію.

атрибут «book_idbook» - ціле число. Це поле містить ідентифікатор книги, яка була зарезервована.

атрибут «status_idstatus» - ціле число. Це поле містить ідентифікатор статусу резервації.

Сутність «language»:

атрибут «idlanguage» - ціле число. Це поле містить ідентифікатор мови.

атрибут «name_language» - текстовий тип даних. Це поле містить назву мови

Сутність «cover»:

атрибут «idcover» - ціле число. Це поле містить ідентифікатор обкладинки.

атрибут «cover_name» - ціле число. Це поле містить назву обкладинки.

Сутність «publishing_house»:

атрибут «idph» - ціле число. Це поле містить ідентифікатор видавництва.

атрибут «name_ph» - текстовий тип даних. Це поле містить назву видавництва.

Сутність «recommendation_rule»:

атрибут «idrec» - ціле число. Це поле містить ідентифікатор правила.

атрибут «confidence» - число з плаваючою комою. Це поле містить значення довіри.

Сутність «rule_item»:

атрибут «idbook» - ціле число. Це поле містить зовнішній ключ книги.

атрибут «idrec» - ціле число. Це поле містить зовнішній ключ правила.

атрибут «id» - ціле число. Це поле містить ідентифікатор елемента.

атрибут «item_type» - текстове поле. Це поле містить тип правила. Для реалізації в СУБД буде використано тип даних enum

## 4.2 Створення бази даних на платформі СУБД

СУБД для ІС бібліотеки обрано MySQl. На рисунку 3.3 зображено EER-модель БД у СУБД MySQL[13].

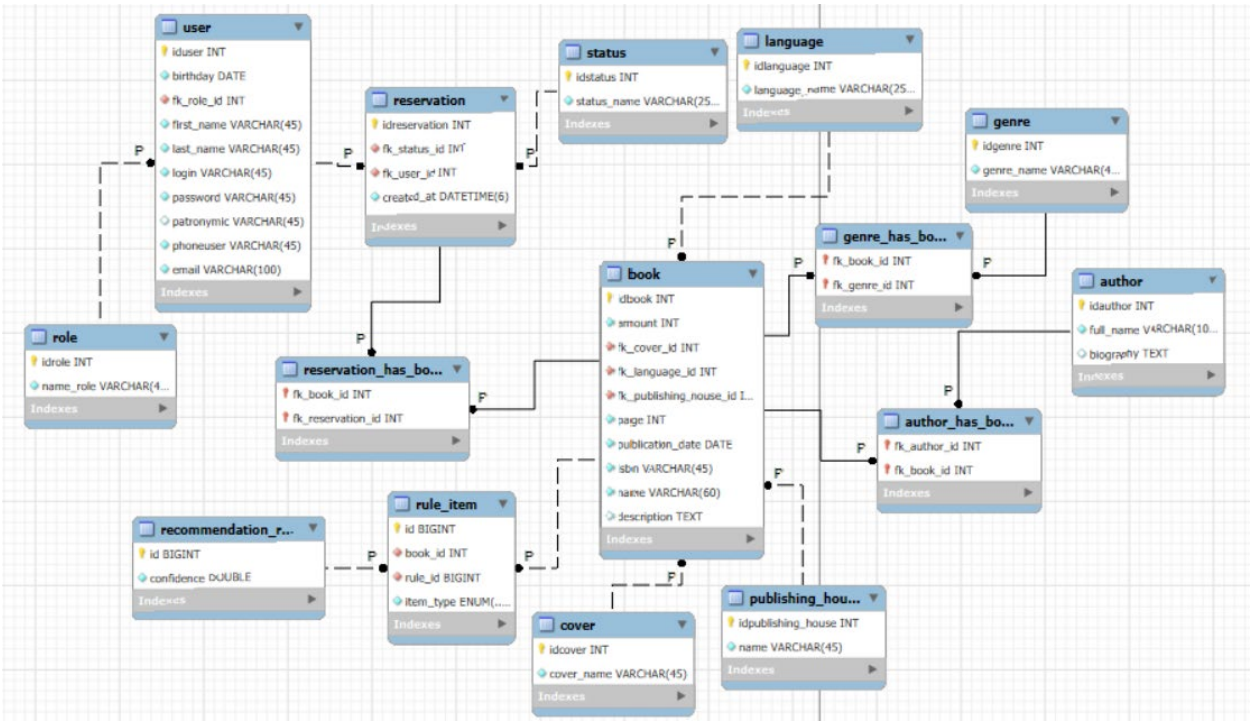



Рисунок 4.3 - EER-модель БД у СУБД MySQL

В таблиці 4.1 описані зв’язки між таблицями бази даних.

Таблиця 4.1 – Опис зв’язків між таблицями в базі даних

| № | Ім’я таблиці 1, зовнішній ключ | Ім’я таблиці 2, зовнішній ключ | Тип посилальної цілісності для інструкції UPDATE | Тип посилальної цілісності для інструкції DELETE |
|---|---|---|---|---|
| 1 | user, fk_idrole | role, idrole | CASCADE | RESTRIC |
| 2 | Reservation, fk_idbook | Book, idbook | RESTRIC | RESTRIC |
| 3 | Reservation, fk_iduser | User, iduser | RESTRIC | RESTRIC |
| 4 | Reservation, fk_idstatus | Status, idstatus | CASCADE | RESTRIC |

| | | | | |
|---|---|---|---|---|
| 5 | Book, fk_idlanguage | Language, idlanguage | CASCADE | RESTRIC |
| 6 | Book, fk_idcover | Cover, idcover | CASCADE | RESTRIC |
| 7 | Book, ph_idph | Publishing_hou se, idph | CASCADE | SET NULL |
| 8 | Genre_has_book, fk_idbook | Book, idbook | CASCADE | CASCADE |
| 9 | Genre_has_book, fk_idgenre | Genre, idgenre | CASCADE | CASCAD |
| 10 | Author_has_book, fk_idbook | Book, idbook | CASCADE | CASCADE |
| 11 | Author_has_book, fk_idauthor | Author, idauthor | CASCADE | CASCAD |
| 12 | Rule_item, fk_idrec | Reccomendation_rule, idrec | CASCADE | CASCADE |
| 13 | Rule_item, fk_idbook | Book, idbook | CASCADE | CASCADE |

## 4.3 Розробка системних вимог до системи

Апаратні вимоги для серверу, необхідного для запуску та підтримки інформаційної системи електронної бібліотеки, що використовує Spring Boot MySQL.

Процесор. Мінімум Intel Core i3 або еквівалентний AMD процесор.

Рекомендовано Intel Core i5 або вище для кращої продуктивності.

Оперативна пам'ять (RAM). Мінімум 4 ГБ RAM. Рекомендовано 8 ГБ або більше для оптимальної продуктивності та швидкодії системи.

## 4.4 Розробка уявлень та sql запитів серверної частини системи

Для виконання функції «Звіт про бронювання» розроблено уявлення «view_user_book» Воно демонструє всю важливу інформацію про те, хто бронював конкретну книгу, а також подробиці цього бронювання. Код для створення уявлення наведено нижче.

```
CREATE OR REPLACE VIEW `view_user_book` AS
SELECT
    user.iduser AS `iduser`,
    user.first_name as`first_name`,
    user.last_name as`last_name`,
    user.patronymic as`patronymic`,
    user.email as`email`,
    user.Phoneuser as`Phoneuser`,
    book.idbook As `idbook`,
    book.name AS `nameBook`,
    reservation.date_reserv AS `datereserv`,
    reservation.date_end_reserv AS `dateEndreserv`,
    reservation.use AS `use`,
    status.name_status AS `nameStatus`
FROM user
    JOIN reservation  ON reservation.user_iduser = user.iduser
    JOIN book  ON reservation.book_idbook = book.idbook
    JOIN status  ON status.idstatus = reservation.status_idstatus
```

На русунку 4.4 зображені дані, які виводить уявлення «view_user_book»

| iduser | first_name | last_name | patronymic | email | Phoneuser | idbook | nameBook | datereserv | dateEndreserv | use | nameStatus |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Іван | Петренко | Олексійович | ivan.petrenko@example.com | +380951234567 | 1 | Гіперіон | 2023-06-15 | 2023-06-30 | 1 | завершено |
| 3 | Михайло | Коваль | Вікторович | mikhail.koval@example.com | +380997654321 | 2 | Фах | 2024-05-01 | 2024-05-15 | 1 | підтверджено |
| 2 | Олена | Іванова | Петрівна | elena.ivanova@example.com | +380671234567 | 3 | Двері до літа | 2024-05-02 | 2024-05-16 | 1 | підтверджено |
| 12 | Тетяна | Сидоренко | Ігорівна | tetyana.sydorenko@example.com | +380668945123 | 4 | Солярис | 2023-05-03 | 2023-05-17 | 1 | завершено |
| 7 | Ігор | Ковальчук | Олексійович | igor.kovalchuk@example.com | +380667894512 | 5 | Володар Перснів | 2024-04-04 | 2024-04-18 | 1 | просрочено |
| 6 | Анастасія | Бойко | Олександрівна | anastasiya.boiko@example.com | +380931234567 | 6 | Гобіт | 2024-05-05 | 2024-05-19 | 0 | новий |

Рисунок 4.4 – Звіт про бронювання

Також розроблено sql-запити[14] для відстеження статусу бронювання. Роботу запитів зображено на рисунку 4.5-4.7

```
select * from view_user_book where nameStatus = 'завершено'
```

| iduser | first_name | last_name | patronymic | email | Phoneuser | idbook | nameBook | datereserv | dateEndreserv | use | nameStatus |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Іван | Петренко | Олексійович | ivan.petrenko@example.com | +380951234567 | 1 | Гіперіон | 2023-06-15 | 2023-06-30 | 1 | завершено |
| 12 | Тетяна | Сидоренко | Ігорівна | tetyana.sydorenko@example.com | +380668945123 | 4 | Соляріс | 2023-05-03 | 2023-05-17 | 1 | завершено |

Рисунок 4.5 – Запит для формування звіту про бронювання зі статусом «завершено»

```
select * from view_user_book where nameStatus = 'просрочено'
```

| iduser | first_name | last_name | patronymic | email | Phoneuser | idbook | nameBook | datereserv | dateEndreserv | use | nameStatus |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 7 | Ігор | Ковальчук | Олексійович | igor.kovalchuk@example.com | +380667894512 | 5 | Володар Перснів | 2024-04-04 | 2024-04-18 | 1 | просрочено |

Рисунок 4.6 – Запит для формування звіту про бронювання зі статусом «просрочено»

```
select * from view_user_book where nameStatus = 'підтверджено'
```

| iduser | first_name | last_name | patronymic | email | Phoneuser | idbook | nameBook | datereserv | dateEndreserv | use | nameStatus |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | Михайло | Коваль | Вікторович | mikhail.koval@example.com | +380997654321 | 2 | Фах | 2024-05-01 | 2024-05-15 | 1 | підтверджено |
| 2 | Олена | Іванова | Петрівна | elena.ivanova@example.com | +380671234567 | 3 | Двері до літа | 2024-05-02 | 2024-05-16 | 1 | підтверджено |

Рисунок 4.7 – Запит для формування звіту про бронювання зі статусом «підтверджено»

Для розробки звіту для читачів розроблено уявлення «view_user». На рисунку 4.8 зображено дані які він виводить. Серед них всі дані с таблиці «user» крім паролю та назва ролі.

```
CREATE VIEW `view_user` AS
SELECT
   user.iduser as `iduser`,
   user.first_name as`first_name`,
   user.last_name as`last_name`,
   user.patronymic as`patronymic`,
   user.email as`email`,
   user.login as`login`,
   user.birthday as`birthday`,
   user.Phoneuser as`Phoneuser`,
   role.name_role as `name_role`
   from user join  role on user.role_idrole = role.idrole
```

| iduser | first_name | last_name | patronymic | email | login | birthday | Phoneuser | name_role |
|---|---|---|---|---|---|---|---|---|
| 1 | Іван | Петренко | Олексійович | ivan.petrenko@example.com | ivan | 1990-05-15 | +380951234567 | Адміністратор |
| 2 | Олена | Іванова | Петрівна | elena.ivanova@example.com | elena | 1985-08-20 | +380671234567 | Користувач |
| 3 | Михайло | Коваль | Вікторович | mikhail.koval@example.com | mikhail | 1993-11-10 | +380997654321 | Користувач |
| 4 | Наталія | Сидоренко | Володимирівна | nataliya.sydorenko@example.com | nataliya | 1988-03-25 | +380987654321 | Користувач |
| 5 | Олександр | Ковальчук | Миколайович | oleksandr.kovalchuk@example.com | oleksandr | 1992-06-18 | +380936547812 | Користувач |
| 6 | Анастасія | Бойко | Олександрівна | anastasiya.boiko@example.com | anastasiya | 1990-02-14 | +380931234567 | Користувач |
| 7 | Ігор | Ковальчук | Олексійович | igor.kovalchuk@example.com | igor | 1994-09-30 | +380667894512 | Користувач |
| 8 | Марія | Петренко | Анатоліївна | maria.petrenko@example.com | maria | 1987-12-08 | +380677894512 | Користувач |
| 9 | Володимир | Сидоренко | Вікторович | volodymyr.sydorenko@example.com | volodymyr | 1991-07-22 | +380631234567 | Користувач |
| 10 | Катерина | Іванова | Олексіївна | kateryna.ivanova@example.com | kateryna | 1989-04-12 | +380677894512 | Користувач |
| 11 | Євген | Петренко | Андрійович | yevhen.petrenko@example.com | yevhen | 1995-11-05 | +380671234567 | Користувач |

Рисунок 4.8 – Звіт про користувачів

Для перегляду адміністраторів створений запит. Його робота зображена на рисунку 4.9

```
select * from view_user where name_role = 'Адміністратор'
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

| iduser | first_name | last_name | patronymic | email | login | birthday | Phoneuser | name_role |
|---|---|---|---|---|---|---|---|---|
| 1 | Іван | Петренко | Олексійович | ivan.petrenko@example.com | ivan | 1990-05-15 | +380951234567 | Адміністратор |

Рисунок 4.9 – Запит для формування звіту про адміністраторів

Результат запиту для перегляду звичайних користувачів наведено на рисунку 4.10

```
select *  from view_user where name_role = "Користувач"
```

| iduser | first_name | last_name | patronymic | email | login | birthday | Phoneuser | name_role |
|---|---|---|---|---|---|---|---|---|
| 2 | Олена | Іванова | Петрівна | elena.ivanova@example.com | elena | 1985-08-20 | +380671234567 | Користувач |
| 3 | Михайло | Коваль | Вікторович | mikhail.koval@example.com | mikhail | 1993-11-10 | +380997654321 | Користувач |
| 4 | Наталія | Сидоренко | Володимирівна | nataliya.sydorenko@example.com | nataliya | 1988-03-25 | +380987654321 | Користувач |
| 5 | Олександр | Ковальчук | Миколайович | oleksandr.kovalchuk@example.com | oleksandr | 1992-06-18 | +380936547812 | Користувач |
| 6 | Анастасія | Бойко | Олександрівна | anastasiya.boiko@example.com | anastasiya | 1990-02-14 | +380931234567 | Користувач |
| 7 | Ігор | Ковальчук | Олексійович | igor.kovalchuk@example.com | igor | 1994-09-30 | +380667894512 | Користувач |
| 8 | Марія | Петренко | Анатоліївна | maria.petrenko@example.com | maria | 1987-12-08 | +380677894512 | Користувач |
| 9 | Володимир | Сидоренко | Вікторович | volodymyr.sydorenko@example.com | volodymyr | 1991-07-22 | +380631234567 | Користувач |
| 10 | Катерина | Іванова | Олексіївна | kateryna.ivanova@example.com | kateryna | 1989-04-12 | +380677894512 | Користувач |
| 11 | Євген | Петренко | Андрійович | yevhen.petrenko@example.com | yevhen | 1995-11-05 | +380671234567 | Користувач |
| 12 | Тетяна | Сидоренко | Ігорівна | tetyana.sydorenko@example.com | tetyana | 1993-08-28 | +380668945123 | Користувач |

Рисунок 4.10 - Запит для формування звіту про користувачів

## 4.5 Розробка інтерфейсу для функцій інформаційної системи

Відповідно до спроектованої карти сайту розроблено інтерфейс користувача. Для розробки структури і стилю сторінок використана мова HTML та CSS[14]. Для динамічних елементів використовується JS[15].

Спочатку розроблено сторінки реєстрації та авторизації, вони зображені на рисунках 4.11 – 4.12

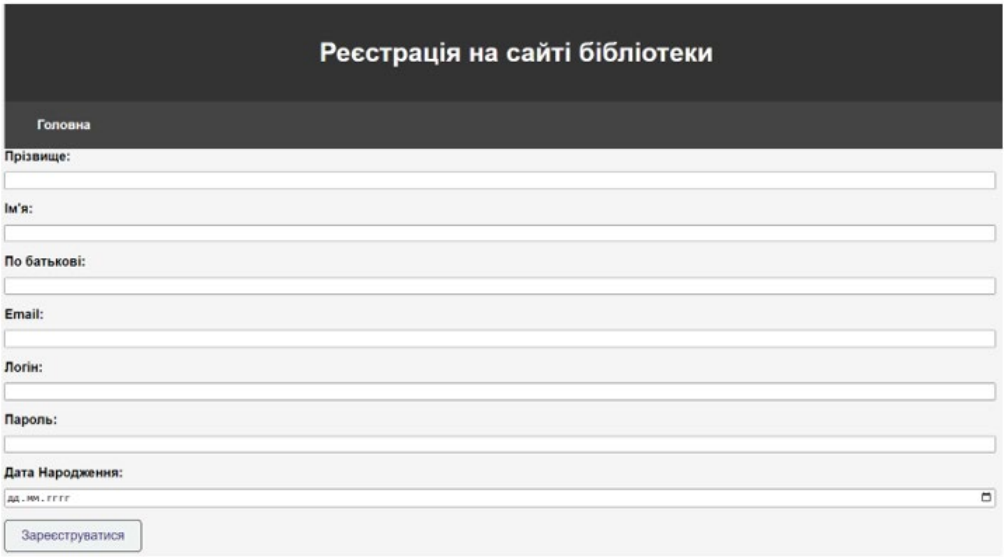



Рисунок 4.11 - Сторінка реєстрації

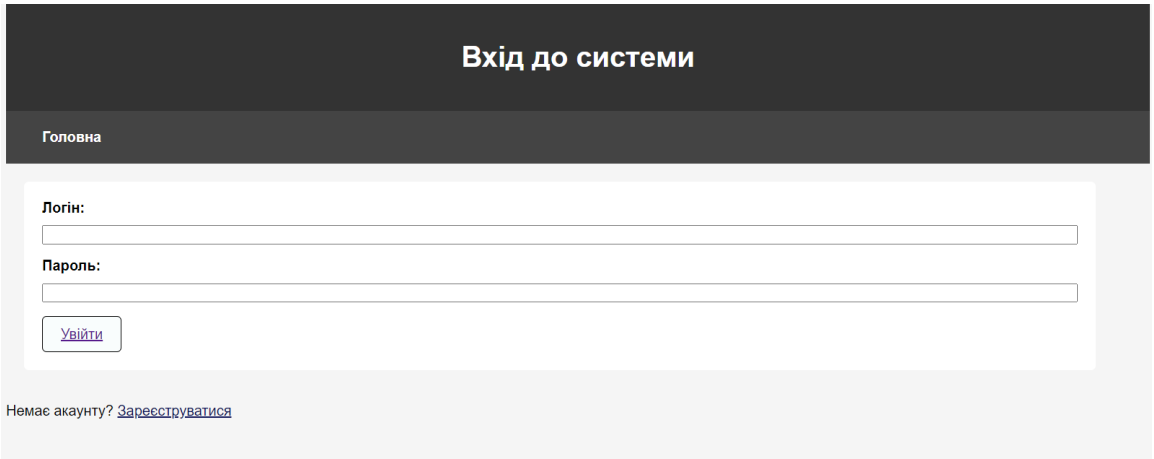



Рисунок 4.12 - Сторінка авторизації

Після авторизації в систему, користувач має перейти до головної сторінки, з якої він отримає доступ до навігації на інші сторінки відповідно до його ролі. Головна сторінка зображена на рисунку 4.13

На ній зображені ініціали користувача а також його пошта та логін, щоб він їх не забував.

На панелі навігації написані всі доступні сторінки користувачу. В даному випадку це головна сторінка адміністратора, якому, окрім базового функціоналу, доступні також сторінки додавання, видалення та редагування книг, а також сторінки створення звітів.

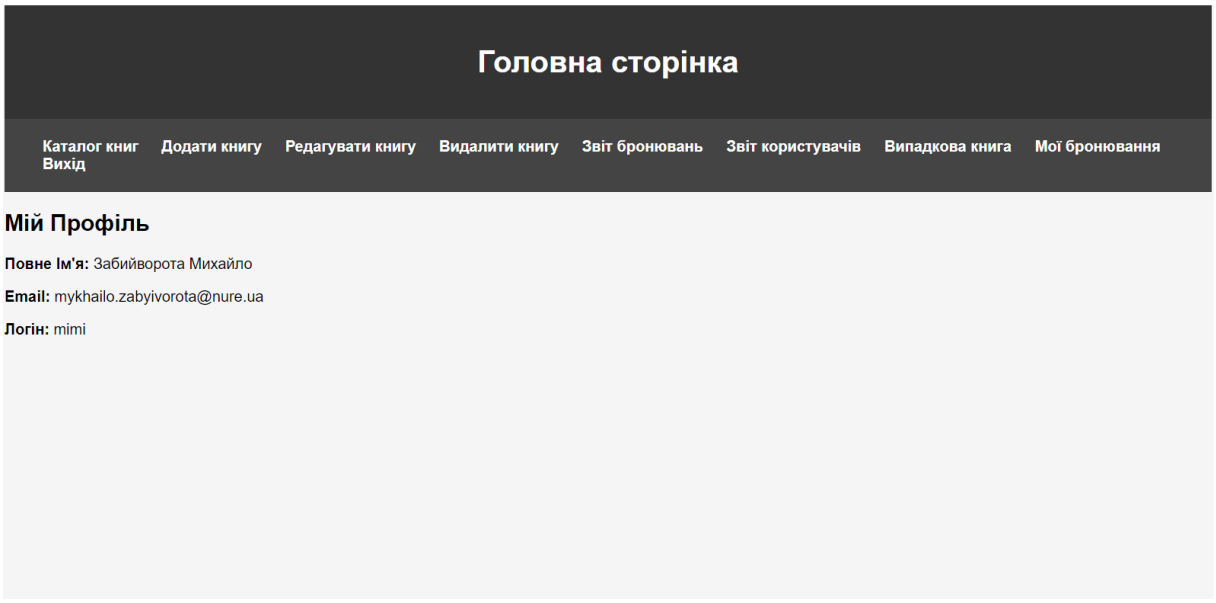



Рисунок 4.13 - Головна сторінка

Для адміністратора розроблено сторінку для перегляду звіту бронювання, яка зображена на рисунку 4.14. Йому доступний список всіх бронювань. В звіті відображається вся необхідна інформація про користувача, книгу та деталі бронювання.

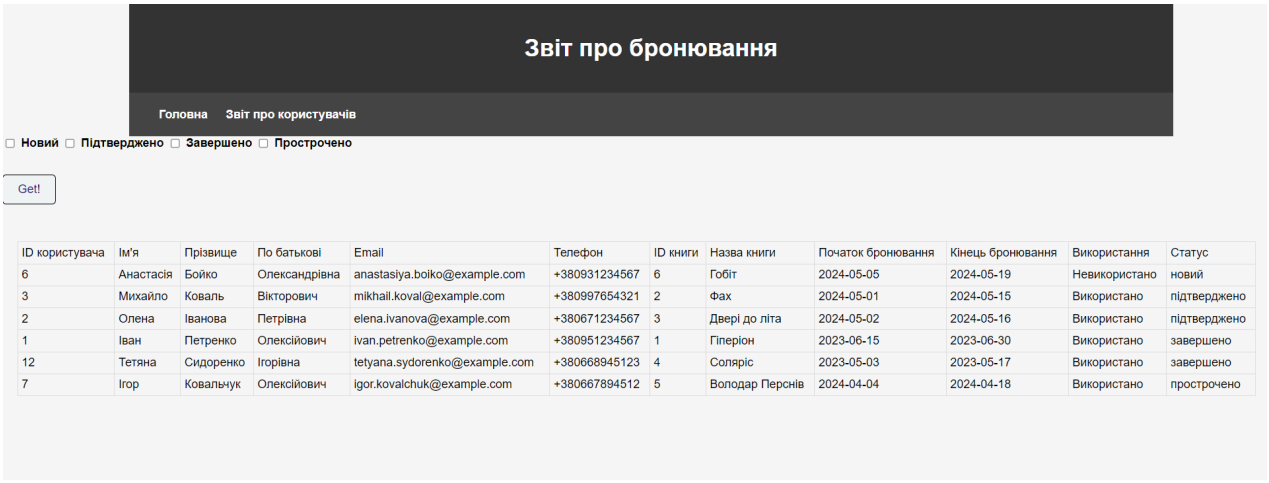



Рисунок 4.14 – Інтерфейс звіту про бронювання

За бажанням адміністратор може дивитися бронювання тільки з обраним статусом. Наприклад, тільки прострочені бронювання або підтверджені або вже прострочені. Роботу фільтрації звіту про бронювання зображено на рисунках 4.15 – 4.17

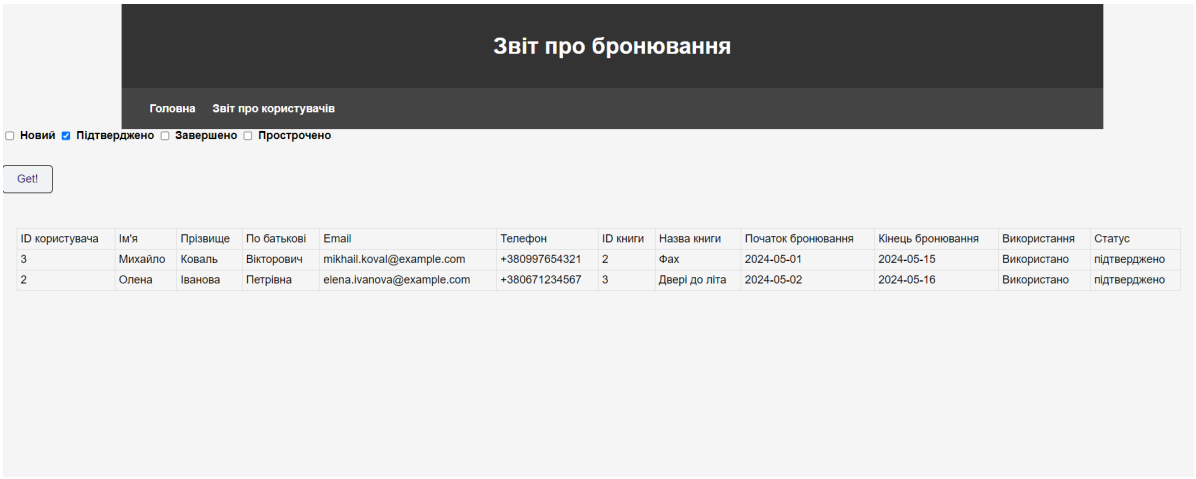



Рисунок 4.15 – Список бронювання зі статусом «підтверджено»

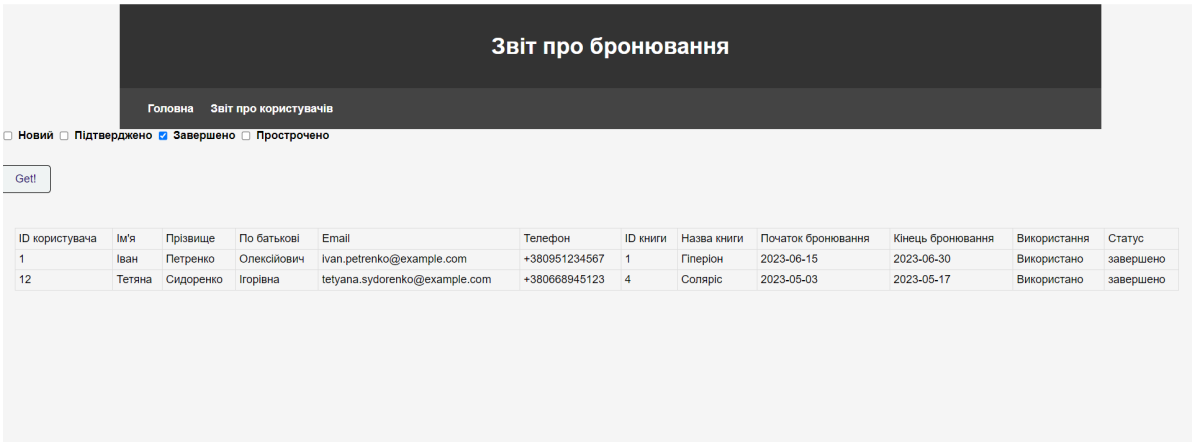



Рисунок 4.16 – Список бронювання зі статусом «завершено»

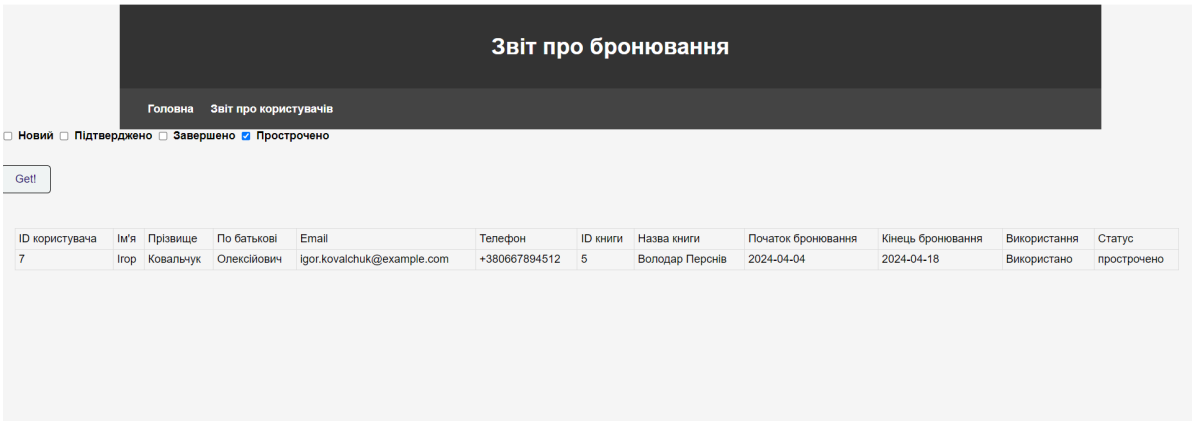
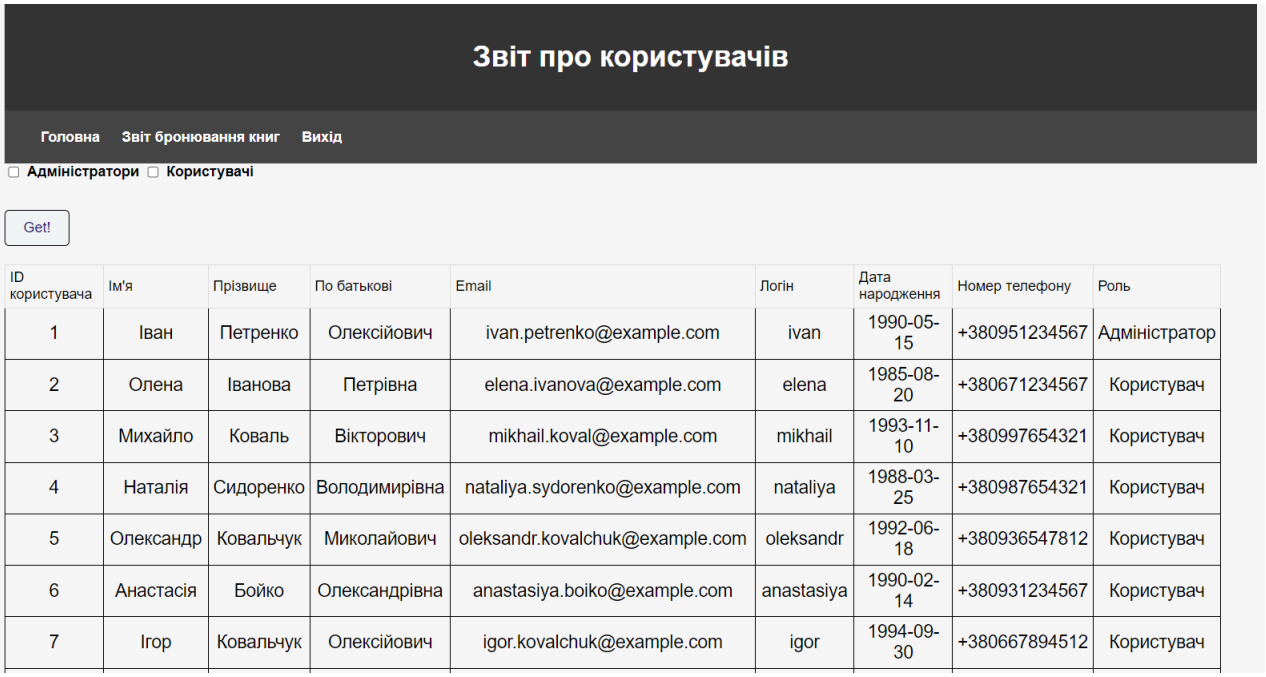



Рисунок 4.17 – Список бронювання зі статусом «прострочена»

Це допоможе відстежувати всі бронювання в системі, і в випадку, якщо якась із них прострочена, зателефонувати або написати на пошту боржнику.

Також для адміністратора розроблена окрема сторінка для перегляду звіту про користувачів. У виді таблиці йому доступна вся необхідна інформація про користувачів. А саме id, ім’я, прізвище, по батькові, електронна пошта, логін, дата народження, номер телефону та роль. Інтерфейс звіту про користувачів зображено на рисунку 4.18




| ID користувача | Ім'я | Прізвище | По батькові | Email | Логін | Дата народження | Номер телефону | Роль |
|---|---|---|---|---|---|---|---|---|
| 1 | Іван | Петренко | Олексійович | ivan.petrenko@example.com | ivan | 1990-05-15 | +380951234567 | Адміністратор |
| 2 | Олена | Іванова | Петрівна | elena.ivanova@example.com | elena | 1985-08-20 | +380671234567 | Користувач |
| 3 | Михайло | Коваль | Вікторович | mikhail.koval@example.com | mikhail | 1993-11-10 | +380997654321 | Користувач |
| 4 | Наталія | Сидоренко | Володимирівна | nataliya.sydorenko@example.com | nataliya | 1988-03-25 | +380987654321 | Користувач |
| 5 | Олександр | Ковальчук | Миколайович | oleksandr.kovalchuk@example.com | oleksandr | 1992-06-18 | +380936547812 | Користувач |
| 6 | Анастасія | Бойко | Олександрівна | anastasiya.boiko@example.com | anastasiya | 1990-02-14 | +380931234567 | Користувач |
| 7 | Ігор | Ковальчук | Олексійович | igor.kovalchuk@example.com | igor | 1994-09-30 | +380667894512 | Користувач |

Рисунок 4.18 – Звіт про користувачів

Адміністратор може фільтрувати користувачів за ролями. Наприклад, переглянути тільки адміністраторів або тільки звичайних користувачів. Звіти, , який дає список всіх адміністраторів в системі зображено на рисунку 4.19

Звіт, який дозволяє переглянути тільки звичайних користувачів, тобто читачів бібліотеки, зображено на рисунку 4.20.

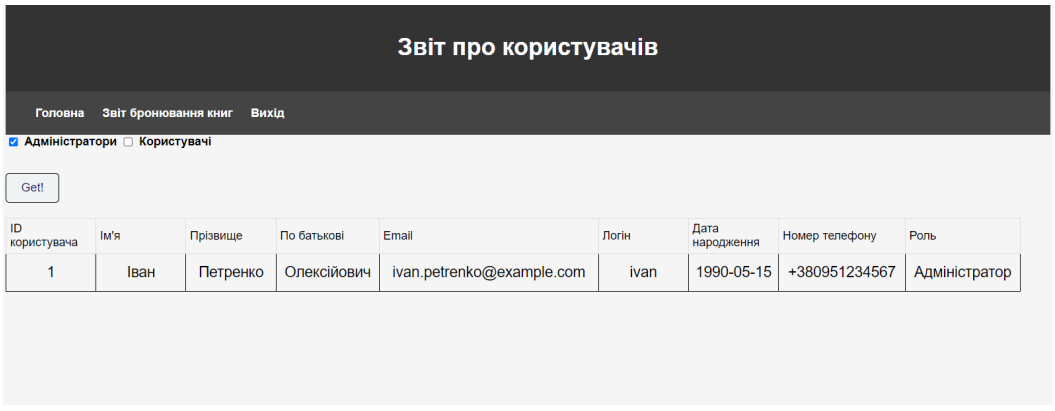
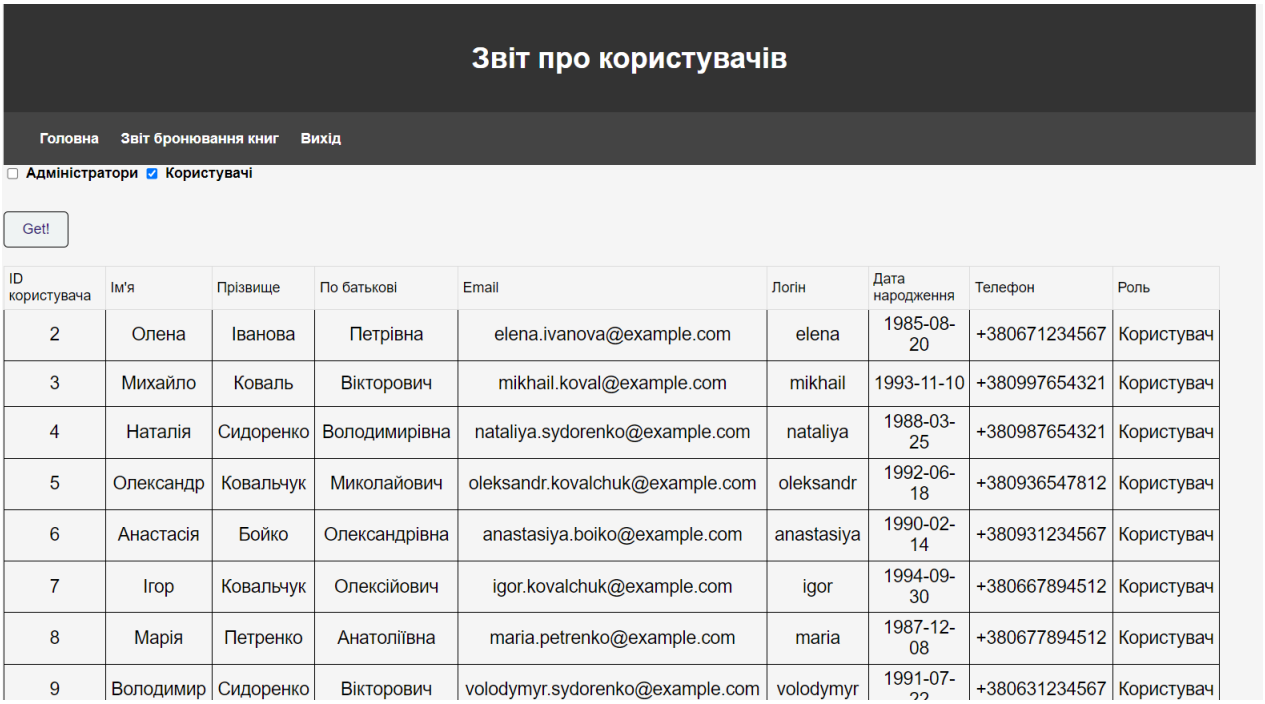



Рисунок 4.19 – Звіт про користувачів(адміністратори)




| ID користувача | Ім'я | Прізвище | По батькові | Email | Логін | Дата народження | Телефон | Роль |
|---|---|---|---|---|---|---|---|---|
| 2 | Олена | Іванова | Петрівна | elena.ivanova@example.com | elena | 1985-08-20 | +380671234567 | Користувач |
| 3 | Михайло | Коваль | Вікторович | mikhail.koval@example.com | mikhail | 1993-11-10 | +380997654321 | Користувач |
| 4 | Наталія | Сидоренко | Володимирівна | nataliya.sydorenko@example.com | nataliya | 1988-03-25 | +380987654321 | Користувач |
| 5 | Олександр | Ковальчук | Миколайович | oleksandr.kovalchuk@example.com | oleksandr | 1992-06-18 | +380936547812 | Користувач |
| 6 | Анастасія | Бойко | Олександрівна | anastasiya.boiko@example.com | anastasiya | 1990-02-14 | +380931234567 | Користувач |
| 7 | Ігор | Ковальчук | Олексійович | igor.kovalchuk@example.com | igor | 1994-09-30 | +380667894512 | Користувач |
| 8 | Марія | Петренко | Анатоліївна | maria.petrenko@example.com | maria | 1987-12-08 | +380677894512 | Користувач |
| 9 | Володимир | Сидоренко | Вікторович | volodymyr.sydorenko@example.com | volodymyr | 1991-07-22 | +380631234567 | Користувач |

Рисунок 4.20 – Звіт про користувачів

## 4.6 Реалізація FP-Growth в інформаційній системі

Алгоритм FP-Growth реалізований у бібліотеці SPMF у класі AlgoFPGrowth. Для реалізації та імплементації в інформаційну систему використано мову Java[16], фреймворк Spring Boot[17] та версію SPMF для роботи кодом.

Використання бібліотеки дозволило абстрагуватися від деталей реалізації та оптимізації алгоритму. Натомість, основна увага зосереджена на підготовці даних та обробці вихідних результатів. Це значно пришвидшило розробку.

Отримані результати в виді асоціативних правил зберігаються в базі даних разом з деякими важливими метриками. Це забезпечує розділення процесу навчання від процесу надання рекомендацій[17].

При обробці нового бронювання система спочатку шукає рекомендаційні правила на основі книг, щойно взятих користувачем. Якщо

правил для певного випадку не знайдено, система автоматично переходить до альтернативного методу. У цьому випадку користувачу будуть запропоновані книги, які належать до найбільш поширеного жанру в поточній транзакції.

Така архітектура гарантує, що система здатна пропонувати доцільні книги для нових користувачів та рідкісних комбінацій, де асоціативні правила ще не сформувалися.

# ВИСНОВКИ

Під час реалізації проекту проведено аналіз рекомендаційних алгоритмів, завдяки чому обґрунтовано обрано метод генерації асоціативних правил FP-Growth та розроблено систему електронної бібліотеки з використанням сучасних технологій.

Створено систему для оформлення бронювання книг та інтегровано в неї модуль для пошуку асоціативних правил та рекомендування на їх основі пропозицій користувачам. Проектування системи включало аналіз предметної області, створення функціональної моделі, логічної та фізичної моделей даних, а також розробку програмного інструменту для комплексування компонентів системи.

Створена система пройшла всі етапи проектування та розробки, забезпечуючи інтеграцію з базою даних та забезпечуючи зручний користувацький інтерфейс для взаємодії з клієнтами бібліотеки. Завдяки використанню сучасних методів проектування та розробки програмного забезпечення, система дозволить бібліотеці покращити якість обслуговування та оптимізувати процеси роботи з користувачами.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008-2015. Документація. Звіти в сфері науки і техніки. Структура і правила оформлювання. Чинний від 2017-07-01. – Київ : ДП «УкрНДНЦ», 2016. – 26 с.

2. ДСТУ 7.1:2006. Система стандартів з інформації, бібліотечної та видавничої справи. Бібліографічний запис. Бібліографічний опис. Загальні вимоги та правила складання / Нац. стандарт України. – Вид. офіц. – [Чинний від 2007-07-01]. – Київ : Держспоживстандарт України, 2007. – 47 с.

3. Association rules[Електронний ресурс] – Режим доступу до ресурсу: https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/association_rules/

4. Apriori Algorithm[Електронний ресурс] – Режим доступу до ресурсу: https://www.geeksforgeeks.org/machine-learning/apriori-algorithm/

5. FP-Growth Algorithm[Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/image-processing-with-python/fp-growth-algorithm-in-data-mining-e1064accf6a3

6. ECLAT Algorithm[Електронний ресурс] – Режим доступу до ресурсу: https://www.geeksforgeeks.org/machine-learning/ml-eclat-algorithm/

7. SPMF Documentation[Електронний ресурс] – Режим доступу до ресурсу: https://www.philippe-fournier-viger.com/spmf/

8. Нотація IDF0[Електронний ресурс] – Режим доступу до ресурсу: https://elib.lntu.edu.ua/sites/default/files/elib_upload/%D0%9A%D0%BE%D0%BD%D0%B4%D1%96%D1%83%D1%81%202%20%D0%B3%D0%BE%D1%82%D0%BE%D0%B2%D0%B2%D0%B0/page9.html

9. Нотація DFD[Електронний ресурс] – Режим доступу до ресурсу: https://elib.lntu.edu.ua/sites/default/files/elib_upload/%D0%9A%D0%BE%D0%BD%D0%B4%D1%96%D1%83%D1%81%202%20%D0%B3%D0%BE%D1%82%D0%BE%D0%B2%D0%B2%D0%B0/page18.html

10. What is USE CASE Diagram[Електронний ресурс] – Режим доступу до ресурсу: https://www.geeksforgeeks.org/system-design/use-case-diagram/

11. What is Sequence Diagram[Електронний ресурс] – Режим доступу до ресурсу: https://developer.ibm.com/articles/the-sequence-diagram/

12. What is Sequence Diagram[Електронний ресурс] – Режим доступу до ресурсу: https://www.ibm.com/think/topics/relational-databases

13. MySQL Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://dev.mysql.com/doc/

14. SQL tutorial [Електронний ресурс] – Режим доступу до ресурсу: https://www.geeksforgeeks.org/sql/sql-tutorial/

15. Дженніфер Робінсон, "HTML5 and CSS3 All-in-One For Dummies", 2014. – 1104 с.

16. Девід Фленаган, "JavaScript: The Definitive Guide", 2020. – 706 с.

17. Java Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://docs.oracle.com/en/java/

18. Spring Boot Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://docs.spring.io/spring-boot/index.html

19. Забийворота М. А. Використання методів Data Mining для реалізації функції E-Business-системи бібліотеки. 28-й Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті»: зб. матеріалів форуму. Т. 6. Конференція «Інформаційні інтелектуальні системи» (м. Харків, 16-18 квітня 2024 р.). Харків, 2024. С. 892-893.

20. Ситніков Д.Е., Ситнікова П.Е., Тітов С.В., Тітова О.В. Фільтрація результуючого набору асоціативних правил з точки зору оцінки цікавості / Д.Е. Ситніков, П.Е. Ситнікова, С.В. Тітов, О.В. Тітова // Системи обробки інформації № 1(164) 2021, стр. 83-88. https://doi.org/10.30748/soi.2021.164.09

21. Дослідження методів CRM на базі генерації асоціативних правил та їх використання в інформаційній системі бібліотеки / Забийворота М.А., Тітов С.В. // Міжнародна науково-технічна конференція «Інформаційно-комунікаційні технології та кібербезпека (ІКТК-2025)». Харків, 4-5 грудня

2025 р. – ХНУРЕ, 2025р. – Ст. 81-82.