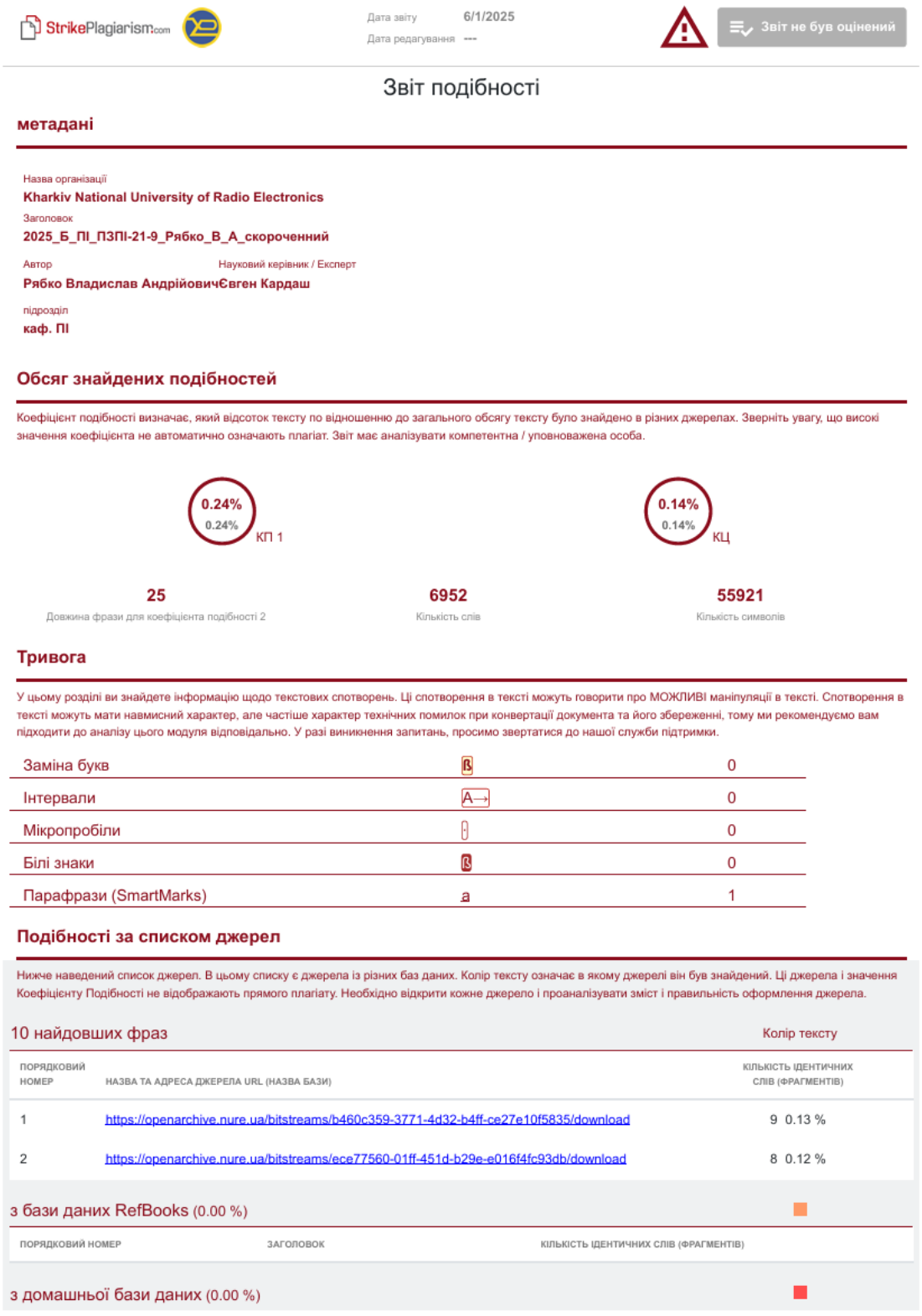
# ДОДАТОК А

## Результат перевірки на плагіат



StrikePlagiarism.com

Дата звіту 6/1/2025
Дата редагування ---

Звіт не був оцінений

### Звіт подібності

**метадані**

Назва організації
**Kharkiv National University of Radio Electronics**
Заголовок
**2025_Б_ПІ_ПЗПІ-21-9_Рябко_В_А_скороченний**
Автор
**Рябко Владислав Андрійович**
Науковий керівник / Експерт
**Євген Кардаш**
підрозділ
**каф. ПІ**

**Обсяг знайдених подібностей**

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

0.24% 0.24% КП 1

0.14% 0.14% КЦ

**25** Довжина фрази для коефіцієнта подібності 2

**6952** Кількість слів

**55921** Кількість символів

**Тривога**

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

| | | |
|---|---|---|
| Заміна букв | | 0 |
| Інтервали | | 0 |
| Мікропробіли | | 0 |
| Білі знаки | | 0 |
| Парафрази (SmartMarks) | | 1 |

**Подібності за списком джерел**

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

| ПОРЯДКОВИЙ НОМЕР | НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ) | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ) |
|---|---|---|
| 1 | https://openarchive.nure.ua/bitstreams/b460c359-3771-4d32-b4ff-ce27e10f5835/download | 9 0.13 % |
| 2 | https://openarchive.nure.ua/bitstreams/ece77560-01ff-451d-b29e-e016f4fc93db/download | 8 0.12 % |

з бази даних RefBooks (0.00 %)

| ПОРЯДКОВИЙ НОМЕР | ЗАГОЛОВОК | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ) |
|---|---|---|

з домашньої бази даних (0.00 %)

Рисунок А.1 – Перша сторінка перевірки на плагіат

| ПОРЯДКОВИЙ НОМЕР | ЗАГОЛОВОК | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ) |
|---|---|---|

з програми обміну базами даних (0.00 %)

| ПОРЯДКОВИЙ НОМЕР | ЗАГОЛОВОК | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ) |
|---|---|---|

з Інтернету (0.24 %)

| ПОРЯДКОВИЙ НОМЕР | ДЖЕРЕЛО URL | КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ) |
|---|---|---|
| 1 | https://openarchive.nure.ua/bitstreams/b460c359-3771-4d32-b4ff-ce27e10f5835/download | 9 (1) 0.13 % |
| 2 | https://openarchive.nure.ua/bitstreams/ece77560-01ff-451d-b29e-e016f4fc93db/download | 8 (1) 0.12 % |

**Список прийнятих фрагментів** (немає прийнятих фрагментів)

| ПОРЯДКОВИЙ НОМЕР | ЗМІСТ | КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ) |
|---|---|---|

1

ВСТУП
Сьогоднішній етап розвитку цифрових технологій має суттєвий вплив на освітні та культурні установи, серед яких особливе місце займають бібліотеки. У зв'язку зі зростанням кількості користувачів та збільшенням обсягу інформаційних ресурсів, усе актуальнішою стає потреба в автоматизації ключових процесів. Це стосується як обслуговування читачів, так і працівників бібліотеки.
З огляду на це, метою цієї роботи стало створення інтерактивного веб-застосунку для міської бібліотеки, який б поєднував зручність використання, гнучку структуру ролей та можливість масштабування. Система передбачає як функціонал для читачів – пошук і бронювання книг, перегляд історії позичань, перегляд заходів, – так і інструменти для бібліотекарів та адміністраторів, зокрема керування каталогом, модерацію користувачів і обробку запитів.
У ході реалізації було проаналізовано особливості існуючих систем та їх обмеження, після чого спроєктовано архітектуру, яка включає клієнтську частину (frontend), серверну частину (backend) і базу даних. Особливу увагу приділено ролям користувачів – з чітким розмежуванням доступу для читача, бібліотекаря та адміністратора. Таке рішення дозволяє зберігати гнучкість системи, а також легко доповнювати її новими функціями в майбутньому.
Запровадження такої системи може помітно покращити обслуговування відвідувачів, полегшити роботу працівників і зробити бібліотеку більш доступною та зручною для користувачів в онлайн форматі. Розробка буде корисною не лише для міських бібліотек, а й для шкіл чи культурних установ, де важливо впорядкувати роботу з інформацією та заохотити людей долучатися до заходів.

2

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ
1.1 Аналіз предметної галузі
При дослідженні існуючих рішень для управління бібліотеками був застосований метод конкурентного бенчмаркінгу, який дозволяє провести порівняльний аналіз наявних на ринку продуктів та визначити їх сильні та слабкі сторони. Для об'єктивної оцінки були обрані ключові характеристики, важливі для бібліотечних систем, які оцінювалися за десятибальною шкалою.
Серед існуючих рішень для автоматизації бібліотек було проаналізовано кілька систем, які найбільш повно відповідають потребам сучасної міської бібліотеки.
Перший продукт – Koha.
Переваги системи:
– відкритий код, що дозволяє адаптувати систему під конкретні потреби;
– повноцінна підтримка міжнародних бібліотечних стандартів;

Рисунок А.2 – Друга сторінка перевірки на плагіат

# ДОДАТОК Б

## Слайди презентації

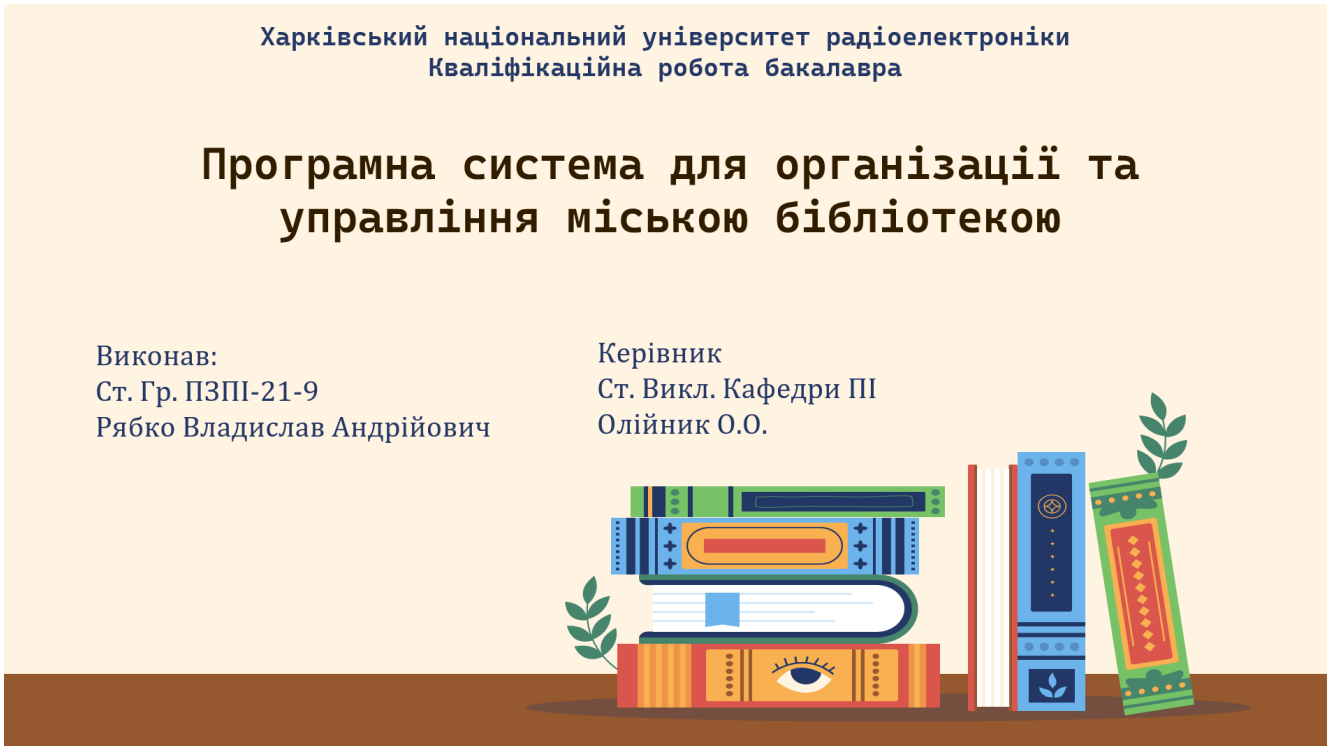



Рисунок Б.1 – Слайд 1

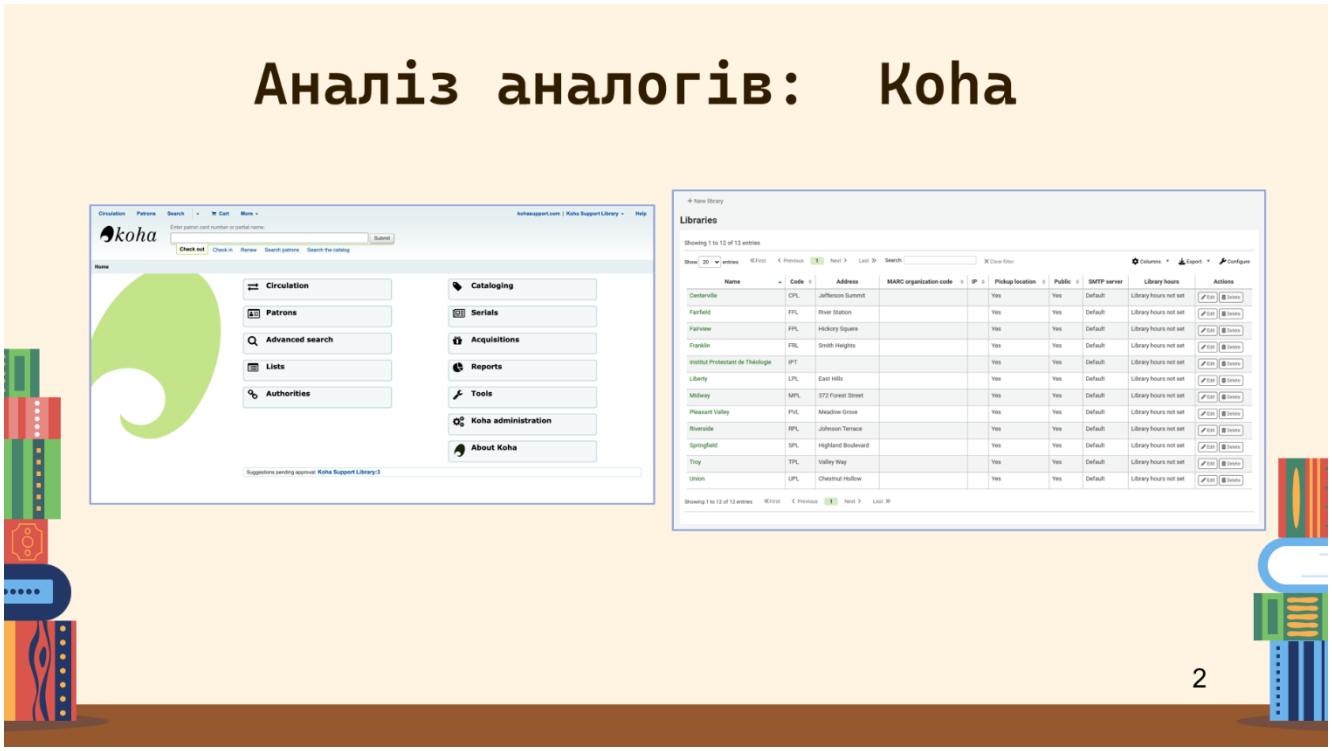
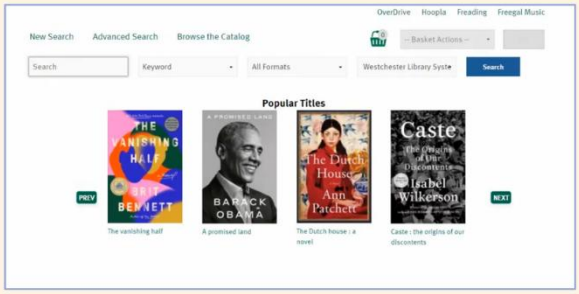
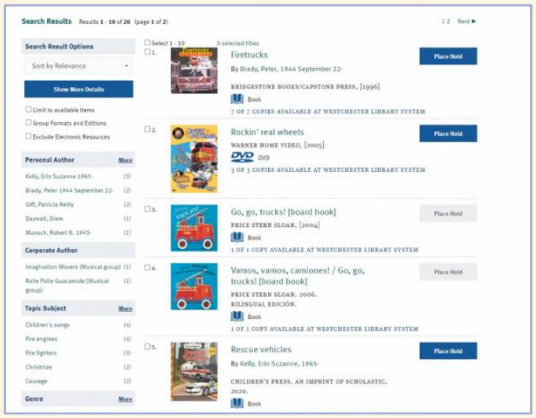



Рисунок Б.2 – Слайд 2

# Аналіз аналогів: Evergreen




3

Рисунок Б.3 – Слайд 3

# Постановка задачі

Метою є створення веб-застосунку для автоматизації процесів міської бібліотеки.

Основні задачі:

1) Створення інтерактивного каталогу з можливістю фільтрації, пошуку та ознайомлення з книжковим фондом і подіями бібліотеки.

2) Реалізація функціоналу для позичання книг.

3) Надання інтерфейсів для різних ролей: читач, бібліотекар, адміністратор.

4) Забезпечення зручного та безпечного доступу до функцій через клієнтську (React) і серверну (Django) частини системи.

4

Рисунок Б.4 – Слайд 4

# Вимоги

**Функціональні вимоги:**

1) Пошук та перегляд книг і подій для всіх користувачів.
2) Особистий кабінет з історією позичань для зареєстрованих читачів.
3) Додавання та редагування книг і заходів бібліотекарямита адміністраторами.
4) Управління системними параметрами адміністраторами. Можливість видалення користувачів.

**Нефункціональні вимоги:**

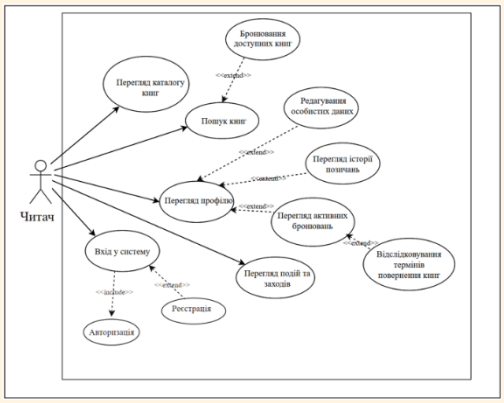
1) Простий та зрозумілий інтерфейс для зручного користування.
2) Безпечна аутентифікація через JWT.
3) Збереження даних у PostgreSQL.
4) REST API для взаємодії між фронтендом і бекендом.
5) Підтримка розмежування прав доступу залежно від ролі.

5

Рисунок Б.5 – Слайд 5

# Діаграми прецендентів

**Читач**




**Бібліотекар**


6

Рисунок Б.6 – Слайд 6

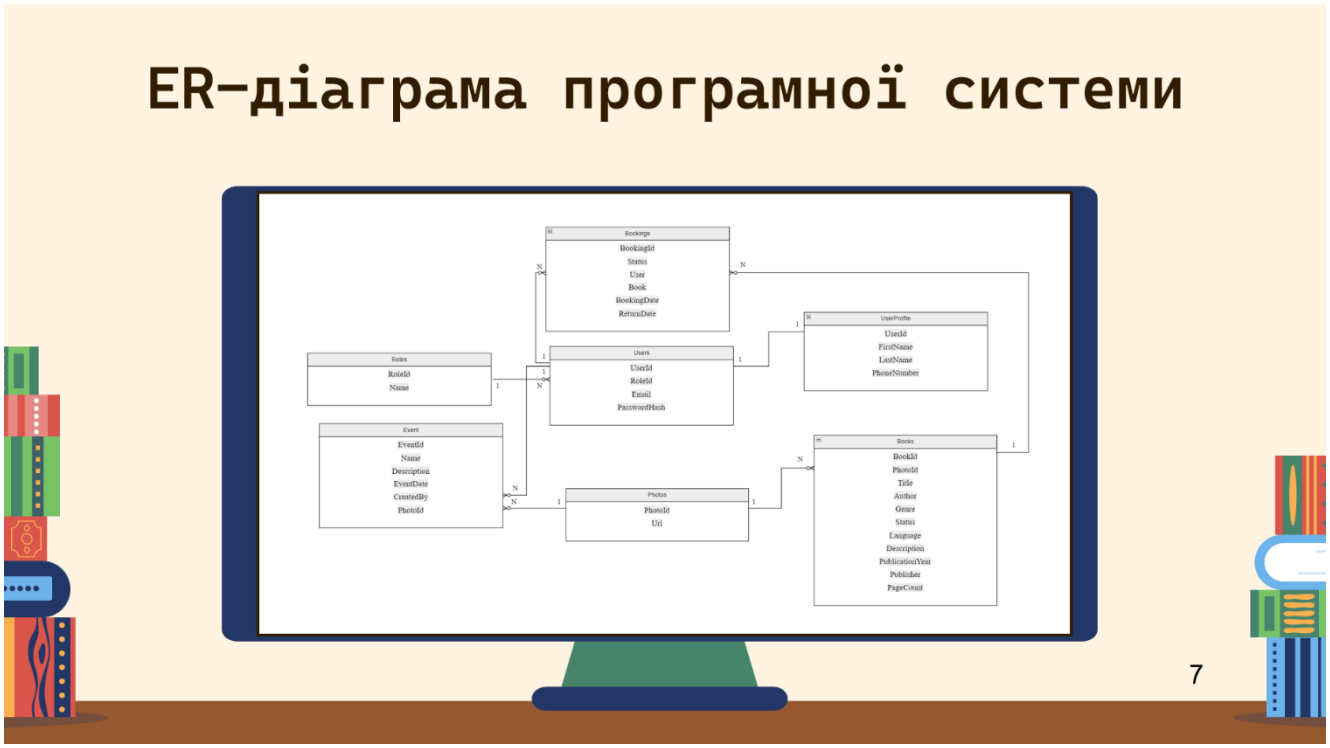



Рисунок Б.7 – Слайд 7


Рисунок Б.8 – Слайд 8

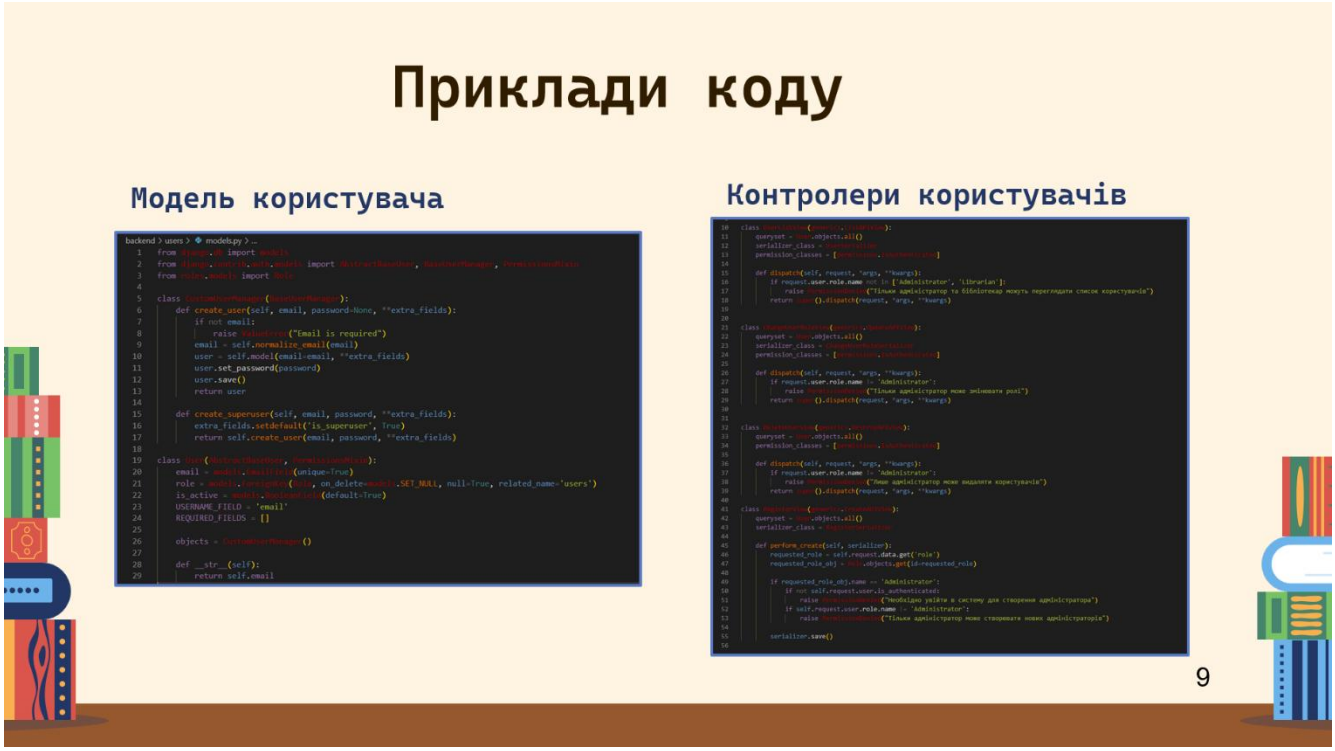



Рисунок Б.9 – Слайд 9

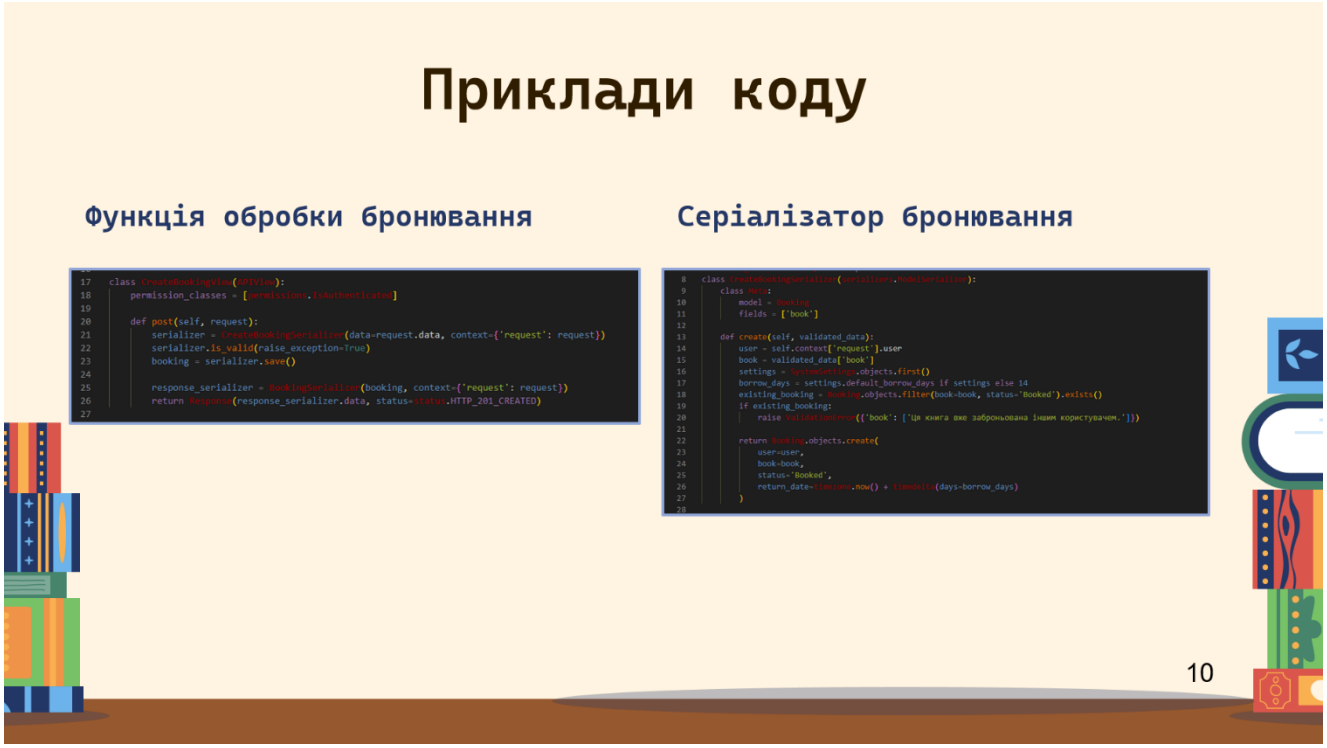



Рисунок Б.10 – Слайд 10

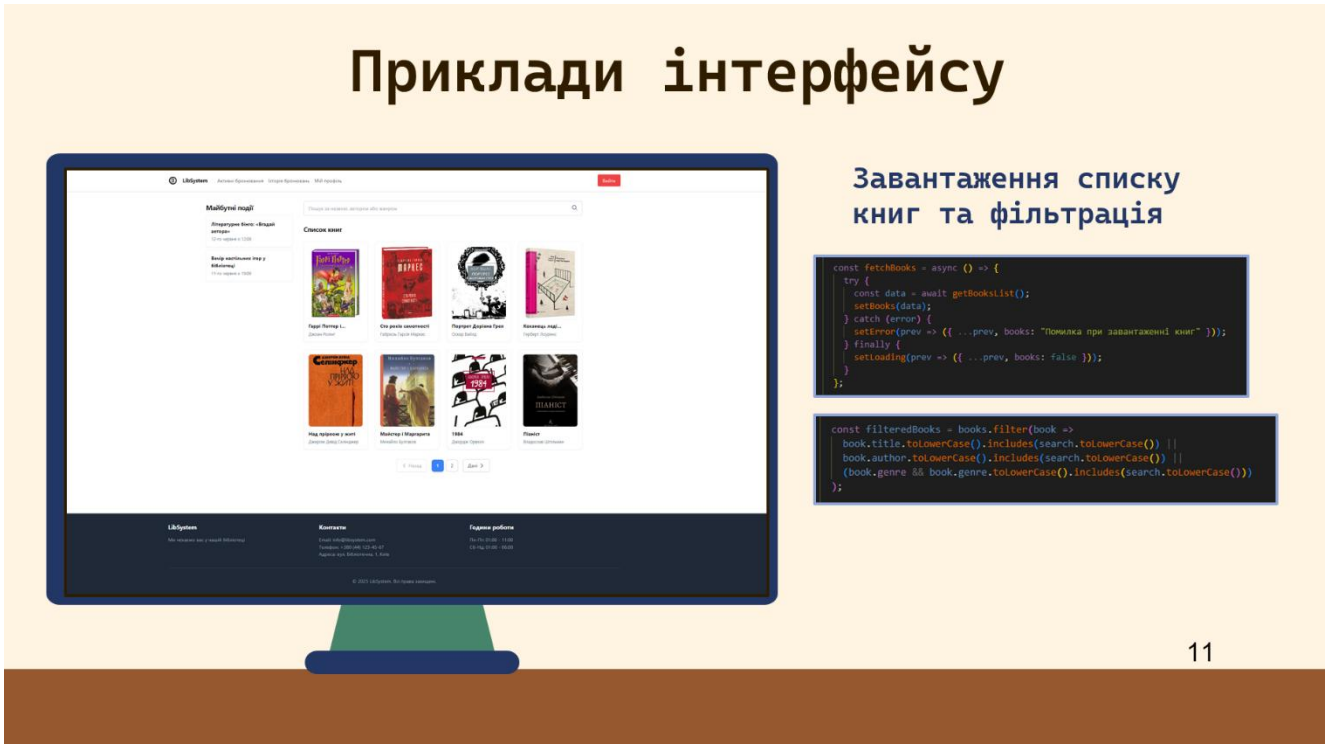



Рисунок Б.11 – Слайд 11

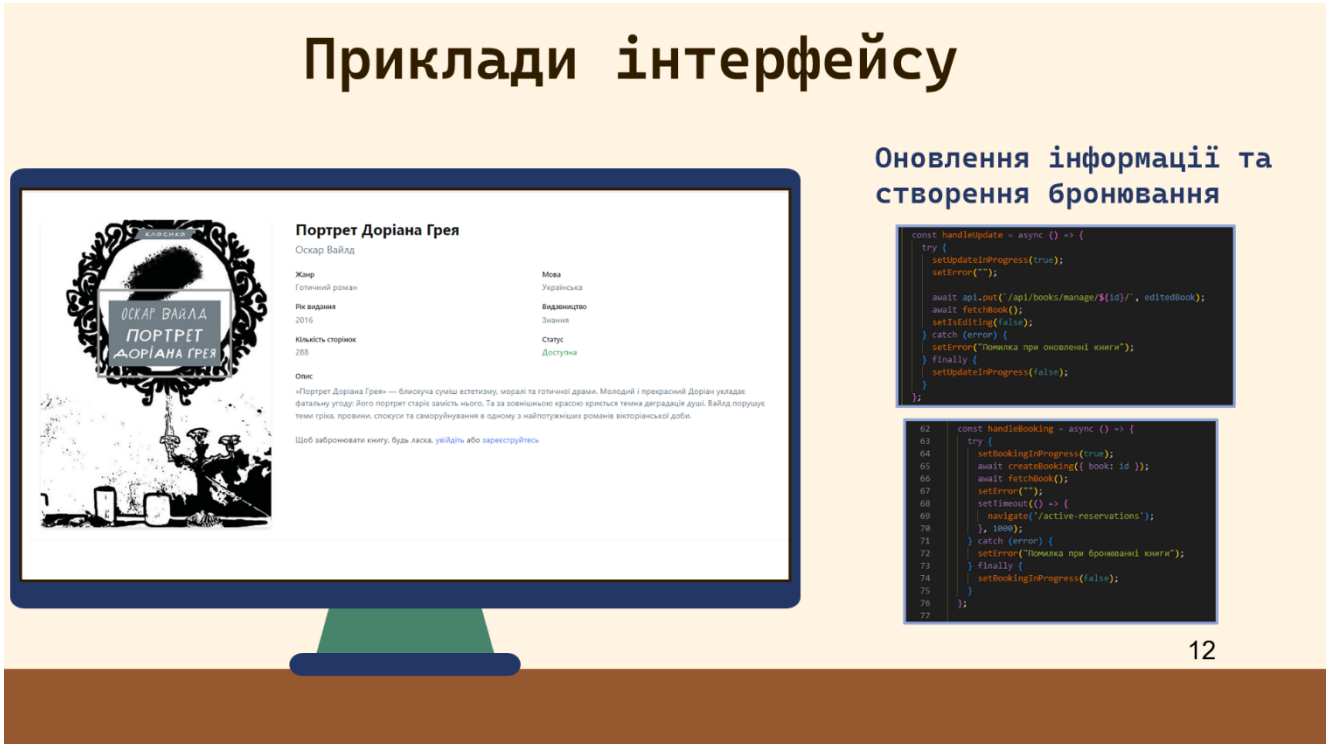
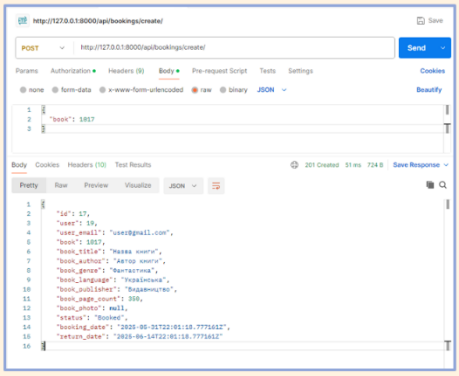


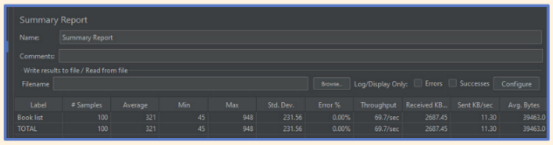

Рисунок Б.12 – Слайд 12

# Тестування

**Функціональне тестування – перевірка роботи API-запитів за допомогою Postman.**



**Навантажувальне тестування – оцінка продуктивності системи при великій кількості запитів за допомогою Apache JMeter**



13

Рисунок Б.13 – Слайд 13

# Висновки

**У результаті роботи реалізовано:**

- Авторизація та розмежування ролей (читач, бібліотекар, адміністратор).
- Електронний каталог книг із пошуком.
- Список подій.
- Резервування та повернення книг.
- Історія бронювань для користувачів.
- Керування подіями та книгами для бібліотекарів та адміністраторів.
- Адмін-функції: зміна параметрів системи, видалення користувачів.

**Плани на подальші релізи:**

- Повідомлення про дату повернення книг
- Система рекомендацій книг
- Підтримка мобільної версії
- Статистика популярності книг
- Тематичні добірки / новинки
- Розширення функціоналу адміністратора

14

Рисунок Б.14 – Слайд 14

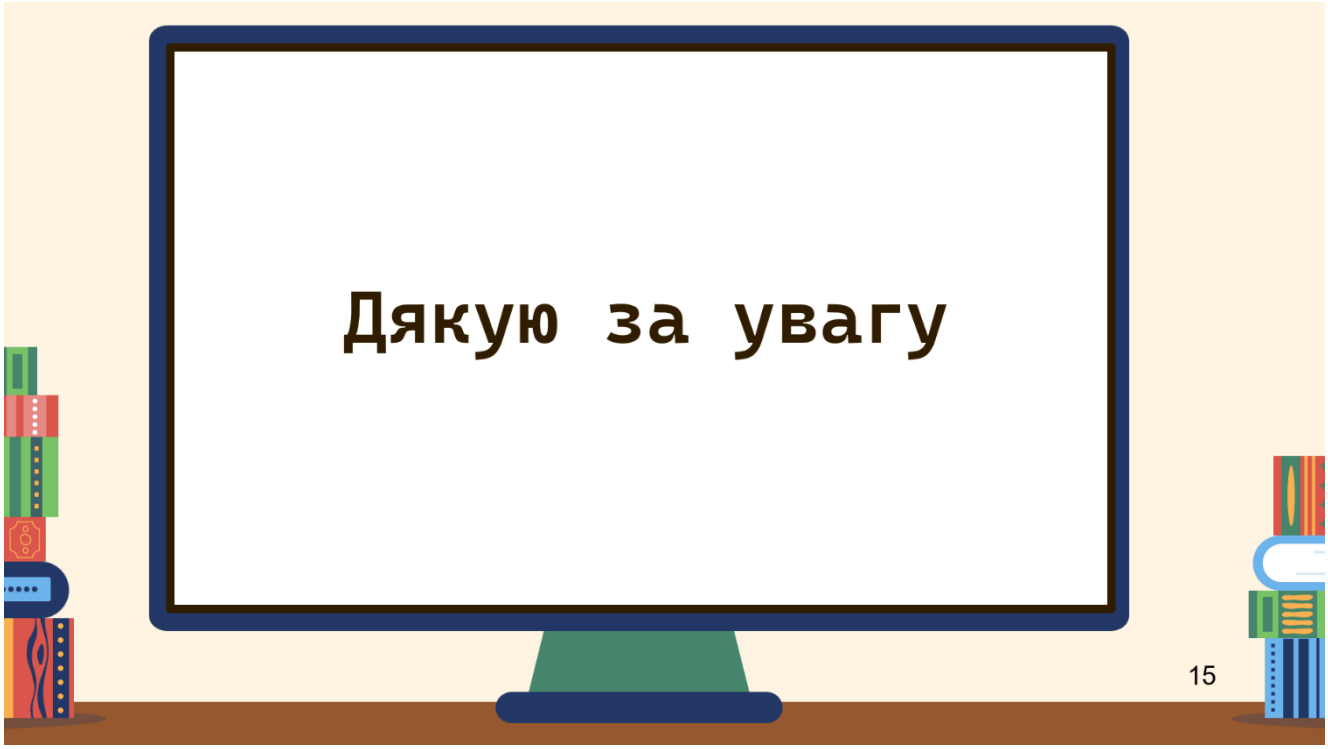



Рисунок Б.15 – Слайд 15

# ДОДАТОК В

## Код моделі користувача

```
from django.db import models
from django.contrib.auth.models import AbstractBaseUser, BaseUserManager,
PermissionsMixin
from roles.models import Role

class CustomUserManager(BaseUserManager):
    def create_user(self, email, password=None, **extra_fields):
        if not email:
            raise ValueError("Email is required")
        email = self.normalize_email(email)
        user = self.model(email=email, **extra_fields)
        user.set_password(password)
        user.save()
        return user

    def create_superuser(self, email, password, **extra_fields):
        extra_fields.setdefault('is_superuser', True)
        return self.create_user(email, password, **extra_fields)

class User(AbstractBaseUser, PermissionsMixin):
    email = models.EmailField(unique=True)
    role = models.ForeignKey(Role, on_delete=models.SET_NULL, null=True,
related_name='users')
    is_active = models.BooleanField(default=True)
    USERNAME_FIELD = 'email'
    REQUIRED_FIELDS = []

    objects = CustomUserManager()

    def __str__(self):
        return self.email
```

# ДОДАТОК Г

## Код серіалізаторів користувача

```
from rest_framework import serializers
from .models import User
from roles.models import Role
from rest_framework.exceptions import PermissionDenied

class UserSerializer(serializers.ModelSerializer):
    class Meta:
        model = User
        fields = ['id', 'email', 'role']

class ChangeUserRoleSerializer(serializers.ModelSerializer):
    role = serializers.PrimaryKeyRelatedField(queryset=Role.objects.all())

    class Meta:
        model = User
        fields = ['role']

class RegisterSerializer(serializers.ModelSerializer):
    password = serializers.CharField(write_only=True)
    role = serializers.PrimaryKeyRelatedField(queryset=Role.objects.all())

    class Meta:
        model = User
        fields = ['email', 'password', 'role']

    def validate_role(self, value):
        request = self.context.get('request')
        if value.name == 'Administrator':
            if not request or not request.user.is_authenticated:
                raise PermissionDenied("Необхідно увійти в систему для
створення адміністратора")
            if request.user.role.name != 'Administrator':
                raise PermissionDenied("Тільки адміністратор може
призначати роль адміністратора")
        return value

    def create(self, validated_data):
        role = validated_data.pop('role', None)
        user = User.objects.create_user(
            email=validated_data['email'],
            password=validated_data['password'],
        )
        user.role = role
        user.save()
        return user
```

# ДОДАТОК Д

## Код головної сторінки застосунку

```
import { useState, useEffect } from "react";
import { Link } from "react-router-dom";
import SearchBar from "../components/SearchBar";
import { getEventsList } from "../api/events";
import { getBooksList } from "../api/books";

function Home() {
  const [events, setEvents] = useState([]);
  const [books, setBooks] = useState([]);
  const [search, setSearch] = useState('');
  const [currentPage, setCurrentPage] = useState(1);
  const [loading, setLoading] = useState({
    events: true,
    books: true
  });
  const [error, setError] = useState({
    events: "",
    books: ""
  });

  const BOOKS_PER_PAGE = 8;

  useEffect(() => {
    const fetchEvents = async () => {
      try {
        const data = await getEventsList();
        setEvents(data);
      } catch (error) {
        setError(prev => ({ ...prev, events: "Помилка при завантаженні
подій" }));
      } finally {
        setLoading(prev => ({ ...prev, events: false }));
      }
    };

    const fetchBooks = async () => {
      try {
        const data = await getBooksList();
        setBooks(data);
      } catch (error) {
        setError(prev => ({ ...prev, books: "Помилка при завантаженні книг"
}));
      } finally {
        setLoading(prev => ({ ...prev, books: false }));
      }
    };

    fetchEvents();
    fetchBooks();
  }, []);


  useEffect(() => {
```

```
    setCurrentPage(1);
  }, [search]);

  const formatDate = (dateString) => {
    const date = new Date(dateString);
    const day = date.getDate();
    const monthNominative = date.toLocaleString('uk-UA', { month: 'long'
});
    const monthGenitive = monthNominative.replace(/ень$/, 'ня')
                                         .replace(/й$/, 'я')
                                         .replace(/т$/, 'та')
                                         .replace(/ад$/, 'ада')
                                         .replace(/опад$/, 'опада')
                                         .replace(/есень$/, 'есня');
    const hours = date.getHours().toString().padStart(2, '0');
    const minutes = date.getMinutes().toString().padStart(2, '0');

    return `${day}-го ${monthGenitive} о ${hours}:${minutes}`;
  };

  const filteredBooks = books.filter(book =>
    book.title.toLowerCase().includes(search.toLowerCase()) ||
    book.author.toLowerCase().includes(search.toLowerCase()) ||
    (book.genre && book.genre.toLowerCase().includes(search.toLowerCase()))
  );

  const totalPages = Math.ceil(filteredBooks.length / BOOKS_PER_PAGE);
  const startIndex = (currentPage - 1) * BOOKS_PER_PAGE;
  const paginatedBooks = filteredBooks.slice(startIndex, startIndex +
BOOKS_PER_PAGE);

  return (
    <div className="gap-1 px-6 flex flex-1 justify-center py-5">
      <div className="layout-content-container flex flex-col w-80">
        <h2 className="text-[#121416] text-[22px] font-bold leading-tight
tracking-[-0.015em] px-4 pb-3 pt-5">
          Майбутні події
        </h2>
        {loading.events ? (
          <div className="text-center py-4">
            <div className="inline-block h-8 w-8 animate-spin rounded-full
border-4 border-solid border-[#1978e5] border-r-transparent"></div>
          </div>
        ) : error.events ? (
          <p className="text-red-500 text-sm px-4">{error.events}</p>
        ) : events.length === 0 ? (
          <p className="text-gray-500 text-sm px-4">Немає запланованих
подій</p>
        ) : (
          <div className="flex flex-col gap-2 px-4">
            {events.map((event) => (
              <Link
                key={event.id}
                to={`/events/${event.id}`}
                className="block"
              >
                <div
```

```
                  className="p-4 bg-white rounded-lg shadow-sm border
border-gray-100 hover:shadow-md transition-shadow"
                >
                  <h3 className="text-[#121416] text-base font-bold mb-1">
                    {event.name}
                  </h3>
                  <p className="text-[#637488] text-sm">
                    {formatDate(event.event_date)}
                  </p>
                </div>
              </Link>
            ))}
          </div>
        )}
      </div>
      <div className="layout-content-container flex flex-col max-w-[960px]
flex-1">
        <div className="px-4 py-3">
          <SearchBar
            placeholder="Пошук за назвою, автором або жанром"
            value={search}
            onChange={setSearch}
          />
        </div>
        <h2 className="text-[#121416] text-[22px] font-bold leading-tight
tracking-[-0.015em] px-4 pb-3 pt-5">
          Список книг
        </h2>
        {loading.books ? (
          <div className="text-center py-4">
            <div className="inline-block h-8 w-8 animate-spin rounded-full
border-4 border-solid border-[#1978e5] border-r-transparent"></div>
          </div>
        ) : error.books ? (
          <p className="text-red-500 text-sm px-4">{error.books}</p>
        ) : filteredBooks.length === 0 ? (
          <p className="text-gray-500 text-sm px-4">Книг не знайдено</p>
        ) : (
          <div className="p-4">
            <div className="grid grid-cols-4 gap-6">
              {paginatedBooks.map((book) => (
                <Link
                  key={book.id}
                  to={`/books/${book.id}`}
                  className="block transform hover:scale-105 transition-all
duration-200"
                >
                  <div
                    className="flex flex-col gap-3 p-4 bg-white rounded-xl
shadow-sm border border-gray-100 hover:shadow-lg"
                  >
                    <div
                      className="w-full aspect-[3/4] bg-center bg-cover
rounded-lg shadow-sm"
                      style={{
                        backgroundImage: book.photo
                          ? `url(${book.photo})`
```

```
                          :
'url(https://via.placeholder.com/200x300?text=Немає+фото)'
                        }}
                      />
                      <div className="space-y-1">
                        <h3 className="text-[#121416] text-base font-bold
line-clamp-1">
                          {book.title}
                        </h3>
                        <p className="text-[#637488] text-sm">
                          {book.author}
                        </p>
                      </div>
                    </div>
                  </Link>
                ))}
              </div>
              {totalPages > 1 && (
                <div className="flex justify-center items-center mt-8 gap-4">
                  <button
                    onClick={() => setCurrentPage(prev => Math.max(prev - 1,
1))}
                    disabled={currentPage === 1}
                    className="flex items-center px-4 py-2 bg-white border
border-gray-300 rounded-lg text-gray-700 hover:bg-gray-50 disabled:opacity-
50 disabled:cursor-not-allowed transition-all duration-200 shadow-sm"
                  >
                    <svg className="w-5 h-5 mr-1" fill="none"
stroke="currentColor" viewBox="0 0 24 24">
                      <path strokeLinecap="round" strokeLinejoin="round"
strokeWidth={2} d="M15 19l-7-7 7-7" />
                    </svg>
                    Назад
                  </button>
                  <div className="flex items-center gap-2">
                    {Array.from({ length: totalPages }, (_, i) => i +
1).map(page => (
                      <button
                        key={page}
                        onClick={() => setCurrentPage(page)}
                        className={`w-10 h-10 flex items-center justify-
center rounded-lg transition-all duration-200 ${
                          currentPage === page
                            ? 'bg-blue-500 text-white font-medium shadow-md'
                            : 'bg-white text-gray-700 hover:bg-gray-50 border
border-gray-300'
                        }`}
                      >
                        {page}
                      </button>
                    ))}
                  </div>
                  <button
                    onClick={() => setCurrentPage(prev => Math.min(prev + 1,
totalPages))}
                    disabled={currentPage === totalPages}
```

```
                  className="flex items-center px-4 py-2 bg-white border
border-gray-300 rounded-lg text-gray-700 hover:bg-gray-50 disabled:opacity-
50 disabled:cursor-not-allowed transition-all duration-200 shadow-sm"
                >
                  Далі
                  <svg className="w-5 h-5 ml-1" fill="none"
stroke="currentColor" viewBox="0 0 24 24">
                    <path strokeLinecap="round" strokeLinejoin="round"
strokeWidth={2} d="M9 5l7 7-7 7" />
                  </svg>
                </button>
              </div>
            )}
          </div>
        )}
      </div>
    </div>
  );
}

export default Home;
```