МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Кафедра інженерії програмного забезпечення

**КУРСОВИЙ РОБОТА**

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

з дисципліни Web-технології Ч.2

на тему:

**«CMS-система для інтернет-магазину спортивного харчування**

**«Grozer Sports Nutrition» »**

студента II курсу групи ІПЗ-20-4

спеціальності 121 «Інженерія програмного забезпечення»

Васьківського Віталія Юрійовича

(прізвище, ім’я та по-батькові)

Керівник: Морозов А.В .

Дата захисту: " \_\_\_ " \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ р.

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оцінка: ECTS \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Граф М.С .

(підпис) (прізвище та ініціали) \_\_\_Кузьменко О.В\_ .

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Чижмотря О.Г .

(підпис) (прізвище та ініціали)

Житомир – 2022

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Факультет інформаційно-комп’ютерних технологій

Кафедра інженерії програмного забезпечення

Освітній рівень: бакалавр

Спеціальність 121 «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»

В.о зав кафедри ІПЗ

\_\_\_\_\_\_\_\_\_\_\_\_ А.В Морозов

“\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_2021 р.

ЗАВДАННЯ

НА КУРСОВУ РОБОТУ СТУДЕНТУ

Васьківському Віталію Юрійовичу

1. Тема роботи: Розробка інтернет-магазину спортивного харчування

керівник роботи: Морозов А.В .

1. Строк подання студентом: “ ” 2021р.
2. Вихідні дані до роботи: Розробити інтернет-магазин спортивного харчування – розробити та реалізувати систему адміністрування для сайту
3. Зміст розрахунково-пояснювальної записки (перелік питань, що підлягають розробці)
   * + 1. Постановка завдання
       2. Аналіз аналогічних розробок
       3. Алгоритми роботи програми
       4. Опис роботи програми
4. Перелік графічного матеріалу(з точним зазначенням обов’язкових креслень)

1. Презентація до КР\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Посилання на репозиторій: https://gitlab.com/ki201\_vvyu/courseworkgrozersportsnutrition.git

1. Консультанти розділів проекту (роботи)

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Прізвище, ініціали та посади консультанта | Підпис, дата | |
| завдання  видав | завдання прийняв |
| 1, 2, 3 | Чижмотря О.Г., ст. викладач каф. ІПЗ |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Дата видачі завдання “ ” 2021 р.

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| № з/п | Назва етапів курсової роботи | Термін виконання етапів роботи | Примітки |
| 1 | Постановка задачі | 4.12.21 – 5.12.21 | Виконано |
| 2 | Пошук, огляд та аналіз аналогічних розробок | 5.12.21 – 6.12.21 | Виконано |
| 3 | Формулювання технічного завдання | 6.12.21 – 8.12.21 | Виконано |
| 4 | Опрацювання літературних джерел | 8.12.21 – 12.12.21 | Виконано |
| 5 | Проектування структури | 12.12.21 – 14.12.21 | Виконано |
| 6 | Написання програмного коду | 14.12.21 – 31.12.21 | Виконано |
| 7 | Відлагодження | 01.01.22 –  10.01.22 | Виконано |
| 8 | Написання пояснювальної записки | 05.01.22 –  10.01.22 | Виконано |
| 9 | Захист | 17.01.22 |  |

**Студент***\_\_\_\_\_\_\_\_\_\_\_\_\_\_* Васьківський В. Ю.

(підпис) (прізвище та ініціали)

**Керівник проекту** \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Морозов А. В.

(підпис) (прізвище та ініціали)

**РЕФЕРАТ**

Пояснювальна записка до курсового проекта на тему «Розробка інтернет магазину спортивного харчування» складається з переліку умовних скорочень, вступу, трьох розділів, висновків, списку використаної літератури та додатку.

Текстова частина викладена на 32 сторінках друкованого тексту.

Пояснювальна записка має 1 сторінки додатків. Список використаних джерел містить 12 найменувань і займає 1 сторінку. В роботі наведено 23 рисунів. Загальний обсяг роботи – 36 сторінок.

У першому розділі було обґрунтовано створення сайту на тему Розробка інтернет-магазину спортивного харчування.

У другому розділі проведено проектування і розробка сайту.

У третьому розділі проведено тестування програмного продукту.

Висновок містить в собі результати виконаної роботи при створенні сайту на тему «Розробка інтернет-магазину спортивного харчування».

У додатку представлений лістинг розробленого програмного продукту.

Ключові слова:

MYSQL, JAVASCRIPT, TYPESCRIPT, REACT, NEST JS, GRAPH QL, REDUX, APOLLO CLIENT, TYPEORM, SINGLE PAGE APPLICATION, WEB-САЙТ, MVC, СЕРВЕР, АДМІН, АВТОРИЗАЦІЯ, МАГАЗИН, ІНТЕРНЕТ, ДАНІ.

# 

**Зміст**

[**ВСТУП** 7](#_Toc92540836)

[РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ 8](#_Toc92540837)

[**1.1** **Аналіз задачі, засобів та методів її вирішення** 8](#_Toc92540838)

[**1.2** **Аналіз існуючого програмного забезпечення за тематикою курсової роботи.** 8](#_Toc92540839)

[**Висновки до першого розділу :** 14](#_Toc92540840)

[РОЗДІЛ 2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 16](#_Toc92540841)

[**2.1 Проектування загального алгоритму роботи веб-додатку** 16](#_Toc92540842)

[Загальна схема роботи програми : 17](#_Toc92540843)

[**2.2 Розробка функціональних алгоритмів роботи програми** 18](#_Toc92540844)

[**2.3 Розробка програмного забезпечення** 19](#_Toc92540845)

[РОЗДІЛ 3 ОПИС РОБОТИ З ПРОГРАМНИМ ДОДАТКОМ ТА ЙОГО ТЕСТУВАННЯ 24](#_Toc92540846)

[**3.1 Опис роботи з програмним додатком (Опис інтерфейсу)** 24](#_Toc92540847)

[**3.2** **Тестування роботи програмного забезпечення** 30](#_Toc92540848)

[**Висновки до 3 розділу :** 31](#_Toc92540849)

[**ВИСНОВКИ** 32](#_Toc92540850)

[**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ** 33](#_Toc92540851)

[**ДОДАТКИ** 34](#_Toc92540852)

**ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ**

БД – База даних

КР – курсова робота

MVC – Модель–вигляд–контролер (або Модель–представлення–контролер, Model-view-controller, MVC) — архітектурний шаблон, який використовується під час проектування та розробки програмного забезпечення.

ПЗ – Програмне забезпечення

GRUD – Create – створення, Read – читання, Update – оновлення, Delete – видалення.

JWT – Json Web Token

# **ВСТУП**

У цій курсовій роботі буде наведено процес розробки інтернет-магазину спотивного харчування. Взагалом останнім часом інтернет-магазини ствють все більш і більш популярними, тому що дозволяють здійснювати покупки не виходячи з дому, тому дана тема курсової є актуальною.

**Метою** курсової роботи є дослідження особливостей проектування та реалізації cистеми адміністрування інтернет-магазину книг.

Завданням на курсову роботу є:

* аналіз теоретичних засад проектування та реалізації CMS;
* аналіз існуючих веб-сайтів на обрану вами тематику, визначення унікальності;
* визначення інформаційних потреб предметної області дослідження;
* розробка адаптивного інтерфейсу веб-сайту за допомогою HTML5, CSS3, JavaScript, React, TypeScript;
* розробка математичної та алгоритмічної моделі функціонування системи на основі БД;
* проектування бази даних за визначеною предметною областю;
* написання серверної частини за допомогою Nest JS, Nest JS Graph QL, TypeOrm;
* реалізація CMS.

**Об’єктом дослідження** є методи та засоби розробки CMS для інтернет-магазину спортивного харчування.

**Предмет дослідження** є використання веб-технологій для забезпечення спортсменів по всьому світу спортивним харчуванням.

# РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ

## **Аналіз задачі, засобів та методів її вирішення**

Завдання полягає в тому, що потрібно:

1. Визначити потреби звичайного користувача.
2. Зрозуміти основні принципи та алгротитми роботи інтернет-магазину
3. Продумати можливість GRUD операцій для товарів, категорій, файлів, сторінок, замовлень.
4. Реалізувати сортування та пошук для кожного з модулів.
5. Реалізувати userfriendly інтерфейс.
6. Продумати розподілення ролей та можливість створення нових адмінів.
7. Створити корректну базу даних.
8. Продумати дизайн сайту.
9. Реалізувати адаптивність для різних пристроїв.

Для написання фронтенду було обрано бібліотеки React – для реалізація SPA, Redux – для стейт менеджменту, Apollo Client для Graph QL запитів на сервер.

Для написання бекенду було обрано фреймворки Nest JS – як основа, з доповненням Nest JS Graph QL для написання Graph QL серверу, який є дуже зручним для використання на будь-якому клієнтському додатку. Адже він не віддає надлишкові данні та є можливість кастомізування формат отримання.

У якості БД було обрано MySQL.

Для розробки КР використано такі середовище розробки WebStorm, тому що воно підтримує всі необхідні інструмети в ході розробки.

## **Аналіз існуючого програмного забезпечення за тематикою курсової роботи.**

При аналізі вже існуючих веб-сайтів за тематикою курсової роботи було виявлено декілька проектів. Всі вони схожі і мають подібну реалізацію.

Aналогічні інтренет-магазини за заданою темою:

**Belok**

Посилання: https://belok.ua/

Розглянемо даний аналог:

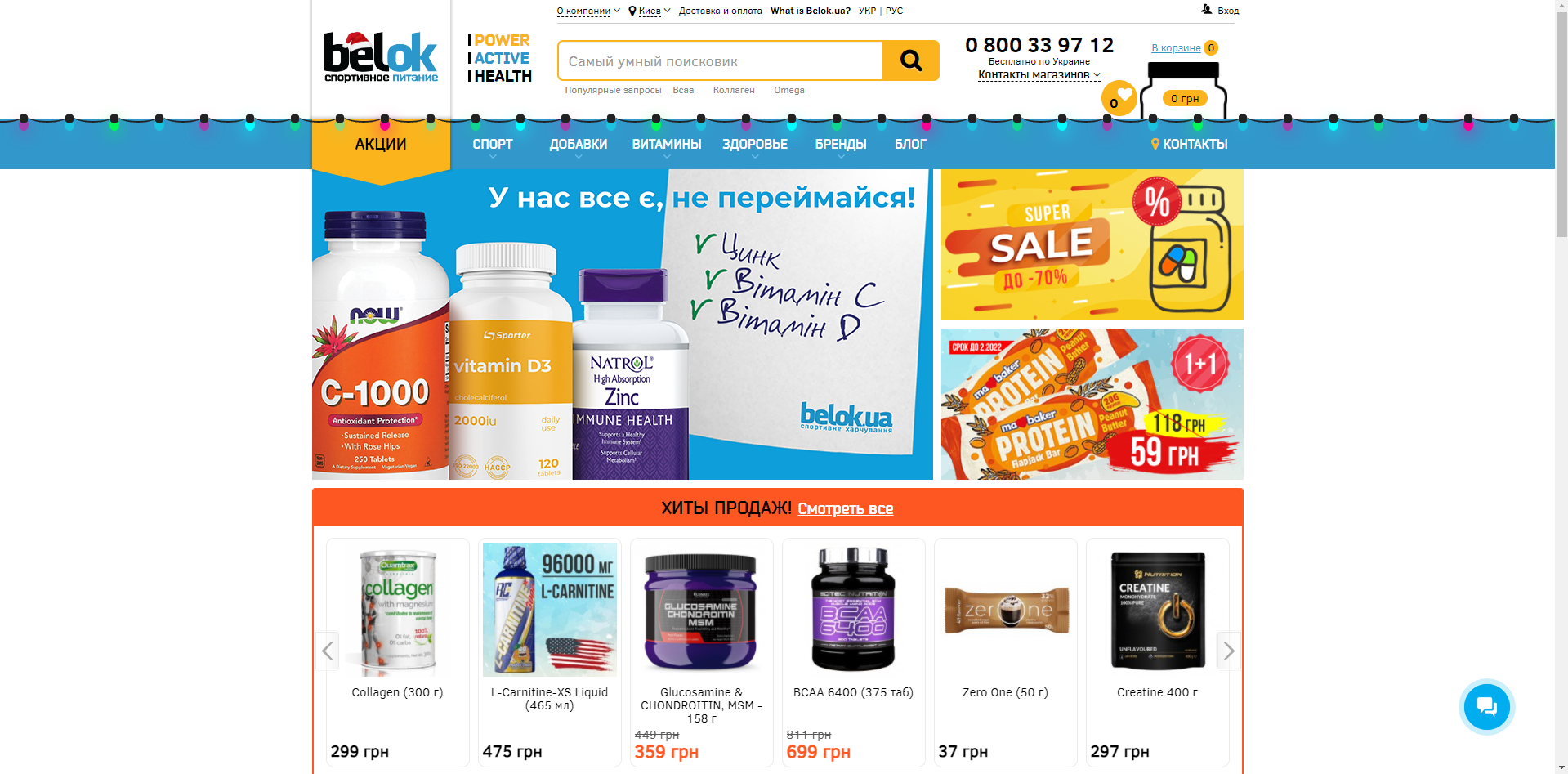


Рис 1.1 Belok

Інтернет-магазин Belok є одним із найпопулярніших інтернет-магазинів спортивного харчування, адже має інтуїтивно зрозумілий користувацький інтріейс та широкий товарів. Також має широкий вибір параметрів для сортування товарів. Серед мінусім можна виділити відсутність адаптиву.

Основні плюси:

* Інтуїтивно зрозумілий користувацький інтерфейс
* Широкий вибір параметрів для сортування товарів
* Широкий асортимент товарів
* Гарний дизайн

Основні мінуси:

* Відсутність адаптиву

**SportMenu**

Посилання : https://sportmenu.com.ua/

Розглянемо даний аналог:

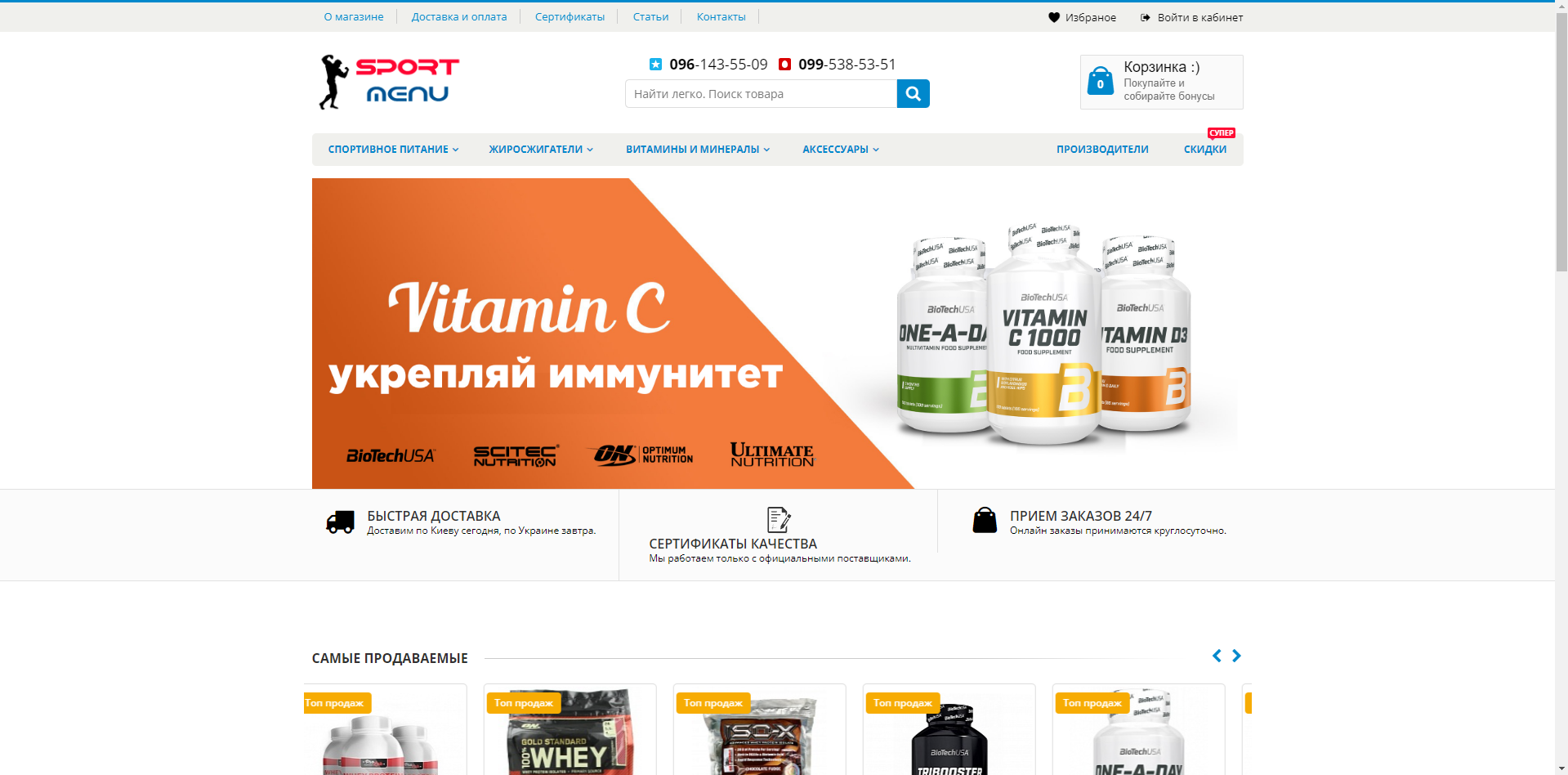


Рис 1.2 SportMenu

Інтернет-магазин SportMenu є одним із найпопулярніших інтернет-магазинів спортивного харчування, адже має інтуїтивно зрозумілий користувацький інтріейс та широкий товарів. Також має широкий вибір параметрів для сортування товарів. Серед мінусім можна виділити не найбільш привабливий інтерфейс

Основні плюси:

* Широкий вибір параметрів для сортування товарів
* Широкий асортимент книг
* Один з найперших інтернет-магазинів спортивного харчування

Основні мінуси:

* Відсутність адаптиву
* Незначно надмірний користувацький інтерфейс

**BBR**

Посилання: https://bbr.in.ua

Розглянемо даний аналог:

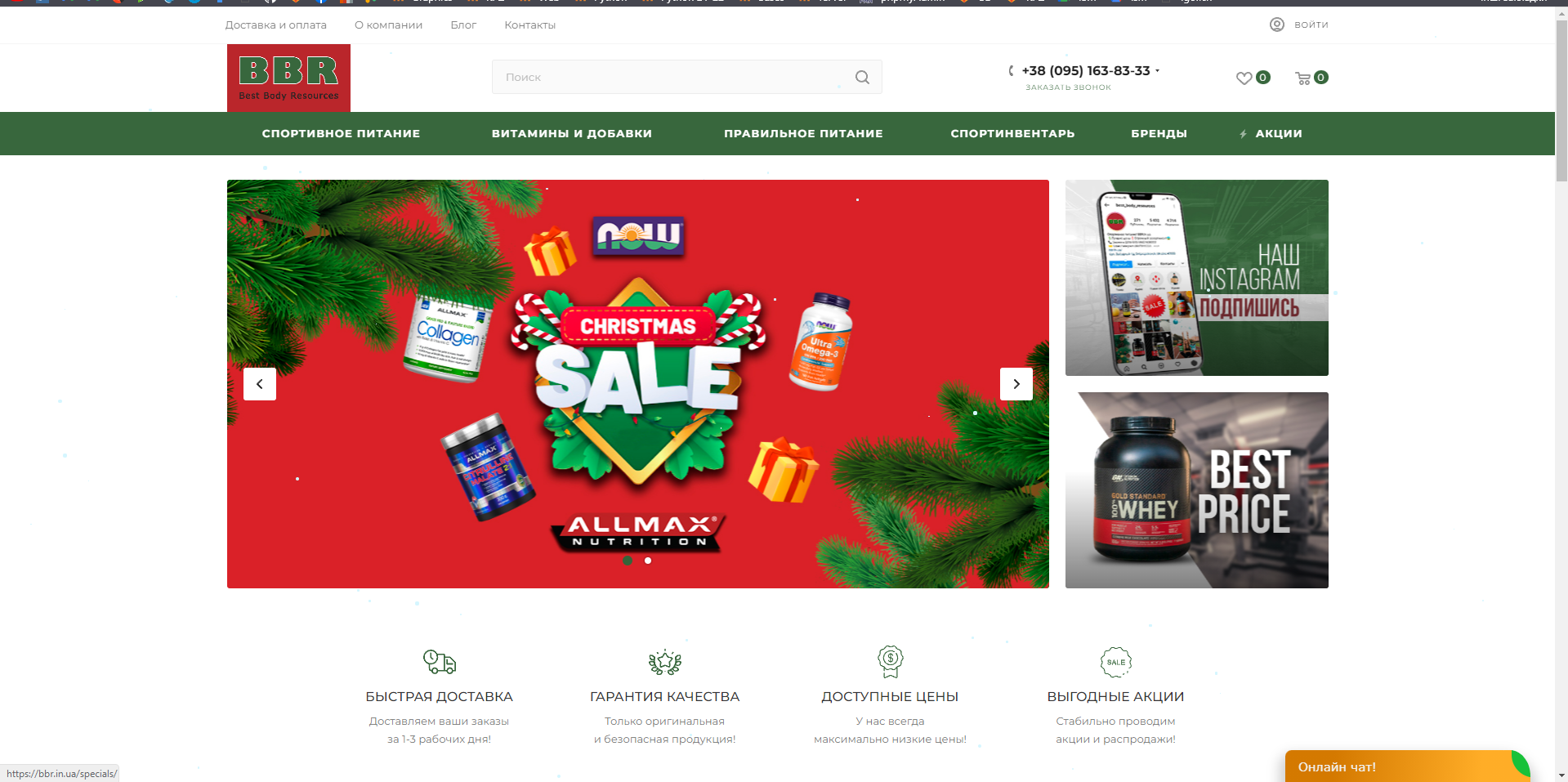


Рис 1.3 BBR

Інтернет-магазин BBR це інтернет-магаизин, який має стильний дизайн та можливості попередніх розглянутих продуктів. Навідміну від попередньо-розглянутих сайтів, має також адаптив.

Основні плюси:

* Всі, що і в попередньо-розглянутих інтерент-магазинів
* Адаптив для різних пристроїв

Основні мінуси:

* Не такий відомий як попередньо-розглянуті магазини

Актуальним напрямком реалізації власного інтернет-магазину є створення веб-сайту з бізнес логікою подібною до реалізованої у всіх представлених продуктах, реалізувати користувацький інтерфейс по принципам Belok, уникаючи не гарного дизайну як у SportMenu, але послідувати основним аспектам інтерфейсу цього продукту. Також продумати сучасний дизай сайту, звертаючи увагу до дизайну Belok та реалізувати адаптив.

## **1.3 Технічне завдання на курсову роботу**

**1. Загальне положення**

**1.1. Найменування програмного засобу**

Повне найменування програмної системи: "Розробка інтернет-магазину спортивного харчування GrozerSportsNutrition" (надалі "веб-додаток"). Коротка назва програмної системи - " Інтернет-магазин"

**1.2. Призначення розробки та область застосування**

Веб-сайт "Інтернет-магазин спортивного харчування" призначений для розміщення каталогу товарів, їх пошуку, та продажу. Система може бути впроваджена у роботу інтернет-магазину. Веб-сайт " Інтернет-магазин спортивного харчування" дозволить швидко переглядати інформацію про товари, категорії та забезпечить керівництву підприємства своєчасною та повною інформацією про актуальний стан замовлення покупок інтернет-магазину у повному обсязі, а також надасть користувачам можливість вибору спортивного харчування за їх смаком.

**1.3. Найменування розробника та замовника.**

Розробник даного продукту - студент групи ІПЗ-20-4 Васьківський Віталій Юрійович (надалі "розробник"). Замовник програмного продукту – кафедра інженерія програмного забезпечення Житомирського державного технологічного університету в межах виконання курсової з дисципліни «Web-тееології» Чижмотря Олена Геннадіївна, Морозов Андрій Васильович **2. Підстава для розробки**

**2.1. Документ на підставі якого ведеться розробка**

Робота ведеться на підставі навчального плану за напрямом 121 «Інженерія програмного забезпечення».

**3. Вимоги до програми**

**3.1. Вимоги до функціональних характеристик.**

**3.1.1. Загальні вимоги**

Веб-додаток має забезпечувати:

• можливість дистанційної роботи з робочих станцій локальної та глобальної мережі підприємства;

• постійний доступ користувачів веб-додатку;

• оформлення замовлення;

• організацію управління сайтом;

• можливість доступ до бази даних;

**3.1.2. Склад виконуваних функцій**

Розробити інтернет-магазин книг, що підтримує виконання наступних операцій:

1. Глобальний пошук.
2. GRUD операції для модулів товари, категорії, файли, замовлення, сторінки

## **Висновки до першого розділу :**

У ході виконання першого розділу було поставлено завдання КР, а саме який має бути функціонал і визначено актуальність теми. Також визначено мови і фреймворки для розробки та актуальний напрямок розробки власного продукту. Розглянуто існуючі програмні продукти зі схожою тематикою та більш детально проаналізовано деякі із них, а саме : Belok, SportMenu та BBR.

# РОЗДІЛ 2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## **2.1 Проектування загального алгоритму роботи веб-додатку**

На даному етапі розробки розглянемо загальний алгоритм роботи інтернет-магазину. Загальна схема роботи приведена на рисунку 2.1.

Для початку розробки бекенду було стоврено моделі, потім для них сервіси для роботи з базою данних та Graph QL резолвери, які є чимось схожі до методів контролера при розробці за домогою патерну MVC

Аторизація реалізована на основі JWT токену в якому зашифровано id, email та ролі користувача

Тепер спроектуємо візуальну частину. Для цього використаємо бібліотеку React для написання SPA, та також за допомогою бібліотеки React Router Dom створити роути для кожного з модулів веб додатку. Для запитів на сервер використовується бібліотека Apollo Client, за допомогою неї можна зручного робити запити прямо з компоненти з використанням хуків useQuery – для отримання данних, та useMutation – для змінення.

У веб-додатку реалізовано 3 ролі. Customer (покупець) має можливість перегладати частину додатку для покупців та оформляти замовлення. Moderator (модератор) має можливісті customer`a та також керувати контектом сайту з адмін панелі. Admin (адмін) – має всі вище перераховані можливості та також видавання іншим користувачам ролі.

При запуску сайту користувач буде бачити основну сторінуку, сторінку магазину. Де може перегладати товари, категорії, сторінки. Але для замовлення товарів йому потрібно зареєструватися.

## Загальна схема роботи програми :

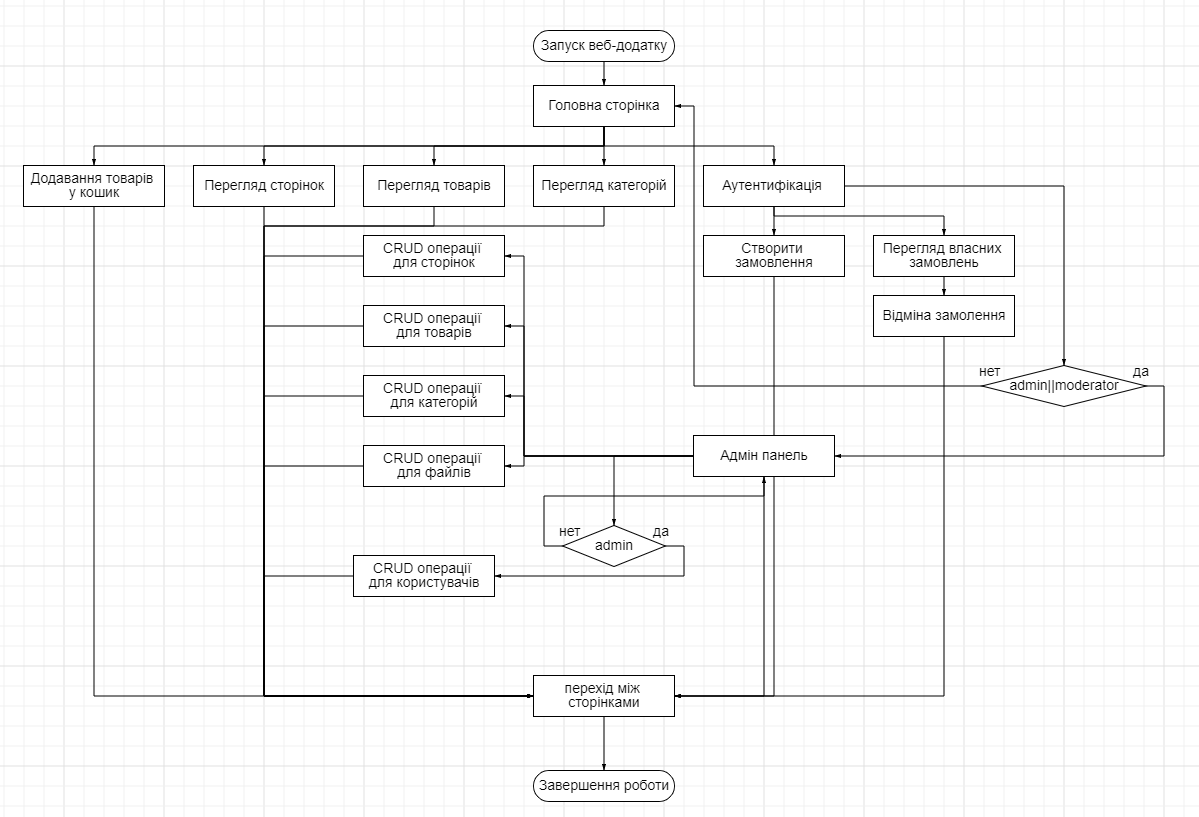


Рисунок 2.1 – Загальна схема роботи програми

## **2.2 Розробка функціональних алгоритмів роботи програми**

Залишилося розглянути детальніше алгоритми робити основних функцій.

Звичайно, основне призначення інтренет-магазину спортвного харчування. Для додавання товарів потрібно заповнити відповідну форму, після чого товар розміститься на сайті. При заповненні форми будуть з’являтися підзказки про коректність вводу. Після розміщення інший користувач зможе замовити даний товар. Користувач додає товар в корзину, звідки можна оформити замовлення. Для цього потрібно заповнити відповідну форму, вказавши персональні данні та адресу доставки. Адресу можна вказати вручну при замовленні курєром Нової Пошти або якщо на відділення то вибрати своє місто проживання та потім відділення (для реалізації данного функціоналу використовується API Нової Пошти).

Для корректонсті обробки даних реалізована подвійна валідація, як бекенді так і на фронтенді. При будь-якій віправці форми висвітлюється вікно з оповіщення про статус успішності дії. При редагування дані автоматично заповнюють відповідні поля, для полегшення їх зміни.

Основним алгоритмом кожної реалізованої функції є:

отримання даних на фронтенді -> створення запиту -> передача запиту на бекенд -> обробка отриманих даних у відповідних сервісах(взаємодія з БД) -> передача оброблених даних на фронтенд.

## **2.3 Розробка програмного забезпечення**

Більша частина програмного забезпечення розробляється відповідно до алгоритму описаного вище, тому розглянемо цей принцип на прикладі реалізованих методів книжок. Розглянему приклад коду створення товару. Для початку нам потрібно створити Input Type, де описано які поля будуть прийматися. Якщо передати щось зайве, або не додати якогось поля – видасться виключення.

@InputType()  
export class CreateProductInput {  
 @Field(() => ***Boolean***)  
 isShown: boolean;  
  
 @Field()  
 name: string;  
  
 @Field(() => ***Number***)  
 quantity: number;  
  
 @Field(() => ***Number***)  
 priceUAH: number;  
  
 @Field(() => ***String***)  
 description: string;  
  
 @Field(() => [CharacteristicInputType], { nullable: true })  
 characteristics: CharacteristicInputType[];  
  
 @Field(() => [UpdateFileInput])  
 files: UpdateFileInput[];  
  
 @Field(() => [UpdateCategoryInput])  
 categories: UpdateCategoryInput[];  
}

async createAsync(createProductInput: CreateProductInput): Promise<Product> {  
 const product = this.productsRepository.create(createProductInput);  
 product.slug = getSlug(product.name);  
 return await this.productsRepository.save(product);  
}

Також важливим компонетом веб додатку є автентифікація, реалізована на основі JWT. Роглянемо її код:

@Resolver()  
export class AuthResolver {  
 constructor(  
 private authService: AuthService,  
 private usersService: AdminUsersService,

) {  
 }  
  
 @Mutation(() => AuthResponse)  
 async login(@Args('loginInput') loginInput: LoginInput): Promise<AuthResponse> {  
 const user = await this.authService.validateUserAsync(loginInput.email, loginInput.password);  
 if (!user)  
 throw new ***Error***('Bad credentials');  
  
 if (!user.confirmedEmail) {  
 await this.authService.sendConfirmationEmailTokenAsync(user);  
 throw new ***Error***('Please confirm your email');  
 }  
  
 const accessToken = await this.authService.getAuthJwtTokenAsync(user.id, user.email, user.roles);  
 const authResponse = new AuthResponse();  
 authResponse.user = user;  
 authResponse.accessToken = accessToken;  
 return authResponse;  
 }  
  
 @Mutation(() => ***String***)  
 async register(  
 @Args('registerInput') registerInput: RegisterInput,  
 ): Promise<string> {  
 if (!await this.authService.isEmailFreeAsync(registerInput.email))  
 throw new ***Error***('Email already taken');  
 const user = await this.authService.registerAsync(registerInput);  
  
 await this.authService.sendConfirmationEmailTokenAsync(user);  
 return 'Please confirm you email';  
 }  
  
 @UseGuards(GqlAuthGuard)  
 @Query(() => AuthResponse)  
 async me(@CurrentUser() user: User): Promise<AuthResponse> {  
 const currentUser = await this.usersService.getByIdAsync(user.id);  
  
 if (!currentUser.confirmedEmail) {  
 await this.authService.sendConfirmationEmailTokenAsync(currentUser);  
 throw new ***Error***('Please confirm your email');  
 }  
  
 const authResponse = new AuthResponse();  
 authResponse.user = await this.usersService.getByIdAsync(user.id);  
 authResponse.accessToken = await this.authService.getAuthJwtTokenAsync(user.id, user.email, user.roles);  
 return authResponse;  
 }  
  
 @UseGuards(GqlAuthGuard)  
 @Mutation(() => AuthResponse)  
 async updateMe(  
 @Args('updateMeInput', {type: () => UpdateMeInput}) updateMeInput: UpdateMeInput,  
 @CurrentUser() currentUser: User,  
 ): Promise<AuthResponse> {  
 await this.authService.updateMeAsync(currentUser.id, updateMeInput);

const authResponse = new AuthResponse();  
 authResponse.user = await this.usersService.getByIdAsync(currentUser.id);  
 authResponse.accessToken = await this.authService.getAuthJwtTokenAsync(currentUser.id, currentUser.email, currentUser.roles);  
 return authResponse;  
 }  
  
 @UseGuards(GqlAuthGuard)  
 @Mutation(() => ***String***)  
 async updateEmail(  
 @Args('updateEmailInput', {type: () => UpdateEmailInput}) updateEmailInput: UpdateEmailInput,  
 @CurrentUser() currentUser: User,  
 ): Promise<string> {  
 if (!await this.authService.isEmailFreeAsync(updateEmailInput.email))  
 throw new ***Error***('Email already taken');  
 const user = await this.authService.updateEmailAsync(currentUser.id, updateEmailInput);  
 await this.authService.sendConfirmationEmailTokenAsync(user);  
 return 'You successfully updated email. Please confirm you email';  
 }  
  
 @UseGuards(GqlAuthGuard)  
 @Mutation(() => AuthResponse)  
 async updatePassword(  
 @Args('updatePasswordInput', {type: () => UpdatePasswordInput}) updatePasswordInput: UpdatePasswordInput,  
 @CurrentUser() currentUser: User,  
 ): Promise<AuthResponse> {  
 await this.authService.updatePasswordAsync(currentUser.id, updatePasswordInput.oldPassword, updatePasswordInput.newPassword);  
 const authResponse = new AuthResponse();  
 authResponse.user = await this.usersService.getByIdAsync(currentUser.id);  
 authResponse.accessToken = await this.authService.getAuthJwtTokenAsync(currentUser.id, currentUser.email, currentUser.roles);  
 return authResponse;  
 }  
  
 @Mutation(() => AuthResponse)  
 async confirmationEmail(  
 @Args('token', {type: () => ***String***}) token: string,  
 ): Promise<AuthResponse> {  
 let user: User = await this.authService.validateConfirmEmailJwtTokenAsync(token);  
 if (!user)  
 throw new ***Error***('Token is not valid');  
  
 user = await this.usersService.confirmEmailAsync(user.email);  
 const authResponse = new AuthResponse();  
 authResponse.user = user;  
 authResponse.accessToken = await this.authService.getAuthJwtTokenAsync(user.id, user.email, user.roles);  
 return authResponse;  
 }  
}

А

А також розглянемо авторизацію користувачів, щоб мати права доступу до різних резолверів:

@Injectable()  
export class RolesGuard implements CanActivate {  
 constructor(private reflector: Reflector) {  
 }  
  
 canActivate(context: ExecutionContext): boolean {  
 // @ts-ignore  
 const requiredRoles = this.reflector.getAllAndOverride<Role[]>('roles', [  
 context.getHandler(),  
 context.getClass(),  
 ]) as string[];  
  
 if (!requiredRoles)  
 return true;  
  
 const gqlReq = GqlExecutionContext.*create*(context).getContext().req;  
 let userRoles: string[];  
 if (gqlReq) {  
 userRoles = gqlReq.user.roles.map(role => role.name);  
 } else {  
 const {user} = context.switchToHttp().getRequest();  
 userRoles = user?.roles.map(role => role.name);  
 }  
 return requiredRoles.some((role) => userRoles.includes(role));  
 }  
}

## **Висновки до 2 розділу :**

Спроектовано загальну схему програми. Визначено та детально описано основні можливості програми. Пояснено алгоритми найважливіших методів інтернет-магазину та приведено код їх основних частин. Приведено приклади коду деяких методів. Роз’яснено деякі нюанси роботи програми. Показано структуру КР.

# РОЗДІЛ 3 ОПИС РОБОТИ З ДОДАТКОМ

## **3.1 Опис роботи з додатком (Опис інтерфейсу)**

Після запуску програми з’являється головна сторінка (Рис 3.1), де користувач бачить список книг та може тільки їх проглядати.

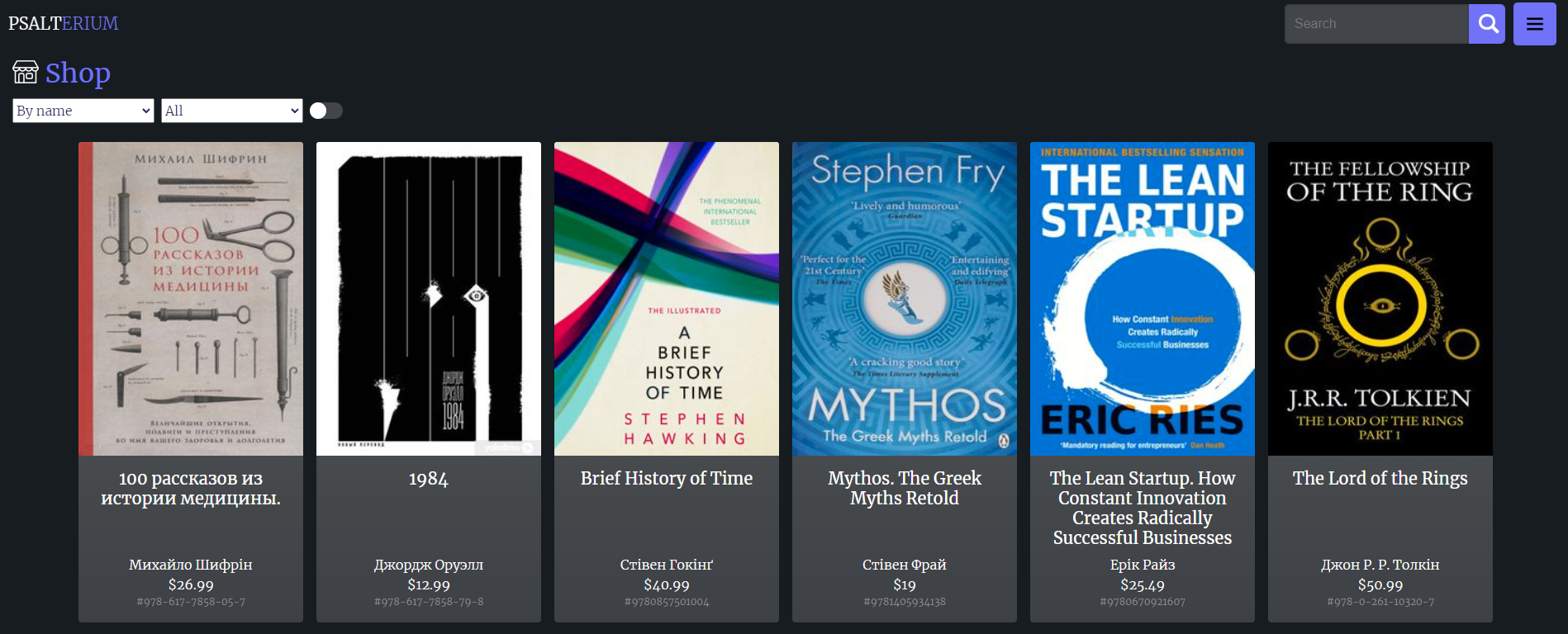


Рис 3.1 Головна сторінка

Головна сторінка має функції переходу до конкретних книжок, сортування та пошуку. Також реалізована динамічна пагінація у виді бескінечної ленти. У верхньому правому куті розміщений пошук та меню, де користувач може побачити наявні сторінки.

Перейдемо до конкретного алгоритму роботи х веб додатком.

Спочатку користувач обирає книгу, наприклад, книгу “Чистий код”. При натиску на карточку з відповідною книгою, він переходить на сторінку книги (Рис 3.3), де він може прочитати інформацію про книгу більш детально та додати деяку кількість книг до власної корзини. Після успішного додання книги до корзини, користувач бачить повідомлення:

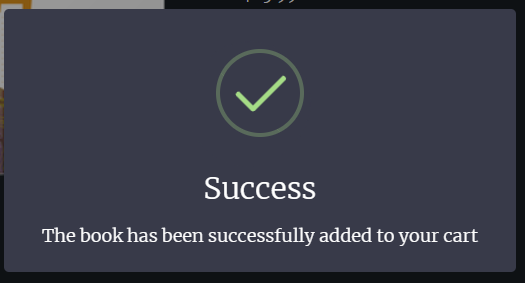


Рис 3.2 Сторінка конкретної книги

Далі користувач може переглядати інші книги та додавати до корзини або оформити замовлення.

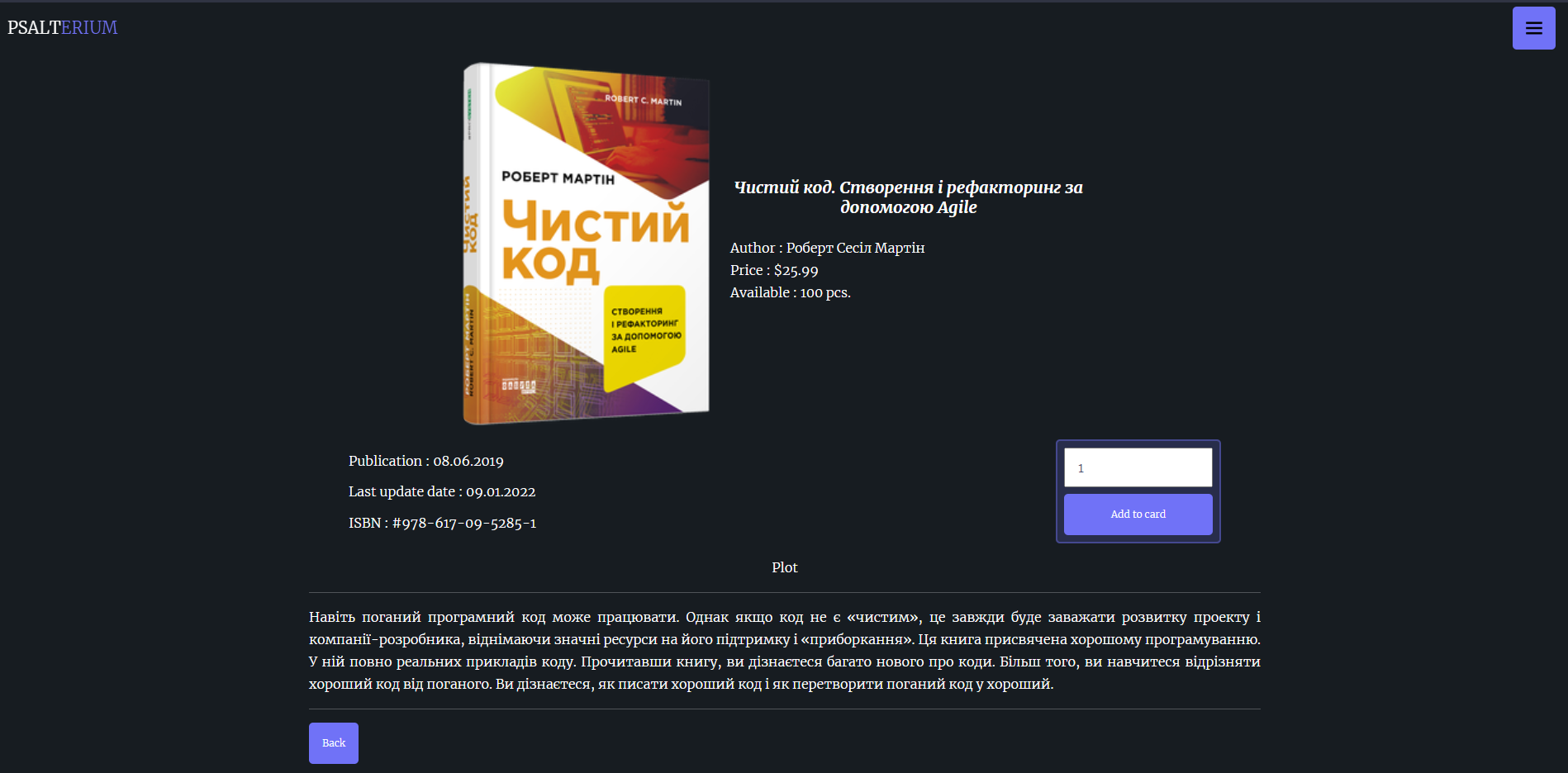


Рис 3.3 Сторінка конкретної книги

Для прикладу оберемо ще одну книгу(“1984”). Додали до корзини 3 екземпляри. Далі користувач переходить до корзини для оформлення замовлення(Рис 3.4). Як можемо бачити, тут відображаються всі нами обрані книги.

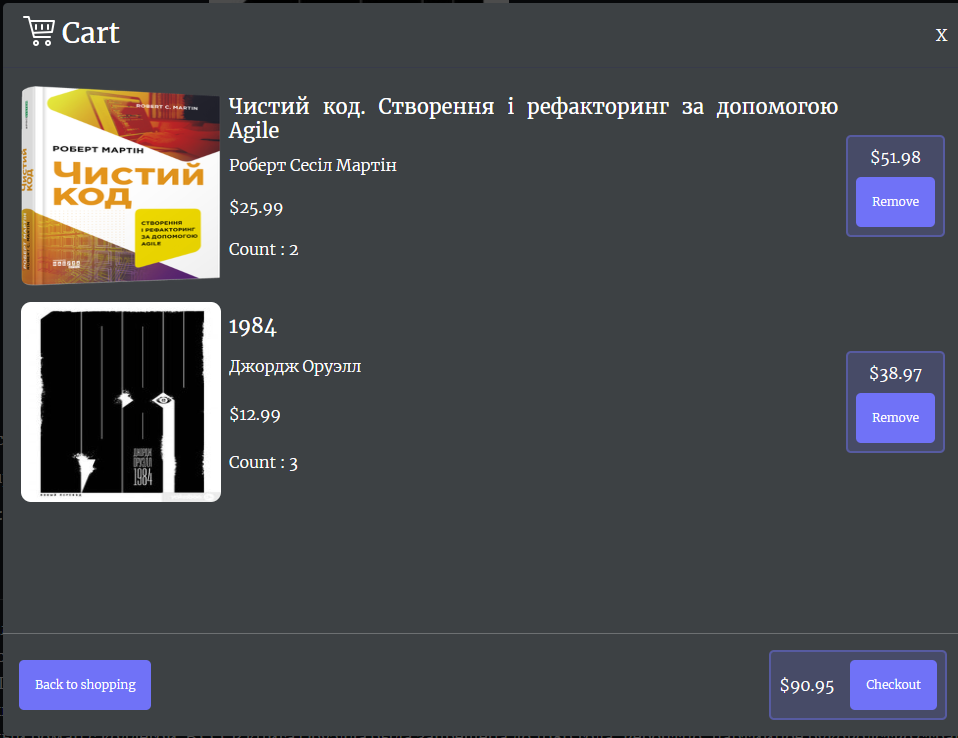


Рис 3.4 Корзина

Також користувач може видалити з корзини книгу, натиснувши на кнопку зліва від книги. Над кнопкою також відображена ціна за всі екземпляри книги. Якщо користувач впевнений, що йому саме ці книги потрібні, він переходить до оформлення замовлення(Рис 3.5). Після оформлення форми замовлення додається в список замовлень(Рис 3.6).

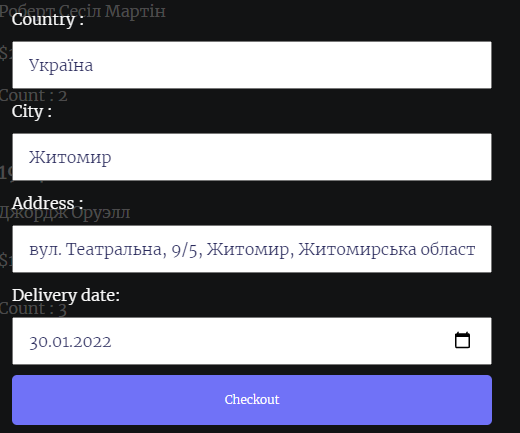


Рис 3.5 Форма оформлення замовлення

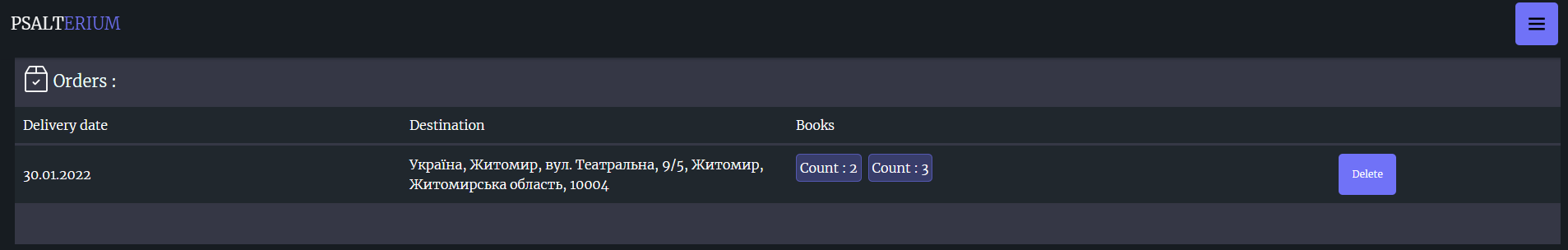


Рис 3.6 Список замовлень

На сторінці список замовлень користувач може відміняти замовлення та переглянути замовлені книги при кліку на підсвічені елементи, які ведуть на сторінки відповідних книжок. На даній сторінці адміну будуть показуватися всі замовлення, а звичайному користувачу тільки його власні.

Далі перейдемо до дадаткової можливості інтернет-магазину – виставлення на продаж власних книг.

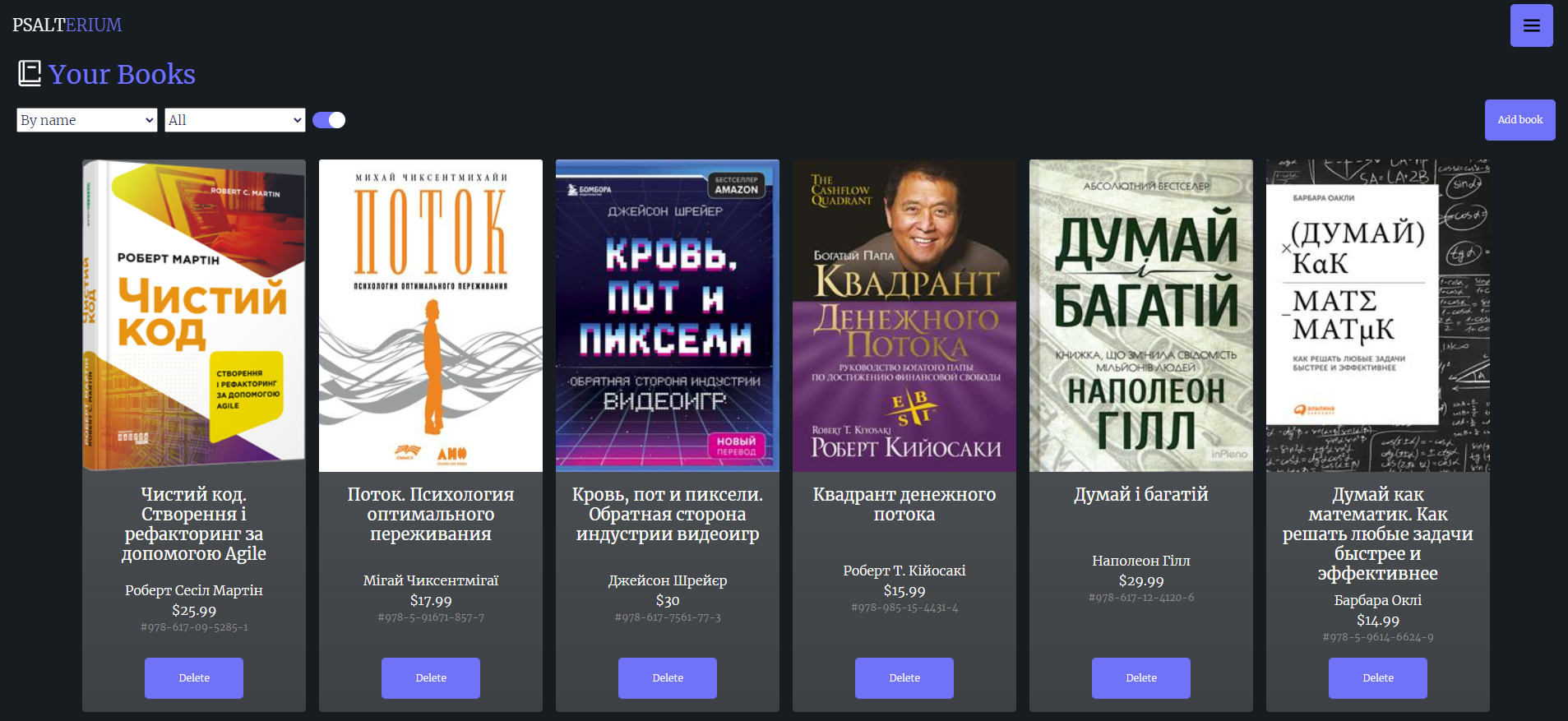


Рис 3.7 Сторінка книг виставлених на продаж користувачем

На даній сторінці у верхньому правому куті розміщена кнопка, яка відкриває форму для книги(Рис. 3.8). Коли користувач заповнить цю форму, книга відобразиться на сторінці та додасться до магазину, де інший користувач зможе її замовити.

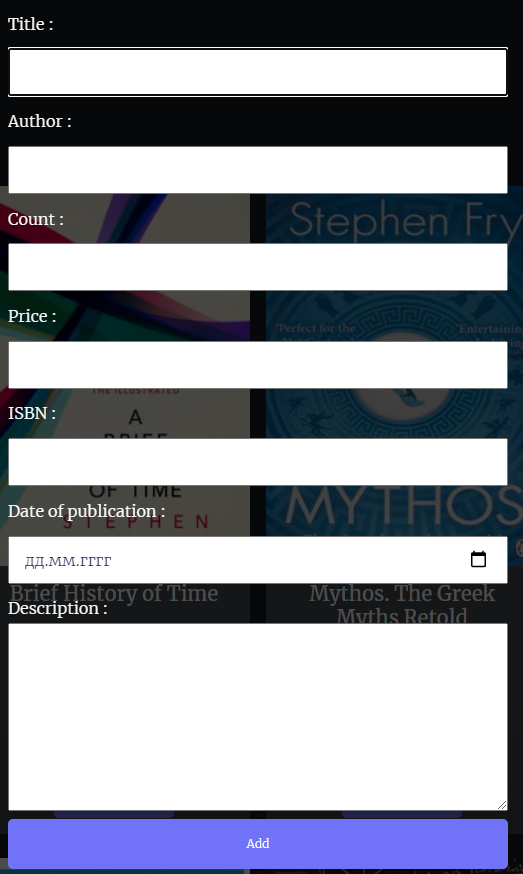


Рис 3.7 Форма для виставлення книг на продаж

Після виставлення книги, її користувач може перейти на неї і побачити додатковий функціонал, а саме можливість зміни ланих про книгу та задати їй жанри.

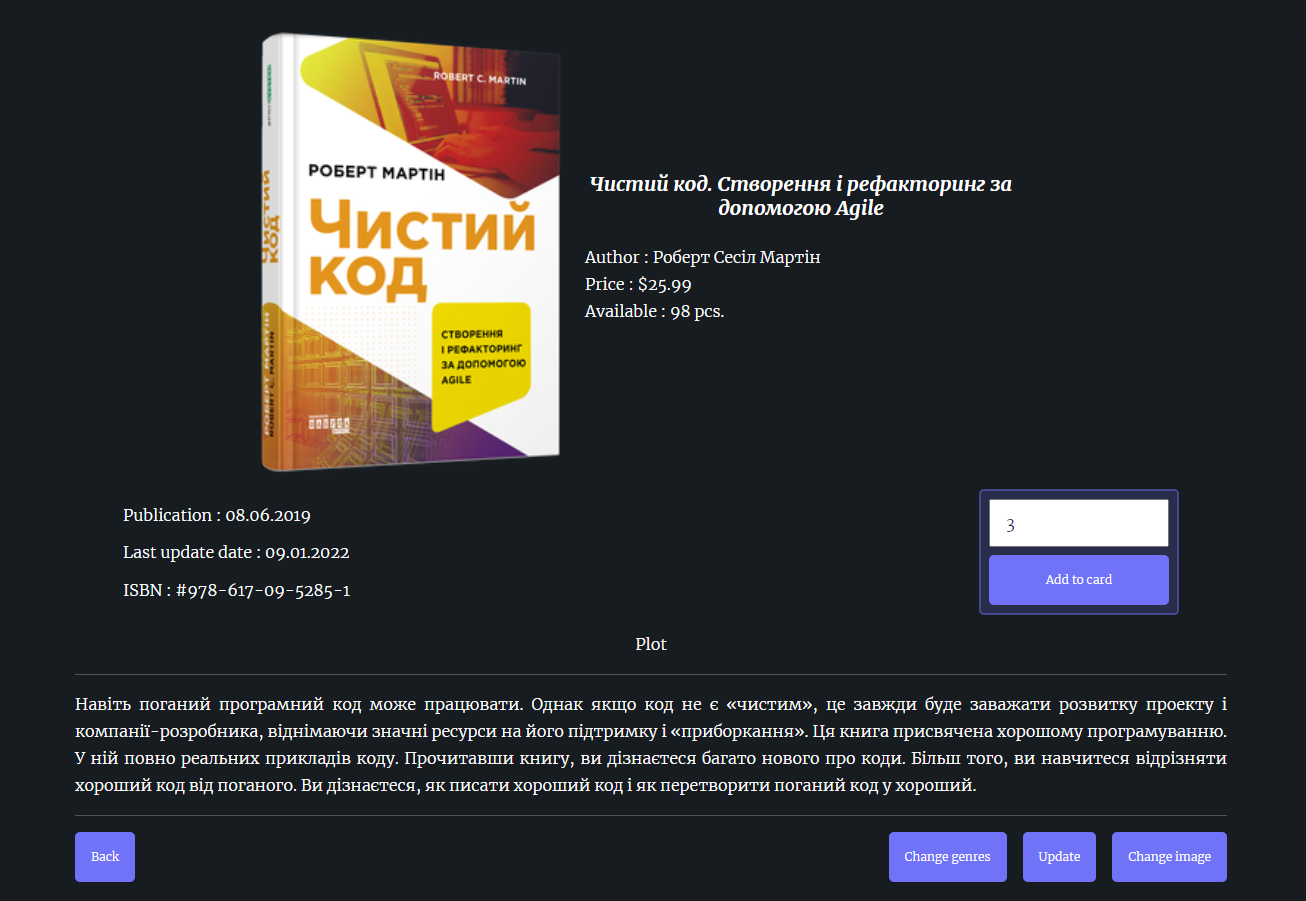


Рис 3.8 Сторінка книги з розширеним функціоналом

Користувач може перейти на власну сторінку(Рисю 3.9) і змінити про себе дані та пароль.

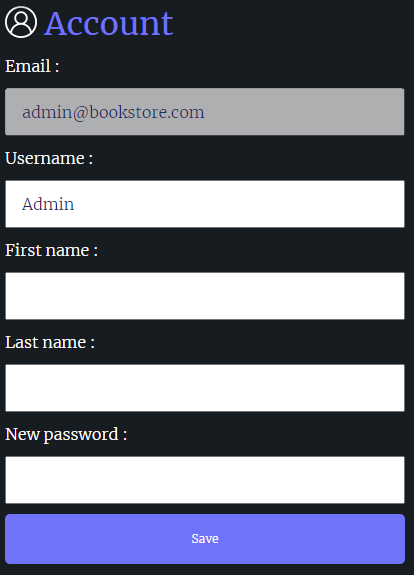


Рис 3.9 Форма для зміни інформації про користувача

Також не можна не розглянути функціонал який доступний тільки для адмінів, а саме сторінку адміністрації(Рис. 3.10).

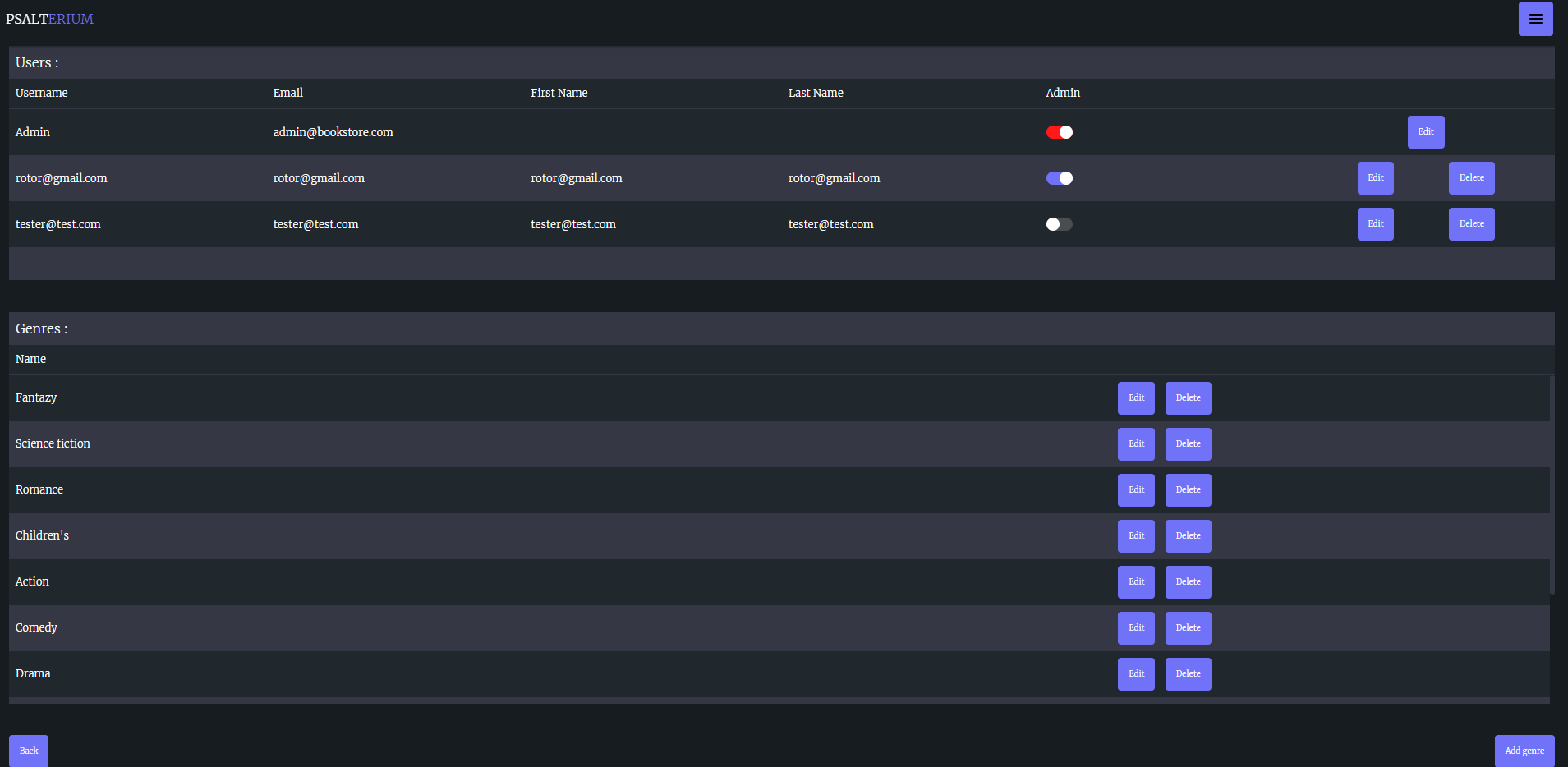


Рис 3.10 Сторінка адміністрації

На сторінку адміністрації мають доступ тільки адміністратори, як не дивно. На даній сторінці адміністратор може переглядати та змінювати дані про користувачів та жанри. Також має можливість видаляти і користувачів і жанри. Користувачі створюються при реєстрації, а для створення жанрів зліва знизу розміщена кнопка для відкриття форми створення жанру(Рис 3.11). Також адмін має можливість надати права адміна іншому користувачу.

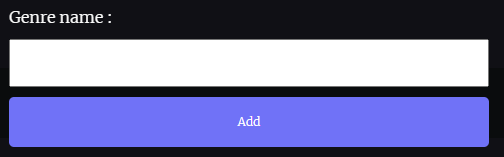


Рис 3.10 Форма створення жанру

Після кожної успішної дії користувач отримує повідомлення про успішність виконання. Розглянемо на прикладі створення нового жанру.

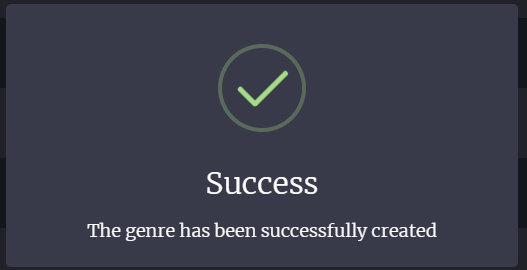


Рис 3.11 Повідомлення про успішне створення жанру

## **Тестування веб-додатку**

Веб-додаток відображає попередження про помилку при при введенні даних для зручного користування користувачем. Наприклад, при спробі створення “пустого” користувача відображаються попередження про обов’язковість полів(Рис 3.12).

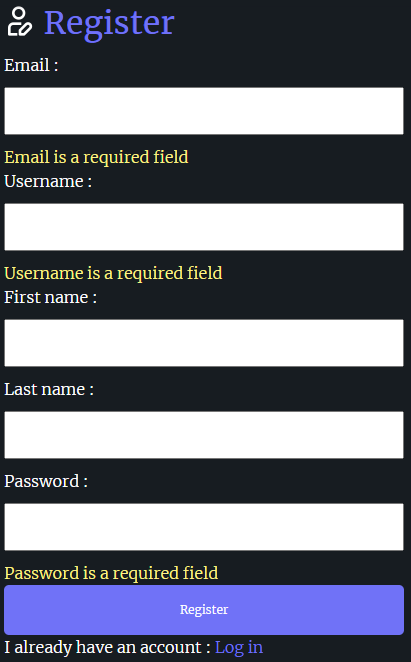


Рис 3.12 Повідомлення про необхідність заповнення

При вводі некоректного значення, з’являється попередження(Рис 3.13)

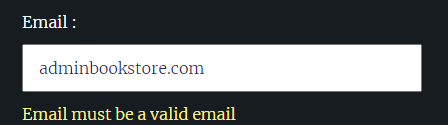


Рис 3.13 Повідомлення про некоректність вводу

При вводі даних які вже існують і не можуть дублюватися, з’являється попередження про існування(Рис 3.14)

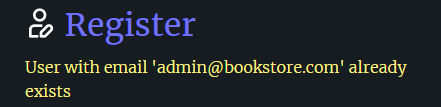


Рис 3.14 Повідомлення про існування користувача

При відправці форми з некоректними даними аутентифікації, з’являється попередження некоректність(Рис 3.15)

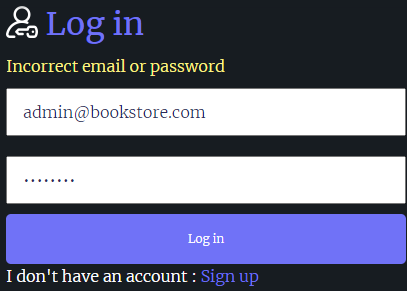


Рис 3.15 Повідомлення некоректність email або пароля

Подібна валідація реалізована на всіх формах

## **Висновки до 3 розділу :**

Було детально описано роботу програмного додатку, користувацький інтефейс, очікувані дії користувача та їх результати. Також було розписано моменти уникнення помилок завдяки допоміжним перевіркам при введенні даних або при роботі з зображеннями. Завдяки цим перевіркам вдалося уникнути частих та серйозних помилок. Наведено приклади обробки помилок.

# **ВИСНОВКИ**

Під час написання даної курсової роботи було отримано навички роботи з розробки веб-додатку.

В першому розділі курсової роботи, було проаналізовано знайдені інтернет-магазини зі схожим функціоналом та інтерфейсом, та було сформовано уявлення про інтернет-магазин, що розроблюється. Було виявлено основні потреби для написання даного веб-додатку.

В другому розділі курсової роботи, було проведено розробку функціональних і загальних частин веб-додатку. При розробці функціонального алгоритму, розробили уявлення про методи розробки програмного коду. На заключному етапі, розроблено його. В результаті, отримано готовий програмний продукт, який виконуває всі заплановані функції і задачі.

В третьому розділі курсової роботи було описана методи користування створеним інтернет-магазином, були пояснені і продемонстровані всі створені елементи інтерфейсу. Також було проведено тестування, тобто інтернет-магазин було запущено в реальних умовах її користування. В ході тестування проблем не виникало.

В результаті виконання курсової роботи отримано інтернет-магазин книжок, який повністю відповідає запланованому функціоналу, умовам та вимогам, поставленим на початку проектування та привищує його. Інтернет-магазин є зрозумілим та зручним в користуванні, адаптивним дял різних пристроїв та швидкодіючим.

# **СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. Документація React [електронний ресурс]. Режим доступу: <https://reactjs.org/> .
2. Документація Redux [електорнний ресурс]. Режим доступу: <https://redux.js.org/> .
3. Документація Axios [електорнний ресурс]. Режим доступу: <https://axios-http.com/docs/intro>.
4. Документація Ant Design [електорнний ресурс]. Режим доступу: <https://ant.design/components/overview/> .
5. Документація Apollo Client [електорнний ресурс]. Режим доступу: https://www.apollographql.com/docs/react/.
6. Документація Nest JS [електорнний ресурс]. Режим доступу: <https://docs.nestjs.com/> .
7. Документація MySQL [електронний ресурс]. Режим доступу: <https://dev.mysql.com/doc/> .
8. Документація Graph QL [електорнний ресурс]. Режим доступу: https://graphql.org/code/#javascript .
9. Документація TypeORM [електорнний ресурс]. Режим доступу: https://typeorm.io/#/.
10. Документація Passport JWT [електорнний ресурс]. Режим доступу: https://www.passportjs.org/packages/passport-jwt/ .