

ДЕРЖАВНИЙ УНІВЕРСИТЕТ  
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ  
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ  
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ  
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

**КВАЛІФІКАЦІЙНА РОБОТА**

на тему: «Онлайн-магазин одягу "YNG-YOUNGSHOP"»  
для підлітків»

на здобуття освітнього ступеня бакалавра  
зі спеціальності 121 Інженерія програмного забезпечення  
освітньо-професійної програми «Інженерія програмного забезпечення»

*Кваліфікаційна робота містить результати власних досліджень.  
Використання ідей, результатів і текстів інших авторів мають посилання на  
відповідне джерело*

Олексій ФЕДОРЕНКО

\_\_\_\_\_  
(підпис)

Виконав: здобувач вищої освіти групи ПД-42

Олексій ФЕДОРЕНКО

Керівник: Ігор ГАМАНЮК  
ст. викладач

Рецензент: \_\_\_\_\_

**ДЕРЖАВНИЙ УНІВЕРСИТЕТ  
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ**  
**Навчально-науковий інститут інформаційних технологій**

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Бакалавр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

**ЗАТВЕРДЖУЮ**

Завідувач кафедри

Інженерії програмного забезпечення

\_\_\_\_\_ Ірина ЗАМРІЙ

« \_\_\_\_ » \_\_\_\_\_ 2026 р.

**ЗАВДАННЯ  
НА КВАЛІФІКАЦІЙНУ РОБОТУ**

Федоренку Олексію Ігоровичу

1. Тема кваліфікаційної роботи: «Онлайн-магазин одягу "YNG-YOUNGSHOP" для підлітків»

керівник кваліфікаційної роботи ст. викладач кафедри ІІЗ Ігор ГАМАНЮК,  
затверджені наказом Державного університету інформаційно-комунікаційних  
технологій від «20» лютого 2026 року № 51.

2. Строк подання кваліфікаційної роботи «29» травня 2026 року.

3. Вихідні дані до кваліфікаційної роботи: кваліфікаційної роботи є сучасні технології веб-розробки, зокрема React, Node.js, Express та MongoDB, а також результати аналізу предметної галузі електронної комерції та існуючих онлайн-магазинів одягу.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Аналіз існуючих програмних рішень та веб-застосунків для продажу одягу та електронної комерції;
2. Визначення функціональних та нефункціональних вимог до програмного забезпечення;

3. Проектування та програмна реалізація веб-застосунку «YNG-YOUNGSHOP» із використанням React, Node.js, Express та бази даних MongoDB:
4. Аналіз результатів розробки та тестування системи:
5. Перелік графічного матеріалу: *презентація*
  1. Аналіз аналогів онлайн-магазинів електронної комерції.
  2. Функціональні та нефункціональні вимоги до програмного забезпечення.
  3. Програмні засоби та технології реалізації вебзастосунку.
  4. Алгоритм роботи застосунку та моделювання взаємодії користувача із системою.
  5. Діаграма класів основних сутностей системи.
  6. Екранні форми вебзастосунку онлайн-магазину YNG-YOUNGSHOP».
  7. Апробація результатів дослідження.
  8. Тестування
6. Дата видачі завдання «24» квітня 2026 року.

### КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів кваліфікаційної роботи                  | Строк виконання етапів роботи | Примітка |
|-------|------------------------------------------------------|-------------------------------|----------|
| 1     | Аналіз предметної галузі та існуючих аналогів        | 01.02-10.02.2026              |          |
| 2     | Формування вимог до програмного забезпечення         | 11.02-18.02.2026              |          |
| 3     | Проектування структури та архітектури веб-застосунку | 19.02-05.03.2026              |          |
| 4     | Розробка клієнтської частини системи                 | 06.03-25.03.2026              |          |
| 5     | Розробка серверної частини та бази даних             | 26.03-15.04.2026              |          |
| 6     | Реалізація функціоналу авторизації та кошика         | 16.04-30.04.2026              |          |
| 7     | Тестування веб-застосунку                            | 01.05-05.05.2026              |          |
| 8     | Оформлення роботи: вступ, висновки, реферат          | 06.05-10.05.2026              |          |
| 9     | Попередній захист роботи                             | 11.05-22.05.2026              |          |
| 10    | Подання кваліфікаційної роботи                       | 23.05-29.05.2026              |          |

Здобувач вищої освіти

(підпис)

Олексій ФЕДОРЕНКО

Керівник  
кваліфікаційної роботи

(підпис)

Ігор ГАМАНЮК





## РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 70 стор., 4 табл., 20 рис., 25 джерел.

*Мета роботи* – покращення користувацького досвіду під час взаємодії із системою електронної комерції шляхом створення адаптивного онлайн-магазину одягу для підлітків із сучасним інтерфейсом та зручним функціоналом

*Об'єкт дослідження* – процес функціонування онлайн-магазину одягу для підлітків

*Предмет дослідження* – методи, засоби та технології створення веб-застосунку онлайн-магазину одягу для підлітків, а також особливості реалізації користувацького інтерфейсу, серверної частини та взаємодії із базою даних.

*Короткий зміст роботи:* у роботі проаналізовано сучасні веб-технології та існуючі онлайн-магазини одягу. Проведено аналіз аналогів електронної комерції, зокрема ZARA, Bershka та ASOS. Розроблено структуру та архітектуру веб-застосунку «YNG-YOUNGSHOP». Програмно реалізовано основні функціональні можливості системи: перегляд товарів, пошук продукції, реєстрація та авторизація користувачів, формування кошика та оформлення замовлення. Для створення веб-застосунку використано React для клієнтської частини, Node.js та Express для серверної логіки, MongoDB для зберігання даних. Проведено функціональне тестування програмного забезпечення та перевірку працездатності основних модулів системи.

Сферою використання застосунку є організація онлайн-продажу молодіжного одягу та автоматизація процесу електронної комерції.

КЛЮЧОВІ СЛОВА: ІНТЕРНЕТ-МАГАЗИН, REACT, NODE.JS, MONGODB, JAVASCRIPT, E-COMMERCE, ОНЛАЙН-МАГАЗИН ОДЯГУ.

# ЗМІСТ

|                                                                                       |           |
|---------------------------------------------------------------------------------------|-----------|
| <b>ВСТУП.....</b>                                                                     | <b>10</b> |
| <b>1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА АНАЛІЗ АНАЛОГІВ .....</b>                      | <b>13</b> |
| <b>1.1. Особливості сучасних онлайн-магазинів одягу.....</b>                          | <b>13</b> |
| <b>1.2. Аналіз тенденцій електронної комерції.....</b>                                | <b>16</b> |
| <b>1.2. Аналіз існуючих веб-застосунків для продажу одягу .....</b>                   | <b>18</b> |
| 1.3.1 Онлайн-магазин ZARA .....                                                       | 20        |
| 1.3.2 Онлайн-магазин ASOS .....                                                       | 23        |
| 1.3.3 Онлайн-магазин Bershka .....                                                    | 25        |
| 1.3.4 Порівняння функціональних можливостей онлайн-магазинів .....                    | 27        |
| <b>1.4 Формування вимог до веб-застосунку.....</b>                                    | <b>29</b> |
| <b>2 ВИБІР ТЕХНОЛОГІЙ ТА ЗАСОБІВ РОЗРОБКИ.....</b>                                    | <b>32</b> |
| <b>2.1. Середовище розробки .....</b>                                                 | <b>33</b> |
| 2.1.1 Visual Studio Code.....                                                         | 34        |
| 2.1.2 Figma.....                                                                      | 36        |
| <b>2.2 Технології клієнтської частини .....</b>                                       | <b>37</b> |
| 2.2.1 React.....                                                                      | 38        |
| 2.2.2 Tailwind CSS .....                                                              | 39        |
| <b>2.3 Технології серверної частини.....</b>                                          | <b>41</b> |
| 2.3.1 Node.js.....                                                                    | 42        |
| 2.3.2 Express.js.....                                                                 | 43        |
| <b>2.4. Система управління базами даних.....</b>                                      | <b>44</b> |
| <b>2.5. Система контролю версій .....</b>                                             | <b>45</b> |
| 2.5.1 Git.....                                                                        | 46        |
| 2.5.2 GitHub.....                                                                     | 47        |
| <b>3. ПРОЄКТУВАННЯ СИСТЕМИ.....</b>                                                   | <b>48</b> |
| <b>3.1. Проєктування архітектури веб-застосунку .....</b>                             | <b>49</b> |
| 3.1.1 Діаграма варіантів використання.....                                            | 51        |
| 3.1.2 Діаграма станів та предметної галузі .....                                      | 53        |
| <b>3.2. Проєктування структури клієнтської частини .....</b>                          | <b>54</b> |
| <b>3.3. Проєктування бази даних .....</b>                                             | <b>57</b> |
| <b>3.4 Організація взаємодії між клієнтською та серверною частинами системи .....</b> | <b>59</b> |
| <b>4. ПРАКТИЧНА РЕАЛІЗАЦІЯ ВЕБ-ЗАСТОСУНКУ .....</b>                                   | <b>62</b> |
| <b>4.1 Створення клієнтської частини системи.....</b>                                 | <b>62</b> |

|                                                      |           |
|------------------------------------------------------|-----------|
| 4.2 Розробка backend-частини веб-застосунку .....    | 65        |
| 4.3 Реалізація бази даних та роботи з товарами ..... | 66        |
| 4.4 Аналіз роботи та тестування системи .....        | 69        |
| <b>ВИСНОВКИ .....</b>                                | <b>71</b> |
| <b>ДОДАТОК А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ .....</b>     | <b>75</b> |
| <b>ДОДАТОК Б. ЛІСТИНГИ ПРОГРАМНИХ МОДУЛІВ .....</b>  | <b>83</b> |

## **ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ**

API – Application Programming Interface

CSS – Cascading Style Sheets

CRUD – Create, Read, Update, Delete

DB – Database

HTML – HyperText Markup Language

HTTP – HyperText Transfer Protocol

IDE – Integrated Development Environment

JSON – JavaScript Object Notation

JWT – JSON Web Token

NoSQL – Not Only SQL

REST – Representational State Transfer

SPA – Single Page Application

UI – User Interface

UML – Unified Modeling Language

URL – Uniform Resource Locator

UX – User Experience

VS Code – Visual Studio Code

npm – Node Package Manager

## ВСТУП

У сучасному світі інформаційні технології стали невід'ємною частиною повсякденного життя людини. Стрімкий розвиток мережі Інтернет, цифрових платформ та мобільних пристроїв суттєво змінив способи взаємодії користувачів із різноманітними сервісами та інформаційними системами. Однією з галузей, яка найбільше зазнала впливу цифровізації, є сфера торгівлі. Традиційні методи продажу товарів поступово доповнюються або повністю замінюються сучасними електронними сервісами, що дозволяють здійснювати покупки дистанційно за допомогою вебзастосунків та мобільних платформ.

Електронна комерція сьогодні є одним із найпоширеніших напрямків розвитку цифрового бізнесу. Онлайн-магазини дозволяють автоматизувати процес продажу товарів, забезпечити постійний доступ користувачів до каталогу продукції та значно спростити процес взаємодії між продавцем і покупцем. Використання вебзастосунків дає можливість здійснювати покупки у будь-який час та з будь-якого пристрою, що має доступ до мережі Інтернет. Це значно підвищує комфорт користувачів та ефективність роботи бізнесу.

Особливо популярними онлайн-магазини є серед молоді та підлітків. Молодіжна аудиторія активно використовує цифрові технології, соціальні мережі та мобільні застосунки для щоденного спілкування, отримання інформації та здійснення покупок. Саме тому сучасні інтернет-магазини повинні не лише забезпечувати базові функції електронної комерції, але й відповідати актуальним вимогам щодо дизайну, швидкодії, адаптивності та зручності користування.

Одним із найбільш популярних напрямків онлайн-продажу є реалізація одягу та аксесуарів. Інтернет-магазини одягу дозволяють користувачам переглядати великий каталог товарів, використовувати систему пошуку та фільтрації, оформлювати замовлення та здійснювати оплату без необхідності фізичного відвідування магазину. Для молодіжної аудиторії важливу роль

відіграє не лише функціональність вебзастосунку, але й його візуальний стиль, сучасний інтерфейс та зручна навігація.

На сьогодні існує велика кількість онлайн-магазинів одягу, серед яких можна виділити такі популярні платформи, як ZARA, Bershka, ASOS та інші. Дані сервіси мають широкий функціонал, адаптивний дизайн та сучасний користувацький інтерфейс. Проте більшість існуючих рішень орієнтовані на масовий ринок та не завжди враховують індивідуальні потреби молодіжної аудиторії, зокрема у сфері streetwear та альтернативного стилю одягу.

У зв'язку з цим актуальною є задача створення сучасного веб-застосунку онлайн-магазину одягу для підлітків, який поєднуватиме сучасний дизайн, зручність використання та необхідний функціонал електронної комерції. Такий застосунок повинен забезпечувати швидкий доступ до каталогу товарів, можливість реєстрації та авторизації користувачів, формування кошика покупок, оформлення замовлень та зручну взаємодію з системою через сучасний вебінтерфейс.

Актуальність теми кваліфікаційної роботи полягає у необхідності розробки сучасного онлайн-магазину одягу для підлітків із використанням актуальних технологій веброзробки. Використання сучасних інструментів програмування дозволяє створити масштабований, адаптивний та продуктивний веб-застосунок, який відповідатиме сучасним вимогам електронної комерції.

Для реалізації системи було обрано сучасний стек вебтехнологій, який включає React для створення клієнтської частини, Node.js та Express для реалізації серверної логіки, а також MongoDB для організації бази даних. Використання даних технологій дозволяє забезпечити високу швидкодію застосунку, гнучкість структури системи та ефективну взаємодію між клієнтом і сервером через API.

Метою покращення користувацького досвіду під час взаємодії із системою електронної комерції шляхом створення адаптивного онлайн-магазину одягу для підлітків із сучасним інтерфейсом та зручним функціоналом.

Для досягнення поставленої мети необхідно виконати такі завдання:

- провести аналіз предметної галузі електронної комерції;
- дослідити існуючі онлайн-магазини одягу та визначити їх особливості;
- сформулювати функціональні та нефункціональні вимоги;
- розробити структуру та архітектуру системи;
- реалізувати клієнтську частину;
- реалізувати серверну частину системи та базу даних;
- створити систему авторизації користувачів;
- реалізувати функціонал кошика та оформлення замовлень;
- забезпечити адаптивний дизайн інтерфейсу;
- провести тестування функціональних можливостей «YNG-YOUNGSHOP».

Об'єктом дослідження є процес функціонування онлайн-магазину одягу для підлітків.

Предметом дослідження методи, засоби та технології створення веб-застосунку онлайн-магазину одягу для підлітків, а також особливості реалізації користувацького інтерфейсу, серверної частини та взаємодії із базою даних.

## 1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА АНАЛІЗ АНАЛОГІВ

### 1.1. Особливості сучасних онлайн-магазинів одягу

Сучасний розвиток цифрових технологій суттєво вплинув на сферу електронної комерції та способи взаємодії між продавцем і покупцем. Одним із найбільш популярних напрямків електронної торгівлі стали онлайн-магазини одягу, які дозволяють користувачам здійснювати покупки через мережу Інтернет без необхідності фізичного відвідування торгових центрів чи магазинів [1].

Онлайн-магазин одягу являє собою веб-застосунок, основною функцією якого є автоматизація процесів продажу товарів. Такі системи забезпечують можливість перегляду каталогу продукції, пошуку товарів, фільтрації за категоріями, оформлення замовлень та взаємодії користувача із системою через сучасний графічний інтерфейс.

На сьогодні електронна комерція є одним із найбільш динамічних секторів цифрової економіки. Поширення мобільних пристроїв та постійний доступ до мережі Інтернет сприяли збільшенню популярності веб-застосунків для онлайн-покупок. Користувачі все частіше надають перевагу саме онлайн-магазинам завдяки швидкому доступу до товарів, зручності оформлення замовлень та економії часу [2].

Особливо популярними онлайн-магазини є серед молодіжної аудиторії. Молодь активно використовує смартфони, соціальні мережі та сучасні вебсервіси у повсякденному житті. Для даної категорії користувачів важливу роль відіграють не лише функціональні можливості системи, але й візуальний стиль, швидкодія та адаптивність інтерфейсу.

Сучасний онлайн-магазин одягу повинен забезпечувати:

- перегляд каталогу товарів;
- систему категорій та фільтрації;
- пошук продукції;

- реєстрацію та авторизацію користувачів;
- кошик покупок;
- оформлення замовлень;
- адаптивний дизайн;
- систему адміністрування товарів.

Однією з ключових особливостей сучасних веб-застосунків є адаптивність інтерфейсу. Адаптивний дизайн дозволяє коректно відображати сторінки сайту на різних типах пристроїв: персональних комп'ютерах, планшетах та смартфонах. Значна частина користувачів здійснює покупки саме з мобільних пристроїв, тому адаптивність є важливою вимогою до сучасних онлайн-магазинів [3].

Ще однією важливою характеристикою є швидкодія системи. Користувачі очікують швидкого завантаження сторінок та миттєвої реакції інтерфейсу на дії користувача. Повільна робота веб-застосунку може негативно впливати на користувацький досвід та призводити до втрати потенційних покупців. Саме тому при розробці сучасних онлайн-магазинів значна увага приділяється оптимізації коду та використанню сучасних технологій веб-розробки.

Для створення сучасних веб-застосунків широко використовуються JavaScript-фреймворки та бібліотеки. Одним із найбільш популярних рішень є React - JavaScript-бібліотека для створення користувацьких інтерфейсів [4]. Основною перевагою React є компонентний підхід до побудови інтерфейсу, що дозволяє створювати масштабовані та зручні у підтримці веб-застосунки.

React забезпечує динамічне оновлення даних на сторінці без необхідності її повного перезавантаження. Це позитивно впливає на швидкодію системи та покращує взаємодію користувача із застосунком. Завдяки цьому сучасні онлайн-магазини можуть забезпечувати інтерактивність інтерфейсу та швидке оновлення інформації про товари.

Серверна частина сучасних веб-застосунків часто реалізується з використанням Node.js та Express [5]. Node.js дозволяє виконувати JavaScript-код

на сервері, а Express забезпечує зручний механізм створення REST API та організації взаємодії між клієнтською та серверною частинами системи.

Для зберігання інформації про товари, користувачів та замовлення використовуються сучасні системи управління базами даних. У сфері електронної комерції популярним рішенням є MongoDB - документоорієнтована NoSQL база даних, яка забезпечує гнучкість структури даних та високу продуктивність [7].

Важливою складовою будь-якого онлайн-магазину є система авторизації користувачів. Авторизація дозволяє створювати персональні акаунти, зберігати інформацію про замовлення та забезпечувати персоналізовану взаємодію користувача із системою. Для захисту даних користувачів використовуються механізми шифрування паролів та JWT-токени.

Ще однією ключовою функцією є кошик покупок. Кошик дозволяє користувачам зберігати обрані товари перед оформленням замовлення, змінювати кількість товарів та переглядати загальну вартість покупки. Реалізація такого функціоналу значно покращує зручність використання системи.

Сучасні онлайн-магазини також активно використовують системи пошуку та фільтрації товарів. Дані механізми дозволяють користувачам швидко знаходити необхідну продукцію за категоріями, ціною або іншими характеристиками. Це підвищує ефективність роботи системи та покращує користувацький досвід.

Окрему увагу сучасні веб-застосунки приділяють дизайну інтерфейсу. Молодіжні онлайн-магазини часто використовують мінімалістичний стиль, темні кольорові палітри, великі зображення продукції та анімації. Візуальне оформлення безпосередньо впливає на сприйняття бренду користувачами та формування загального враження від веб-застосунку.

На сьогодні серед найбільш популярних онлайн-магазинів одягу можна виділити ZARA, Bershka та ASOS [23–25]. Дані сервіси мають сучасний адаптивний дизайн, систему авторизації користувачів, кошик покупок та

інтерактивний інтерфейс. Саме тому аналіз їх функціональних можливостей є важливим етапом при розробці власного веб-застосунку.

Сучасні онлайн-магазини одягу є складними інформаційними системами, які поєднують технології веб-розробки, системи управління базами даних та сучасний дизайн інтерфейсу користувача. Використання сучасних технологій дозволяє забезпечити високу продуктивність системи, адаптивність та зручність використання веб-застосунку для користувачів.

## **1.2. Аналіз тенденцій електронної комерції**

Електронна комерція є одним із найважливіших напрямків розвитку сучасних інформаційних технологій та цифрового бізнесу. За останні роки сфера онлайн-торгівлі значно розширилася завдяки розвитку веб-технологій, мобільних платформ та збільшенню кількості користувачів мережі Інтернет. Сучасні онлайн-магазини дозволяють автоматизувати процес продажу товарів, покращити взаємодію з клієнтами та забезпечити швидкий доступ до продукції незалежно від місця перебування користувача [1].

Однією з основних тенденцій розвитку електронної комерції є збільшення кількості покупок, які здійснюються через мобільні пристрої. Більшість користувачів використовує смартфони для перегляду товарів, пошуку продукції та оформлення замовлень. У зв'язку з цим сучасні веб-застосунки повинні підтримувати адаптивний дизайн та коректно працювати на різних типах пристроїв [3].

Ще однією важливою тенденцією є персоналізація користувацького досвіду. Сучасні онлайн-магазини використовують системи рекомендацій, аналіз поведінки користувачів та персоналізований контент для підвищення зацікавленості покупців. Це дозволяє пропонувати користувачам товари відповідно до їх інтересів та попередніх дій у системі [20].

Значну роль у розвитку електронної комерції відіграє швидкодія веб-застосунків. Користувачі очікують миттєвого завантаження сторінок та швидкої

реакції інтерфейсу на дії користувача. Саме тому сучасні системи активно використовують JavaScript-фреймворки, асинхронну взаємодію між клієнтом і сервером та оптимізацію запитів до бази даних [4].

Сучасні веб-застосунки електронної комерції також активно використовують SPA-архітектуру (Single Page Application). Даний підхід дозволяє динамічно оновлювати вміст сторінки без її повного перезавантаження, що значно покращує продуктивність системи та зручність використання. Для реалізації SPA-застосунків широко використовується React [10].

Окрему увагу сучасна електронна комерція приділяє дизайну користувацького інтерфейсу. Молодіжна аудиторія приділяє значну увагу зовнішньому вигляду веб-застосунку, тому сучасні онлайн-магазини використовують мінімалістичний стиль, великі зображення продукції, анімації та інтерактивні елементи інтерфейсу. Дизайн безпосередньо впливає на загальне сприйняття бренду користувачами та їх зацікавленість у використанні системи [18].

Ще однією важливою тенденцією є інтеграція онлайн-магазинів із соціальними мережами. Сучасні бренди активно використовують Instagram, TikTok та інші цифрові платформи для просування товарів та взаємодії з аудиторією. Соціальні мережі дозволяють швидко популяризувати продукцію та залучати нових користувачів до веб-застосунку.

У сфері електронної комерції важливу роль також відіграє безпека даних користувачів. Сучасні веб-застосунки повинні забезпечувати захист персональної інформації, безпечну авторизацію та шифрування конфіденційних даних. Для цього використовуються JWT-токени, HTTPS-протокол та механізми хешування паролів [11].

Ще однією тенденцією розвитку онлайн-магазинів є використання хмарних технологій та масштабованих серверних рішень. Це дозволяє забезпечити стабільну роботу системи навіть при великій кількості користувачів та високому навантаженні на сервер.

Сучасні онлайн-магазини також активно впроваджують системи пошуку та фільтрації товарів. Дані механізми дозволяють користувачам швидко знаходити необхідну продукцію за категоріями, ціною, брендом або іншими характеристиками. Це значно покращує зручність використання веб-застосунку та підвищує ефективність взаємодії користувача із системою.

На сьогодні популярними платформами електронної комерції є ZARA, Bershka та ASOS [23–25]. Дані сервіси мають сучасний адаптивний дизайн, систему авторизації користувачів, кошик покупок, інтерактивний інтерфейс та високий рівень оптимізації. Аналіз таких платформ дозволяє визначити основні тенденції розвитку сучасних веб-застосунків електронної комерції та врахувати їх при розробці власного онлайн-магазину.

Сучасна електронна комерція активно розвивається завдяки використанню новітніх вебтехнологій, адаптивного дизайну, інтерактивних інтерфейсів та систем персоналізації. Використання сучасних технологій веброботки дозволяє створювати швидкі, масштабовані та зручні веб-застосунки для онлайн-продажу товарів.

## **1.2. Аналіз існуючих веб-застосунків для продажу одягу**

Процес Для визначення основних функціональних можливостей та особливостей сучасних онлайн-магазинів було проведено аналіз популярних веб-застосунків для продажу одягу. Аналіз існуючих рішень дозволяє визначити переваги та недоліки сучасних систем електронної комерції, а також сформулювати вимоги до розроблюваного веб-застосунку «YNG-YOUNGSHOP».

Сучасні онлайн-магазини мають подібний базовий функціонал, однак відрізняються дизайном, організацією інтерфейсу, швидкодією та додатковими можливостями. Найбільш популярними веб-застосунками у сфері продажу молодіжного одягу є ZARA, Bershka та ASOS [23–25].

Основними критеріями аналізу веб-застосунків були:

- дизайн інтерфейсу;

- адаптивність;
- швидкодія системи;
- зручність навігації;
- система пошуку товарів;
- система авторизації користувачів;
- кошик покупок;
- оформлення замовлення;
- інтерактивність інтерфейсу.

У результаті аналізу було встановлено, що сучасні онлайн-магазини активно використовують мінімалістичний стиль дизайну, великі фотографії товарів та адаптивну верстку. Значна увага приділяється швидкості взаємодії користувача із системою та зручності навігації між сторінками.

Онлайн-магазин ZARA є одним із найбільш популярних веб-застосунків у сфері продажу одягу [23]. Даний сервіс має сучасний мінімалістичний дизайн, адаптивний інтерфейс та систему швидкого пошуку товарів. Основною особливістю ZARA є акцент на візуальному представленні продукції. Великі фотографії товарів та мінімальна кількість тексту створюють зручний інтерфейс для користувача.

Система навігації ZARA реалізована у вигляді категорій та фільтрів, що дозволяє швидко знаходити необхідні товари. Також реалізовано функціонал кошика покупок та авторизації користувачів. Проте одним із недоліків системи є велика кількість графічних елементів, що може впливати на швидкодію веб-застосунку при повільному інтернет-з'єднанні.

Онлайн-магазин Bershka орієнтований переважно на молодіжну аудиторію [24]. Даний веб-застосунок використовує сучасний дизайн, інтерактивні елементи та анімації. Інтерфейс сервісу є більш динамічним у порівнянні з іншими платформами.

Серед основних переваг Bershka можна виділити:

- сучасний адаптивний дизайн;
- інтерактивний інтерфейс;
- швидкий пошук товарів;
- зручне оформлення замовлення;
- інтеграцію із соціальними мережами.

Недоліком системи є велика кількість анімацій та інтерактивних елементів, що може збільшувати навантаження на клієнтську частину веб-застосунку.

Ще одним популярним веб-застосунком є ASOS [25]. Даний онлайн-магазин має широкий функціонал та підтримує велику кількість товарів різних брендів. Основною перевагою ASOS є розвинена система пошуку та фільтрації продукції. Користувачі можуть здійснювати пошук товарів за категоріями, брендом, ціною, кольором та іншими характеристиками.

ASOS також підтримує систему рекомендацій товарів та персоналізацію користувацького досвіду. Це дозволяє підвищити зацікавленість користувачів та покращити взаємодію із системою. Проте через велику кількість функціональних можливостей інтерфейс ASOS є більш складним у порівнянні з іншими веб-застосунками.

### **1.3.1 Онлайн-магазин ZARA**

ZARA є одним із найбільш популярних онлайн-магазинів одягу у світі та належить компанії Inditex Group [23]. Веб-застосунок даного бренду орієнтований на продаж сучасного одягу, взуття та аксесуарів для різних категорій користувачів. Онлайн-магазин активно використовується молодіжною аудиторією завдяки сучасному дизайну, швидкодії та зручному інтерфейсу користувача.

Основною особливістю онлайн-магазину ZARA є мінімалістичний стиль оформлення. Інтерфейс системи побудований таким чином, щоб основна увага користувача концентрувалася на візуальному представленні товарів. На головній

сторінці розміщуються великі фотографії продукції, рекламні банери та нові колекції одягу.



Рис.1.1 Екрані Форми Магазину ZARA



Рис.1.2 Головна сторінка онлайн-магазину ZARA

Навігація веб-застосунку є досить простою та зрозумілою.

Каталог продукції поділено на окремі категорії:

- чоловічий одяг;
- жіночий одяг;
- дитячий одяг;
- взуття;
- аксесуари.

Користувач може швидко перейти до потрібного розділу та скористатися системою фільтрації товарів. Фільтрація реалізована за такими параметрами:

- розмір;
- колір;
- ціна;
- тип товару;
- колекція.

Важливою перевагою онлайн-магазину ZARA є адаптивний дизайн. Веб-застосунок коректно працює як на персональних комп'ютерах, так і на мобільних пристроях. Інтерфейс автоматично адаптується до розміру екрана користувача, що забезпечує комфортне використання системи [18].

Система авторизації користувачів дозволяє створювати персональні акаунти, переглядати історію замовлень та зберігати обрані товари. Для оформлення покупки користувач може додати товар до кошика та перейти до оформлення замовлення через відповідний інтерфейс.

Особливу увагу у веб-застосунку приділено візуальному оформленню продукції. Для кожного товару доступно декілька фотографій, інформація про розміри, опис та ціна. Такий підхід дозволяє покращити взаємодію користувача із системою та спрощує процес вибору товару.

Ще однією перевагою ZARA є швидкодія системи. Сторінки веб-застосунку завантажуються досить швидко навіть при великій кількості графічного контенту. Для реалізації сучасного інтерфейсу використовуються інтерактивні елементи та анімації.

Проте система має і певні недоліки. Значна кількість графічних елементів може збільшувати навантаження на інтернет-з'єднання користувача. Крім того, мінімалістичний стиль інтерфейсу іноді ускладнює пошук окремих функцій для нових користувачів.

Основні переваги онлайн-магазину ZARA:

- сучасний мінімалістичний дизайн;
- адаптивний інтерфейс;

- швидка навігація;
- якісне візуальне представлення товарів;
- система фільтрації продукції;
- швидкодія веб-застосунку.

Основні недоліки:

- велика кількість графічного контенту;
- складність навігації для нових користувачів;
- мінімальна кількість текстових підказок в інтерфейсі.

Онлайн-магазин ZARA є прикладом сучасного веб-застосунку електронної комерції, який поєднує адаптивний дизайн, швидкодію та мінімалістичний інтерфейс користувача. Частина рішень, використаних у даній системі, може бути застосована під час розробки веб-застосунку «YNG-YOUNGSHOP».

### **1.3.2 Онлайн-магазин ASOS**

ASOS є одним із найбільших міжнародних онлайн-магазинів одягу та аксесуарів, який орієнтований переважно на молодіжну аудиторію [25]. Веб-застосунок підтримує продаж великої кількості брендів та забезпечує користувачам широкий функціонал для пошуку, перегляду та оформлення замовлень. На відміну від багатьох інших онлайн-магазинів, ASOS поєднує у собі функції маркетплейсу та класичного інтернет-магазину одягу.

Головною особливістю ASOS є велика кількість товарів та розвинена система персоналізації користувацького досвіду. Веб-застосунок дозволяє користувачам переглядати продукцію різних брендів, створювати списки обраних товарів та отримувати рекомендації відповідно до власних інтересів. Такий підхід дозволяє значно покращити взаємодію користувача із системою та спрощує процес пошуку необхідної продукції.

Інтерфейс онлайн-магазину ASOS має сучасний стиль оформлення та підтримує адаптивний дизайн. Веб-застосунок коректно працює на персональних комп'ютерах, планшетах та смартфонах.

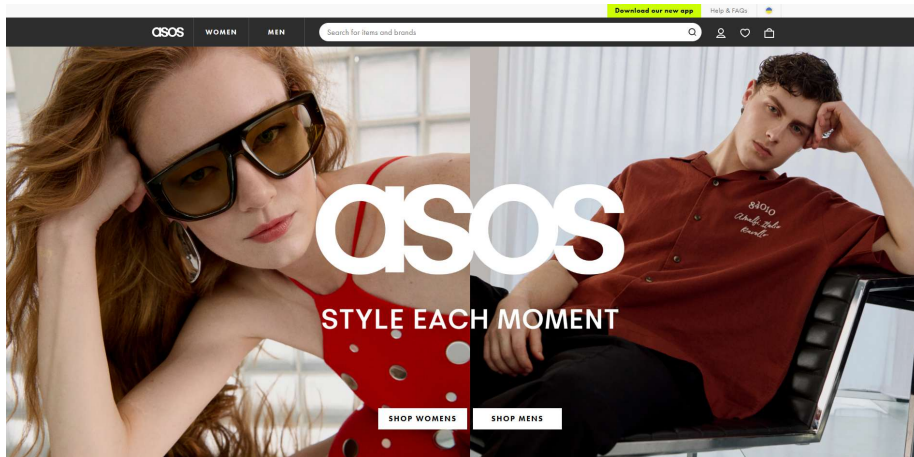


Рис.1.3 Екрана форма магазину ASO

На головній сторінці онлайн-магазину розміщуються популярні товари, нові колекції, сезонні пропозиції та персоналізовані рекомендації. Для привернення уваги користувачів використовуються великі банери, рекламні блоки та сучасні графічні елементи. У порівнянні з іншими онлайн-магазинами ASOS містить більше інформаційного контенту та маркетингових елементів.

Система пошуку товарів у веб-застосунку реалізована на високому рівні. Користувач може здійснювати пошук продукції за назвою, категорією, брендом, стилем або кольором. Крім того, система підтримує розширену фільтрацію товарів за різними параметрами, що значно спрощує роботу з великим каталогом продукції.

Важливою особливістю ASOS є система рекомендацій товарів. Веб-застосунок аналізує дії користувача та формує персоналізовані пропозиції на основі переглянутих товарів та попередніх покупок. Такий підхід активно використовується у сучасній електронній комерції для підвищення зацікавленості користувачів та збільшення кількості продажів [20].

Система авторизації користувачів дозволяє створювати персональні акаунти, переглядати історію покупок, формувати список обраних товарів та отримувати персональні рекомендації. Для забезпечення безпеки даних використовуються сучасні механізми авторизації та захисту користувацької інформації.

Онлайн-магазин також підтримує функціонал кошика покупок та оформлення замовлень через зручний інтерфейс. Користувач може змінювати кількість товарів, переглядати загальну вартість покупки та обирати спосіб доставки продукції.

Попри значну кількість функціональних можливостей, веб-застосунок ASOS має і певні недоліки. Велика кількість інформації та графічних елементів може ускладнювати навігацію для нових користувачів. Крім того, через складність структури сторінок система може працювати повільніше на пристроях із низькою продуктивністю або при слабкому інтернет-з'єднанні.

Основною перевагою ASOS є поєднання широкого функціоналу, сучасного дизайну та персоналізації користувацького досвіду. Саме тому даний веб-застосунок є одним із найбільш популярних сервісів електронної комерції серед молодіжної аудиторії.

Під час розробки веб-застосунку «YNG-YOUNGSHOP» доцільно використати окремі рішення, реалізовані в ASOS, зокрема сучасну систему фільтрації товарів, персоналізований підхід до користувачів та структуровану організацію каталогу продукції.

Онлайн-магазин ASOS є прикладом сучасного масштабованого веб-застосунку електронної комерції, який поєднує широкий функціонал, адаптивний інтерфейс та сучасні механізми взаємодії з користувачами

### **1.3.3 Онлайн-магазин Bershka**

Bershka є сучасним онлайн-магазином молодіжного одягу, який орієнтований переважно на підліткову аудиторію та прихильників streetwear стилю [24]. Веб-застосунок поєднує сучасний дизайн, інтерактивний інтерфейс та функціональні можливості електронної комерції. Основною метою системи є забезпечення швидкого та зручного процесу онлайн-покупок для користувачів.

Інтерфейс онлайн-магазину має яскравий сучасний стиль оформлення та активно використовує графічні елементи. На головній сторінці веб-застосунку

розміщуються рекламні банери, популярні колекції, нові надходження продукції та акційні пропозиції. Значна увага приділяється візуальному представленню товарів, оскільки саме зовнішній вигляд продукції є одним із головних факторів для молодіжної аудиторії.



Рис.1.4 Головна сторінка онлайн-магазину Bershka

Однією з важливих особливостей веб-застосунку є адаптивний дизайн. Система підтримує коректну роботу на різних типах пристроїв та автоматично адаптує інтерфейс до розміру екрана користувача. Це забезпечує комфортне використання веб-застосунку на смартфонах, планшетах та персональних комп'ютерах.

Для організації каталогу продукції використовується поділ товарів на окремі категорії. Користувач може перейти до розділів із чоловічим або жіночим одягом, аксесуарами, взуттям та новими колекціями. Навігація між сторінками реалізована досить просто та не потребує додаткових дій для пошуку потрібної інформації.

У веб-застосунку реалізовано систему пошуку та фільтрації товарів. Користувач може здійснювати пошук продукції за назвою або використовувати фільтри для сортування товарів за ціною, кольором, розміром та типом продукції. Наявність такого функціоналу значно покращує взаємодію користувача із системою та дозволяє швидше знаходити необхідний товар.

Важливим елементом системи є інтерактивність інтерфейсу. Bershka активно використовує анімації, плавні переходи між сторінками та ефекти взаємодії з елементами інтерфейсу. Це створює більш сучасний вигляд веб-застосунку та покращує загальний користувацький досвід.

Система авторизації користувачів дозволяє створювати персональні акаунти та зберігати інформацію про покупки. Користувачі можуть додавати товари до кошика, змінювати кількість продукції та оформлювати замовлення через спеціальний інтерфейс системи.

Ще однією особливістю Bershka є інтеграція із соціальними мережами. Онлайн-магазин активно використовує сучасні цифрові платформи для просування продукції та залучення нових користувачів. Такий підхід є ефективним для молодіжної аудиторії, яка активно користується соціальними мережами у повсякденному житті.

До основних переваг веб-застосунку можна віднести сучасний дизайн, адаптивність інтерфейсу, інтерактивність системи та зручну навігацію. Проте велика кількість анімацій та графічних елементів може негативно впливати на продуктивність системи при низькій швидкості інтернет-з'єднання.

Онлайн-магазин Bershka є прикладом сучасного веб-застосунку електронної комерції для молодіжної аудиторії. Аналіз даної системи дозволяє визначити сучасні підходи до створення інтерфейсу користувача та організації функціоналу онлайн-магазину одягу.

#### **1.3.4 Порівняння функціональних можливостей онлайн-магазинів**

Після проведення аналізу онлайн-магазину ZARA, Bershka та ASOS було визначено основні особливості сучасних онлайн-магазинів одягу. Усі досліджені системи мають схожий базовий функціонал, проте відрізняються дизайном інтерфейсу, рівнем інтерактивності та підходами до організації взаємодії з користувачем.

Онлайн-магазин ZARA орієнтований на мінімалістичний стиль оформлення та акцентує увагу на візуальному представленні товарів. Bershka використовує більш динамічний та молодіжний інтерфейс із великою кількістю інтерактивних елементів. У свою чергу ASOS має найбільш широкий функціонал, розвинену систему персоналізації та складну структуру каталогу товарів.

Для більш наочного порівняння основних характеристик онлайн-магазинів було сформовано таблицю 1.1.

Таблиця 1.1.

Порівняння функціональних можливостей онлайн-магазинів

| Характеристика            | ZARA     | Bershka | ASOS    |
|---------------------------|----------|---------|---------|
| Адаптивний дизайн         | +        | +       | +       |
| Авторизація користувачів  | +        | +       | +       |
| Кошик покупок             | +        | +       | +       |
| Пошук товарів             | +        | +       | +       |
| Фільтрація продукції      | +        | +       | +       |
| Інтерактивний інтерфейс   | Середній | Високий | Високий |
| Мінімалістичний дизайн    | +        | —       | —       |
| Персоналізація контенту   | —        | —       | +       |
| Інтеграція із соцмережами | Частково | +       | +       |
| Швидкодія системи         | Висока   | Середня | Середня |
| Простота навігації        | Висока   | Середня | Середня |

Проведений аналіз онлайн-магазинів ZARA, Bershka та ASOS показав, що сучасні веб-застосунки електронної комерції поєднують адаптивний дизайн, інтерактивний інтерфейс та широкий функціонал для взаємодії з користувачем. Кожна із досліджених систем має власні особливості організації інтерфейсу та структури каталогу продукції, проте всі вони забезпечують зручний процес пошуку товарів, оформлення замовлень та роботи з користувацьким акаунтом.

У результаті аналізу було визначено основні підходи, які доцільно використати під час розробки онлайн-магазину «YNG-YOUNGSHOP».

Зокрема, важливими елементами є адаптивний дизайн інтерфейсу, швидка навігація між сторінками, система фільтрації товарів, кошик покупок та сучасне візуальне оформлення продукції. Також важливу роль відіграє інтерактивність веб-застосунку та оптимізація роботи системи для мобільних пристроїв.

#### **1.4 Формування вимог до веб-застосунку**

Формування вимог є одним із найважливіших етапів розробки програмного забезпечення, оскільки саме на основі визначених вимог здійснюється подальше проєктування та реалізація системи [1]. Для створення веб-застосунку «YNG-YOUNGSHOP» було проведено аналіз сучасних онлайн-магазинів та визначено основні функціональні й нефункціональні вимоги до системи.

Основною метою розроблюваного веб-застосунку є створення сучасного онлайн-магазину одягу для підлітків із зручним інтерфейсом користувача, адаптивним дизайном та необхідним функціоналом електронної комерції.

Функціональні вимоги визначають перелік можливостей, які повинна підтримувати система для забезпечення взаємодії користувача із веб-застосунком.

До основних функціональних вимог системи належать:

- Управління каталогом товарів;
- Управління даними щодо товарів;
- Пошук та фільтрація товарів;
- Забезпечення роботи кошика товарів;
- Управління даними щодо замовлень;

Однією з головних функцій веб-застосунку є система каталогу товарів. Користувач повинен мати можливість переглядати список продукції, переходити між категоріями та отримувати інформацію про конкретний товар. Для кожного товару повинні відображатися фотографії, назва, опис, ціна та доступні розміри.

Система пошуку повинна забезпечувати швидке знаходження товарів за назвою або ключовими словами. Для покращення взаємодії користувача із системою також реалізовується механізм фільтрації продукції за категоріями та іншими параметрами.

Важливою складовою веб-застосунку є система авторизації користувачів. Реєстрація та вхід до системи дозволяють створювати персональні акаунти та зберігати інформацію про замовлення користувача. Для забезпечення безпеки даних повинні використовуватися механізми шифрування паролів та авторизації користувачів [11].

Ще однією важливою функцією є кошик покупок. Користувач повинен мати можливість додавати товари до кошика, змінювати їх кількість або видаляти продукцію перед оформленням замовлення. Система повинна автоматично розраховувати загальну вартість покупки.

Нефункціональні вимоги визначають характеристики якості програмного забезпечення та особливості роботи системи [2].

До основних нефункціональних вимог належать:

- Час завантаження сторінок веб-застосунку не повинен перевищувати 3 секунд;
- Система повинна коректно працювати у браузерях Google Chrome, Safari та Mozilla Firefox;
- Онлайн-магазин повинен адаптуватися під екрани смартфонів, планшетів та персональних комп'ютерів;
- Паролі користувачів повинні зберігатися у зашифрованому вигляді;

Веб-застосунок повинен підтримувати адаптивний дизайн та коректно працювати на різних типах пристроїв. Інтерфейс користувача має бути простим, сучасним та зрозумілим для молодіжної аудиторії. Для створення сучасного дизайну доцільно використовувати мінімалістичний стиль оформлення та інтерактивні елементи інтерфейсу [18].

Важливою вимогою є швидкодія системи. Час завантаження основних сторінок не повинен перевищувати декількох секунд навіть при великій кількості графічного контенту. Для цього необхідно оптимізувати клієнтську та серверну частини веб-застосунку.

Система також повинна забезпечувати безпечне зберігання інформації про користувачів та замовлення. Для цього необхідно використовувати сучасні механізми авторизації та захисту даних. Для більш наочного представлення вимог до веб-застосунку було сформовано таблицю 1.2

Таблиця 1.2

| Тип вимоги      | Опис                                                                                             |
|-----------------|--------------------------------------------------------------------------------------------------|
| Функціональна   | Управління каталогом товарів                                                                     |
| Функціональна   | Управління даними щодо товарів                                                                   |
| Функціональна   | Пошук та фільтрація товарів                                                                      |
| Функціональна   | Забезпечення роботи кошика товарів                                                               |
| Функціональна   | Управління даними щодо замовлень                                                                 |
| Нефункціональна | Адаптивний дизайн                                                                                |
| Нефункціональна | Паролі користувачів повинні зберігатися у зашифрованому вигляді                                  |
| Нефункціональна | Онлайн-магазин повинен адаптуватися під екрани смартфонів, планшетів та персональних комп'ютерів |
| Нефункціональна | Система повинна коректно працювати у браузерях Google Chrome, Safari та Mozilla Firefox          |
| Нефункціональна | Час завантаження сторінок веб-застосунку не повинен перевищувати 3 секунд;                       |

## 2 ВИБІР ТЕХНОЛОГІЙ ТА ЗАСОБІВ РОЗРОБКИ

Для розробки сучасного веб-застосунку необхідно правильно обрати технології та програмні засоби, які забезпечать стабільну роботу системи, швидкодію та зручність подальшої підтримки програмного коду. Під час створення онлайн-магазину одягу «YNG-YOUNGSHOP» було використано сучасний стек веб-технологій, який дозволяє реалізувати адаптивний та функціональний веб-застосунок.

Для створення клієнтської частини системи було обрано React [4]. Дана JavaScript-бібліотека є одним із найбільш популярних інструментів для розробки сучасних користувацьких інтерфейсів. React використовує компонентний підхід, який дозволяє розділяти інтерфейс на окремі частини та повторно використовувати їх у різних сторінках веб-застосунку.

Важливою перевагою React є швидке оновлення інформації на сторінці без її повного перезавантаження. Це позитивно впливає на продуктивність системи та покращує взаємодію користувача із веб-застосунком. Крім того, React підтримує SPA-архітектуру, що дозволяє створювати динамічні веб-застосунки [10].

Для стилізації інтерфейсу використовувався Tailwind CSS [8]. Даний CSS-фреймворк дозволяє швидко створювати сучасний адаптивний дизайн за допомогою готових utility-класів. Використання Tailwind CSS значно спрощує процес стилізації сторінок та дозволяє підтримувати єдиний стиль інтерфейсу у всьому веб-застосунку.

Серверна частина системи була реалізована за допомогою Node.js [5]. Дане середовище дозволяє виконувати JavaScript-код на сервері та забезпечує швидку взаємодію між клієнтською і серверною частинами веб-застосунку. Node.js підтримує асинхронну обробку запитів, що дозволяє ефективно працювати з великою кількістю користувачів.

Для створення серверної логіки та REST API було використано Express.js [6]. Express є популярним веб-фреймворком для Node.js та забезпечує зручний механізм обробки HTTP-запитів, маршрутизації та роботи із серверною частиною системи.

Для зберігання інформації про товари, користувачів та замовлення було використано MongoDB [7]. Дана система управління базами даних належить до NoSQL баз даних та забезпечує гнучку структуру зберігання інформації. MongoDB добре підходить для веб-застосунків електронної комерції завдяки високій продуктивності та масштабованості.

Також використовувалася система контролю версій Git [16]. Використання Git дозволяє відстежувати зміни у програмному коді та забезпечує зручну роботу над проєктом. Для зберігання репозиторію використовувалася платформа GitHub [17].

Середовищем розробки було обрано Visual Studio Code. Дане середовище підтримує роботу з JavaScript, React та Node.js, а також містить велику кількість інструментів для веб-розробки.

Для створення макетів інтерфейсу використовувався сервіс Figma [15]. Даний інструмент дозволив створити структуру сторінок веб-застосунку та продумати дизайн інтерфейсу перед початком програмної реалізації системи.

Використання сучасних веб-технологій дозволяє реалізувати швидкий, адаптивний та функціональний веб-застосунок електронної комерції, який відповідає сучасним вимогам до онлайн-магазинів одягу.

## **2.1. Середовище розробки**

У процесі створення веб-застосунку «YNG-YOUNGSHOP» було задіяно низку сучасних програмних інструментів, які забезпечили зручність реалізації функціоналу, оптимізацію процесу написання коду та підтримку актуальних технологій веб-розробки. Коректний вибір середовища розробки має важливе значення для створення якісного програмного продукту, оскільки від цього

залежить ефективність роботи розробника, швидкість внесення змін та можливість інтеграції різних програмних компонентів [1].

Основним середовищем для написання програмного коду став Visual Studio Code. Даний редактор вирізняється високою швидкістю роботи, підтримкою сучасних мов програмування та широкими можливостями налаштування. Саме тому він активно використовується під час розроблення веб-застосунків різного рівня складності.

Середовище Visual Studio Code використовувалося для роботи з технологіями JavaScript, React та Node.js. Додатково застосовувалися спеціальні розширення, які дозволяли автоматизувати форматування коду, здійснювати перевірку синтаксису та працювати із системою контролю версій Git. Це значно спростило процес розробки та підвищило якість програмного коду.

Проектування зовнішнього вигляду веб-застосунку виконувалося за допомогою сервісу Figma [15]. Використання цього інструмента дало можливість створити макети сторінок, визначити структуру інтерфейсу та продумати розташування основних елементів системи ще до початку етапу програмування.

Важливою перевагою Figma є підтримка спільної роботи над дизайном і можливість швидкого внесення змін до макетів. Окрім цього, сервіс підтримує експорт графічних елементів та взаємодію з іншими інструментами, що використовуються у сфері веб-розробки.

На етапі тестування функціонування веб-застосунку також застосовувалися стандартні браузерні інструменти розробника. Вони використовувалися для перевірки адаптивності інтерфейсу, аналізу мережових запитів та контролю продуктивності веб-застосунку під час роботи.

### **2.1.1 Visual Studio Code**

Для програмної реалізації веб-застосунку «YNG-YOUNGSHOP» було використано середовище розробки Visual Studio Code. Даний редактор програмного коду є одним із найбільш популярних інструментів для веб-

розробки завдяки своїй швидкодії, зручному інтерфейсу та підтримці великої кількості мов програмування [1].

Visual Studio Code підтримує роботу з JavaScript, React, Node.js та іншими технологіями, які використовувалися під час створення онлайн-магазину. Середовище дозволяє ефективно працювати як із клієнтською, так і серверною частинами веб-застосунку.

Однією з головних переваг Visual Studio Code є підтримка великої кількості розширень. Під час розробки веб-застосунку використовувалися розширення для:

- автоматичного форматування коду;
- перевірки синтаксису;
- роботи з React;
- підтримки Tailwind CSS;
- інтеграції із Git;
- підсвічування структури коду.

Використання даних інструментів значно спрощує процес розробки та дозволяє зменшити кількість помилок у програмному коді.

Visual Studio Code також підтримує вбудований термінал, що дозволяє запускати серверну частину веб-застосунку, встановлювати залежності та працювати із системою контролю версій без необхідності використання сторонніх програм.

Важливою перевагою середовища є інтеграція із системою Git [16]. Це дозволяє відстежувати зміни у програмному коді, створювати резервні копії проєкту та зручно працювати з репозиторієм GitHub [17].

Під час розробки веб-застосунку Visual Studio Code використовувався для:

- створення структури проєкту;
- написання клієнтської частини на React;
- реалізації серверної логіки Node.js;
- роботи з API;
- налаштування маршрутизації;

– стилізації інтерфейсу за допомогою Tailwind CSS.

Середовище також забезпечує швидку навігацію між файлами проєкту та підтримує автоматичне оновлення коду під час розробки. Це позитивно впливає на швидкість створення веб-застосунку та спрощує процес тестування функціоналу системи.

Використання Visual Studio Code дозволило забезпечити зручний та ефективний процес розробки веб-застосунку «YNG-YOUNGSHOP», а також спростило роботу з сучасними веб-технологіями.

### **2.1.2 Figma**

Для створення дизайну веб-застосунку «YNG-YOUNGSHOP» використовувався сервіс Figma [15]. Даний інструмент є сучасною платформою для проєктування інтерфейсів та створення макетів веб-застосунків. Використання Figma дозволяє продумати зовнішній вигляд системи ще до початку програмної реалізації, що значно спрощує процес подальшої розробки.

Під час створення онлайн-магазину у Figma були спроектовані основні сторінки веб-застосунку, зокрема головна сторінка, каталог товарів, сторінка авторизації, кошик покупок та сторінка оформлення замовлення. Це дозволило заздалегідь визначити структуру інтерфейсу та розташування основних елементів системи.

Особлива увага приділялася дизайну користувацького інтерфейсу. Оскільки веб-застосунок орієнтований на молодіжну аудиторію, при створенні макетів використовувався сучасний мінімалістичний стиль оформлення. Для інтерфейсу було обрано темну кольорову палітру, великі фотографії продукції та просту структуру навігації.

Використання Figma дозволило створити адаптивний дизайн сторінок та продумати відображення інтерфейсу на різних типах пристроїв. Це є важливим етапом розробки сучасного веб-застосунку, оскільки значна частина користувачів здійснює покупки саме через смартфони та планшети.

Однією з переваг Figma є можливість швидкого редагування макетів та внесення змін у структуру інтерфейсу. Крім того, сервіс підтримує експорт графічних елементів, що спрощує процес інтеграції дизайну у програмний код веб-застосунку.

## 2.2 Технології клієнтської частини

Frontend-частина веб-застосунку є одним із ключових компонентів системи, оскільки саме вона забезпечує взаємодію користувача з онлайн-магазином. Через клієнтську частину користувач переглядає товари, переходить між сторінками, додає продукцію до кошика та оформлює замовлення. Тому під час розробки особлива увага приділялася швидкодії інтерфейсу, сучасному дизайну та адаптивності веб-застосунку.

Основною технологією для створення клієнтської частини було обрано React [4]. Дана бібліотека дозволяє створювати динамічні інтерфейси за допомогою компонентного підходу. У процесі розробки сторінки веб-застосунку були розділені на окремі компоненти, що значно спростило структуру проєкту та зробило програмний код більш зрозумілим і зручним для підтримки.

Однією з переваг React є можливість оновлення даних без повного перезавантаження сторінки. Це дозволяє забезпечити більш швидку взаємодію користувача із системою та покращує загальну продуктивність веб-застосунку [10]. Крім того, React добре підходить для створення SPA-застосунків, де більшість дій виконується динамічно.

Для навігації між сторінками використовувалася бібліотека React Router. За її допомогою було реалізовано маршрутизацію між основними розділами онлайн-магазину, зокрема головною сторінкою, каталогом товарів, сторінкою авторизації, кошиком покупок та оформленням замовлення.

Стилізація інтерфейсу виконувалася за допомогою Tailwind CSS [8]. Використання даного CSS-фреймворку дозволило швидко створити сучасний адаптивний дизайн та підтримувати єдиний стиль оформлення у всіх

компонентах системи. Під час розробки інтерфейсу використовувалися темна кольорова палітра, мінімалістичний стиль та великі графічні елементи, що відповідають сучасним тенденціям веб-дизайну.

Важливою частиною frontend-розробки була реалізація взаємодії із серверною частиною веб-застосунку. Для виконання HTTP-запитів використовувалася бібліотека Axios. За допомогою API клієнтська частина отримує інформацію про товари, користувачів та замовлення, а також передає дані до серверної частини системи.

У процесі розробки клієнтської частини було реалізовано систему авторизації користувачів, пошук продукції, фільтрацію товарів, кошик покупок та адаптивне меню навігації.

### **2.2.1 React**

Серед сучасних технологій frontend-розробки React займає одне з провідних місць завдяки своїй продуктивності та гнучкості [4]. Саме тому дана бібліотека була обрана для створення клієнтської частини «YNG-YOUNGSHOP».

Головна ідея React полягає у побудові інтерфейсу з окремих незалежних компонентів. Такий підхід дозволяє значно спростити структуру веб-застосунку та зробити програмний код більш організованим. Кожен компонент відповідає за певну частину інтерфейсу, наприклад навігаційне меню, картку товару або форму авторизації.

У процесі розробки онлайн-магазину React використовувався для створення всіх основних сторінок системи. За допомогою компонентного підходу було реалізовано:

- головну сторінку;
- каталог продукції;
- систему пошуку товарів;
- кошик покупок;
- сторінку авторизації;
- оформлення замовлення.

Важливою перевагою React є використання Virtual DOM [10]. На відміну від звичайного оновлення HTML-сторінки, React змінює лише ті елементи інтерфейсу, які були оновлені. Це значно покращує швидкодію веб-застосунку та забезпечує плавну взаємодію користувача із системою.

Під час роботи над проектом активно використовувалися React Hooks. Наприклад, `useState` застосовувався для роботи зі станом компонентів, а `useEffect` - для виконання динамічних дій під час завантаження сторінок та отримання інформації із сервера.

Окрему роль React відіграв у реалізації адаптивного інтерфейсу. Компонентна структура дозволила швидко змінювати вигляд сторінок для різних типів пристроїв та забезпечити коректне відображення веб-застосунку на смартфонах і планшетах.

Для організації навігації між сторінками використовувався React Router. Це дозволило створити SPA-застосунок, де переходи між сторінками виконуються без повного перезавантаження браузера. У результаті інтерфейс працює значно швидше та виглядає більш сучасно.

Було також реалізовано взаємодію React із серверною частиною через API-запити. Для цього використовувалася бібліотека `Axios`, за допомогою якої клієнтська частина отримує інформацію про товари, користувачів та замовлення.

### **2.2.2 Tailwind CSS**

Створення сучасного інтерфейсу веб-застосунку потребує використання інструментів, які дозволяють швидко реалізовувати адаптивний дизайн та підтримувати єдиний стиль оформлення системи. Під час розробки онлайн-магазину «YNG-YOUNGSHOP» для стилізації сторінок було використано Tailwind CSS [8].

Tailwind CSS є utility-first CSS-фреймворком, який надає набір готових класів для оформлення елементів інтерфейсу. На відміну від класичного підходу до написання стилів, де створюються окремі CSS-файли, Tailwind дозволяє

застосовувати стилі безпосередньо у структурі компонентів. Це значно прискорює процес створення інтерфейсу та робить код більш структурованим.

У процесі розробки веб-застосунку Tailwind CSS використовувався для побудови адаптивної структури сторінок, налаштування кольорової палітри, стилізації кнопок, форм авторизації та карток товарів. Завдяки готовим utility-класам вдалося швидко реалізувати сучасний мінімалістичний дизайн онлайн-магазину.

Однією з головних переваг Tailwind CSS є підтримка адаптивної верстки. Фреймворк дозволяє легко змінювати структуру сторінок залежно від ширини екрана пристрою. Це дозволило забезпечити коректне відображення веб-застосунку на смартфонах, планшетах та персональних комп'ютерах.

Під час створення інтерфейсу використовувалися flex- та grid-моделі верстки, які значно спрощують розташування елементів на сторінці. Для інтерактивних компонентів були реалізовані ефекти наведення, плавні переходи та анімації, що зробило інтерфейс більш сучасним та динамічним.

Важливою перевагою Tailwind CSS є можливість швидкого редагування дизайну. Зміна стилів не потребує пошуку окремих CSS-файлів, оскільки більшість параметрів задається безпосередньо у компонентах. Це значно спрощує підтримку інтерфейсу та пришвидшує процес розробки нових сторінок.

Під час реалізації веб-застосунку було використано темну кольорову палітру та сучасний стиль оформлення, орієнтований на молодіжну аудиторію. Tailwind CSS дозволив швидко реалізувати дану концепцію та забезпечити єдиний візуальний стиль усієї системи.

Таким чином, використання Tailwind CSS позитивно вплинуло на швидкість розробки, адаптивність інтерфейсу та зовнішній вигляд веб-застосунку «YNG-YOUNGSHOP».

## 2.3 Технології серверної частини

Серверна частина є основою роботи веб-застосунку, оскільки саме вона відповідає за обробку інформації, виконання бізнес-логіки та взаємодію з базою даних. У системі «YNG-YOUNGSHOP» backend забезпечує роботу авторизації, обробку замовлень, керування товарами та передачу даних між користувачем і базою даних.

Для реалізації серверної частини було обрано Node.js [5]. Дана технологія дозволяє виконувати JavaScript-код поза браузером та широко використовується для створення сучасних веб-сервісів. Однією з причин вибору Node.js стала його висока продуктивність та підтримка асинхронної обробки запитів.

Асинхронна модель роботи дозволяє системі ефективно працювати навіть при великій кількості одночасних підключень. Це особливо важливо для онлайн-магазинів, де користувачі постійно взаємодіють із товарами, кошиком та сторінками оформлення замовлення.

Для організації серверної логіки використовувався Express.js [6]. Даний фреймворк значно спрощує створення API та маршрутизацію HTTP-запитів. За допомогою Express.js у проєкті були створені окремі маршрути для роботи з товарами, користувачами, кошиком та замовленнями.

Структура backend-частини була поділена на контролери, маршрути та моделі даних. Такий підхід дозволив зробити код більш структурованим і зручним для підтримки. Кожен модуль відповідає лише за свій функціонал, що спрощує подальше розширення системи.

У процесі розробки також була реалізована система авторизації користувачів. Для захисту акаунтів використовувалося шифрування паролів та JWT-токени [11]. Це дозволило організувати безпечний доступ до функціоналу веб-застосунку та зберегти конфіденційність даних користувачів.

Окрему увагу приділено роботі з API. Клієнтська частина надсилає HTTP-запити до сервера, після чого backend обробляє їх та повертає необхідну

інформацію у форматі JSON. Такий підхід забезпечує швидкий обмін даними між frontend- і backend-частинами системи.

У результаті використання Node.js та Express.js вдалося створити гнучку серверну архітектуру, яка підтримує стабільну роботу онлайн-магазину та може бути розширена у майбутньому новими функціональними можливостями.

### **2.3.1 Node.js**

Сучасні веб-застосунки потребують швидкої обробки великої кількості запитів та стабільної роботи серверної частини. Для реалізації таких можливостей у проєкті «YNG-YOUNGSHOP» було використано Node.js [5], яке є одним із найпоширеніших середовищ для backend-розробки.

Головною причиною вибору Node.js стала його продуктивність та підтримка асинхронного виконання операцій. На практиці це дозволяє серверу працювати з багатьма користувачами одночасно без значних затримок. Для онлайн-магазину це особливо важливо, оскільки система постійно обробляє запити, пов'язані із переглядом товарів, авторизацією та оформленням замовлень.

Node.js побудований на JavaScript-рушії V8, який забезпечує швидке виконання програмного коду. Використання JavaScript як на стороні клієнта, так і на сервері дозволило зробити структуру проєкту більш цілісною та спростило взаємодію між frontend- і backend-частинами системи.

У межах розробленого веб-застосунку Node.js виконує роль центрального модуля обробки даних. Через сервер проходять запити користувачів, після чого система взаємодіє з базою даних та повертає необхідну інформацію клієнтській частині.

Для реалізації функціоналу онлайн-магазину за допомогою Node.js було організовано:

- роботу API;
- обробку інформації про товари;

- систему авторизації;
- керування замовленнями;
- взаємодію з MongoDB.

Окрему роль під час розробки відіграв npm - менеджер пакетів Node.js. Він дозволив швидко підключати додаткові бібліотеки та інструменти, необхідні для роботи серверної частини веб-застосунку.

Завдяки використанню Node.js серверна частина «YNG-YOUNGSHOP» отримала гнучку структуру, високу швидкість роботи та можливість подальшого розширення функціоналу системи.

### **2.3.2 Express.js**

Для організації серверної логіки у веб-застосунку «YNG-YOUNGSHOP» використовувався фреймворк Express.js [6]. Даний інструмент працює поверх Node.js та значно спрощує створення серверної частини веб-застосунків.

Express.js дозволяє швидко налаштовувати маршрути, обробляти HTTP-запити та створювати API для взаємодії клієнтської частини із сервером. Саме через API frontend отримує інформацію про товари, користувачів та замовлення.

У процесі розробки веб-застосунку були створені окремі маршрути для:

- роботи з товарами;
- авторизації користувачів;
- оформлення замовлень;
- роботи з кошиком;
- керування даними акаунта.

Однією з головних переваг Express.js є проста структура проєкту. Серверна логіка була розділена на окремі файли маршрутів, контролерів та моделей даних. Це дозволило зробити код більш зрозумілим та спростило підтримку системи.

Під час роботи із запитами використовувалися middleware-функції. Вони дозволяють перевіряти дані користувача, обробляти токени авторизації та контролювати доступ до окремих функцій системи.

Express.js також забезпечує зручну інтеграцію з MongoDB та іншими інструментами backend-розробки. Завдяки цьому вдалося реалізувати швидкий обмін інформацією між сервером та базою даних.

## **2.4. Система управління базами даних**

У процесі роботи онлайн-магазину формується значний обсяг інформації, яка повинна швидко оброблятися та надійно зберігатися. До таких даних належать відомості про користувачів, каталог продукції, історію покупок та оформлені замовлення. Для організації зберігання цих даних у проєкті «YNG-YOUNGSHOP» було використано MongoDB [7].

На відміну від класичних реляційних систем, MongoDB працює з документами, структура яких нагадує JSON-об'єкти. Це дозволяє більш зручно організувати роботу із JavaScript-технологіями, що використовуються у веб-застосунку.

Під час проєктування системи база даних була побудована за принципом окремих колекцій. Для кожного типу інформації створювалася власна структура:

- колекція товарів;
- колекція користувачів;
- колекція замовлень;
- колекція даних кошика.

Такий підхід спрощує обробку інформації та дозволяє серверній частині швидко отримувати необхідні дані.

У структурі товарів містяться характеристики продукції, включаючи назву, категорію, вартість, фотографії та опис. Для користувачів зберігаються дані авторизації та інформація про покупки. Окремо формується структура

замовлень, яка дозволяє відстежувати оформлені покупки та взаємодію клієнта із системою.

Робота з MongoDB у проєкті виконувалася через Mongoose - бібліотеку для Node.js, яка забезпечує створення моделей даних та підтримує зручну взаємодію із базою даних. За допомогою моделей реалізовано операції створення, пошуку, оновлення та видалення інформації.

Важливою особливістю MongoDB є можливість легко змінювати структуру документів під час розвитку системи. Це дозволяє без складних змін додавати нові поля або розширювати функціонал веб-застосунку.

Для онлайн-магазину також важливе значення має швидкість доступу до інформації. MongoDB забезпечує високу продуктивність під час роботи з великою кількістю товарів та одночасними запитами користувачів.

MongoDB використовувалася для збереження даних про товари, користувачів та замовлення. Такий підхід дозволив організувати гнучку структуру бази даних та спростив взаємодію серверної частини із інформацією.

Робота з MongoDB здійснювалася через бібліотеку Mongoose, яка забезпечує створення моделей даних та виконання операцій із базою даних. За допомогою моделей у системі реалізовано додавання, редагування та видалення інформації.

Однією з переваг MongoDB є висока швидкість обробки даних та можливість масштабування системи. Це робить дану базу даних ефективним рішенням для онлайн-магазинів та веб-застосунків електронної комерції.

## **2.5. Система контролю версій**

Під час розробки веб-застосунку «YNG-YOUNGSHOP» використовувалася система контролю версій Git [16]. Вона дозволяє відстежувати зміни у програмному коді, зберігати різні версії проєкту та спрощує процес розробки.

Git використовувався для створення резервних копій коду, повернення до попередніх версій та контролю змін у файлах проєкту. Це дозволило уникнути втрати даних під час розробки веб-застосунку.

Для зберігання репозиторію використовувалася платформа GitHub [17]. За допомогою GitHub забезпечувалося хмарне зберігання проєкту та зручна робота з програмним кодом.

Використання системи контролю версій позитивно вплинуло на організацію структури проєкту та спростило підтримку веб-застосунку «YNG-YOUNGSHOP».

### 2.5.1 Git

У процесі створення онлайн-магазину «YNG-YOUNGSHOP» важливо було контролювати всі зміни програмного коду та мати можливість швидко відновлювати попередні версії проєкту. Для цього використовувалася система Git [16].

Git дозволяє фіксувати окремі етапи розробки у вигляді комітів. Під час роботи над веб-застосунком це дало можливість поступово додавати новий функціонал без ризику пошкодження вже реалізованих компонентів системи.

Під час розробки Git застосовувався для:

- збереження історії змін;
- тестування нового функціоналу;
- резервного копіювання коду;
- роботи з різними версіями проєкту.

Однією з переваг Git є можливість створення окремих гілок розробки. Це дозволяє працювати над новими модулями незалежно від основної версії системи та безпечно інтегрувати зміни після тестування.

Також система значно спрощує пошук помилок у коді, оскільки будь-які зміни можна переглянути або скасувати. Для великих вебпроєктів це є важливою частиною організації процесу розробки.

Використання Git зробило процес створення веб-застосунку більш структурованим та дозволило контролювати всі етапи реалізації проєкту.

### **2.5.2 GitHub**

Для зберігання програмного коду та роботи з репозиторієм у проєкті «YNG-YOUNGSHOP» використовувалася платформа GitHub [17]. Даний сервіс є одним із найпоширеніших інструментів для хмарного зберігання програмних проєктів та підтримує інтеграцію із системою Git.

У процесі розробки GitHub використовувався як віддалений репозиторій, у якому зберігалися всі файли веб-застосунку. Це дозволило уникнути втрати програмного коду та забезпечити доступ до проєкту з різних пристроїв.

Однією з важливих можливостей GitHub є синхронізація локального проєкту з хмарним репозиторієм. Після внесення змін у код оновлення передавалися до GitHub за допомогою push-запитів.

Платформа також дозволяє:

- переглядати історію змін;
- працювати з різними версіями проєкту;
- зберігати резервні копії системи;
- контролювати структуру програмного коду.

У межах проєкту GitHub використовувався не лише для зберігання коду, а й для організації структури веб-застосунку та контролю етапів розробки.

Завдяки використанню GitHub процес роботи над «YNG-YOUNGSHOP» став більш зручним та організованим.

### 3. ПРОЄКТУВАННЯ СИСТЕМИ

Етап проєктування є одним із найважливіших під час створення програмного забезпечення, оскільки саме на цьому етапі формується структура майбутнього веб-застосунку, визначається взаємодія між компонентами системи та організовується логіка роботи онлайн-магазину.

Для «YNG-YOUNGSHOP» проєктування виконувалося з урахуванням сучасних принципів побудови клієнт-серверних систем. Основною метою було створення масштабованої архітектури, яка забезпечує стабільну роботу веб-застосунку та підтримує подальше розширення функціональних можливостей.

Під час проєктування були визначені основні модулі системи:

- клієнтська частина;
- серверна частина;
- база даних;
- API для обміну даними;
- система авторизації користувачів.

Клієнтська частина відповідає за відображення інтерфейсу та взаємодію користувача із веб-застосунком. Через frontend користувач може переглядати каталог товарів, виконувати пошук продукції, додавати товари до кошика та оформлювати замовлення.

Серверна частина забезпечує обробку запитів, взаємодію з базою даних та виконання основної логіки системи. Backend також контролює процес авторизації користувачів та обробку інформації про замовлення.

Для зберігання інформації використовується база даних MongoDB, у якій розміщуються дані про користувачів, товари та оформлені покупки.

Взаємодія між frontend- та backend-частинами системи реалізована через REST API. Передача інформації здійснюється у форматі JSON, що забезпечує швидкий обмін даними між компонентами веб-застосунку.

Для наочного відображення структури веб-застосунку та визначення взаємозв'язків між окремими компонентами системи було розроблено UML-діаграми й схеми взаємодії модулів. Їх використання дало змогу детальніше описати логіку функціонування застосунку та особливості організації його складових.

У межах розділу будуть розглянуті:

- архітектура веб-застосунку;
- діаграма варіантів використання;
- діаграма діяльності;
- структура бази даних;
- схема взаємодії компонентів системи.

### **3.1. Проєктування архітектури веб-застосунку**

Архітектура веб-застосунку визначає структуру системи, взаємодію між її компонентами та спосіб обробки даних. Для онлайн-магазину «YNG-YOUNGSHOP» було обрано клієнт-серверну архітектуру, яка є одним із найбільш поширених рішень для сучасних веб-застосунків електронної комерції [3].

У даній архітектурі система поділяється на три основні частини:

- frontend;
- backend;
- база даних.

Клієнтська частина реалізована за допомогою React та відповідає за інтерфейс користувача. Саме через frontend здійснюється взаємодія із каталогом товарів, кошиком покупок та сторінками авторизації.

Серверна частина створена на основі Node.js та Express.js. Вона обробляє запити користувачів, взаємодіє з базою даних та виконує логіку роботи системи.

Для зберігання інформації використовується MongoDB, де містяться дані про продукцію, акаунти користувачів та оформлені замовлення.

Обмін даними між frontend- та backend-частинами виконується через REST API. Клієнтська частина надсилає HTTP-запити до сервера, після чого backend обробляє інформацію та повертає відповідь у форматі JSON.

Під час проєктування архітектури особлива увага приділялася:

- швидкодії системи;
- адаптивності інтерфейсу;
- безпеці даних;
- масштабованості структури;
- зручності підтримки програмного коду.

Для кращого розуміння структури системи було створено схему архітектури веб-застосунку.

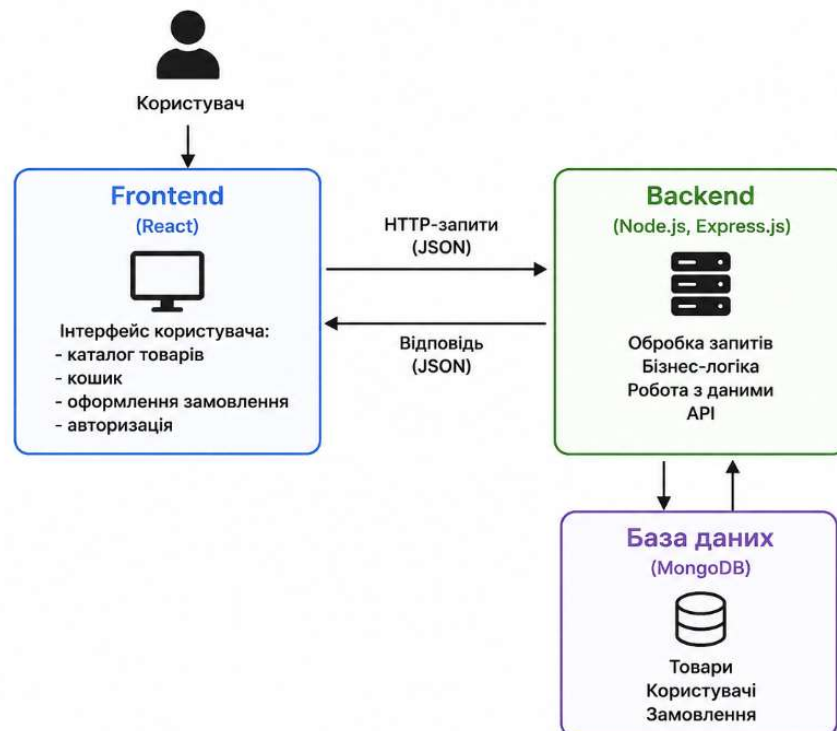


Рис.3.1 Архітектура «YNG-YOUNGSHOP»

На схемі відображено взаємодію між клієнтською частиною, сервером та базою даних. Така структура дозволяє розділити функціонал системи та забезпечує стабільну роботу «YNG-YOUNGSHOP» навіть при великій кількості користувачів.

### **3.1.1 Діаграма варіантів використання**

Одним із важливих етапів проектування веб-застосунку є побудова діаграми варіантів використання (Use Case Diagram). Даний тип UML-діаграм дозволяє визначити основні функціональні можливості системи та відобразити взаємодію користувачів із веб-застосунком [12].

Діаграма варіантів використання використовується для моделювання поведінки системи з точки зору кінцевого користувача. Вона допомагає визначити основні сценарії роботи онлайн-магазину та сформувати структуру взаємодії між окремими компонентами системи.

Для веб-застосунку «YNG-YOUNGSHOP» було визначено два основних актори:

- користувач;
- адміністратор.

Користувач є основним елементом взаємодії із системою. Через інтерфейс онлайн-магазину він може переглядати каталог продукції, виконувати пошук товарів, додавати товари до кошика та оформлювати замовлення.

Основні функції користувача у системі:

- перегляд товарів;
- пошук продукції;
- фільтрація каталогу;
- перегляд інформації про товар;
- реєстрація;
- авторизація;
- додавання товарів до кошика;
- оформлення замовлення.

Адміністратор виконує функції керування веб-застосунком та відповідає за підтримку каталогу продукції. Через адміністративну частину системи він може додавати нові товари, змінювати інформацію про продукцію та контролювати замовлення користувачів.

Функції адміністратора:

- додавання товарів;
- редагування продукції;
- видалення товарів;
- перегляд замовлень;
- керування каталогом.

Створення діаграми варіантів використання дозволило визначити основні функціональні межі веб-застосунку та спростило подальше проєктування серверної та клієнтської частин системи.



7

Рис.3.2 Діаграма варіантів використання «YNG-YOUNGSHOP»

На діаграмі варіантів використання можуть застосовуватися зв'язки include та extend. Зв'язок include використовується у випадках, коли один процес обов'язково викликає інший. Наприклад, оформлення замовлення потребує авторизації користувача.

Зв'язок extend застосовується для додаткових або необов'язкових функцій системи. Такий тип зв'язку використовується тоді, коли певна дія може виконуватися лише за окремих умов.

### **3.1.2 Діаграма станів та предметної галузі**

Для моделювання послідовності дій користувача у веб-застосунку «YNG-YOUNGSHOP» була створена діаграма станів та предметної галузі. Даний тип UML-діаграм використовується для відображення логіки виконання процесів та переходів між окремими етапами роботи системи [12].

Діаграма дозволяє наочно представити алгоритм взаємодії користувача із онлайн-магазином, починаючи від відкриття веб-застосунку та завершуючи оформленням замовлення.

У процесі роботи із системою користувач спочатку переходить на головну сторінку веб-застосунку, після чого може переглядати каталог продукції та виконувати пошук товарів. Після вибору необхідного товару користувач переходить на сторінку продукції та додає товар до кошика.

#### **Створення діаграми**

дозволило визначити основну логіку роботи веб-застосунку та послідовність виконання дій користувача під час взаємодії із системою.



Рис.3.3 Діаграма Станів

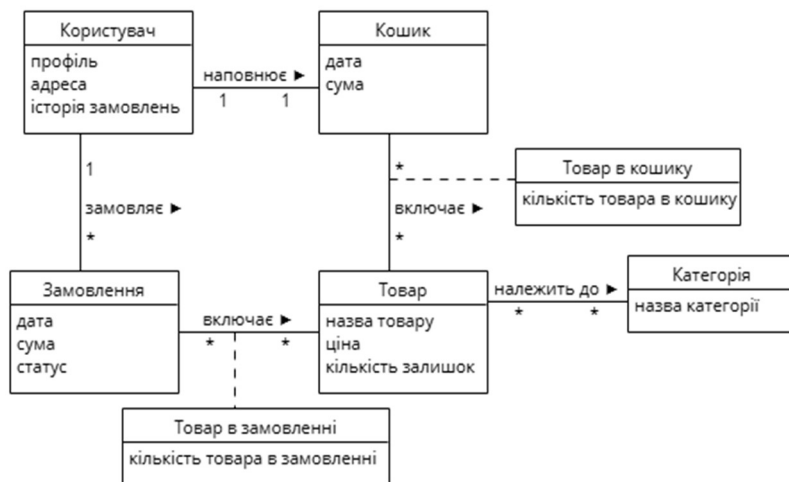


Рис.3.4 Діаграма предметної галузі

### 3.2. Проектування структури клієнтської частини

Клієнтська частина веб-застосунку «YNG-YOUNGSHOP» була спроектована з використанням компонентного підходу, що дозволяє розділити інтерфейс на окремі незалежні елементи. Такий підхід значно спрощує

підтримку програмного коду та дозволяє повторно використовувати компоненти у різних частинах системи.

Frontend-частина реалізована за допомогою бібліотеки React, яка забезпечує створення динамічного інтерфейсу та швидке оновлення інформації без перезавантаження сторінок [4]. Архітектура клієнтської частини побудована за принципом Single Page Application, що дозволяє забезпечити плавну навігацію між сторінками веб-застосунку.

У структурі клієнтської частини було виділено окремі компоненти, кожен із яких виконує власну функцію. Основними елементами системи є:

- Navbar - навігаційне меню;
- Hero - головний банер;
- ProductItem - картка товару;
- Collection - каталог продукції;
- Cart - кошик покупок;
- Login - сторінка авторизації;
- Footer - нижня частина сайту.

Для переходів між сторінками використовувалася бібліотека React Router. Вона дозволяє організувати маршрутизацію без повторного завантаження сторінки та забезпечує швидку роботу інтерфейсу.

Обмін даними між окремими компонентами системи реалізовано із використанням Context API, що забезпечило централізоване зберігання інформації про товари, вміст кошика та дані авторизованого користувача. Такий підхід спростив керування станом застосунку та оптимізував взаємодію між елементами інтерфейсу.

Особливу увагу було приділено забезпеченню адаптивності користувацького інтерфейсу. Усі сторінки веб-застосунку коректно відображаються на різних типах пристроїв, зокрема на смартфонах, планшетах і персональних комп'ютерах. Це дозволило забезпечити зручність користування системою незалежно від розміру екрана.

Для оформлення інтерфейсу застосовувався фреймворк Tailwind CSS [8]. Використання utility-класів дало змогу створити сучасний мінімалістичний дизайн, що відповідає стилістиці бренду «YNG-YOUNGSHOP» та забезпечує єдиний стиль усіх елементів системи.

Архітектура клієнтської частини була побудована з урахуванням можливості подальшого розширення функціоналу. Завдяки модульній структурі нові компоненти можуть інтегруватися до проєкту без суттєвих змін основної архітектури веб-застосунку, що спрощує процес його модернізації та підтримки.

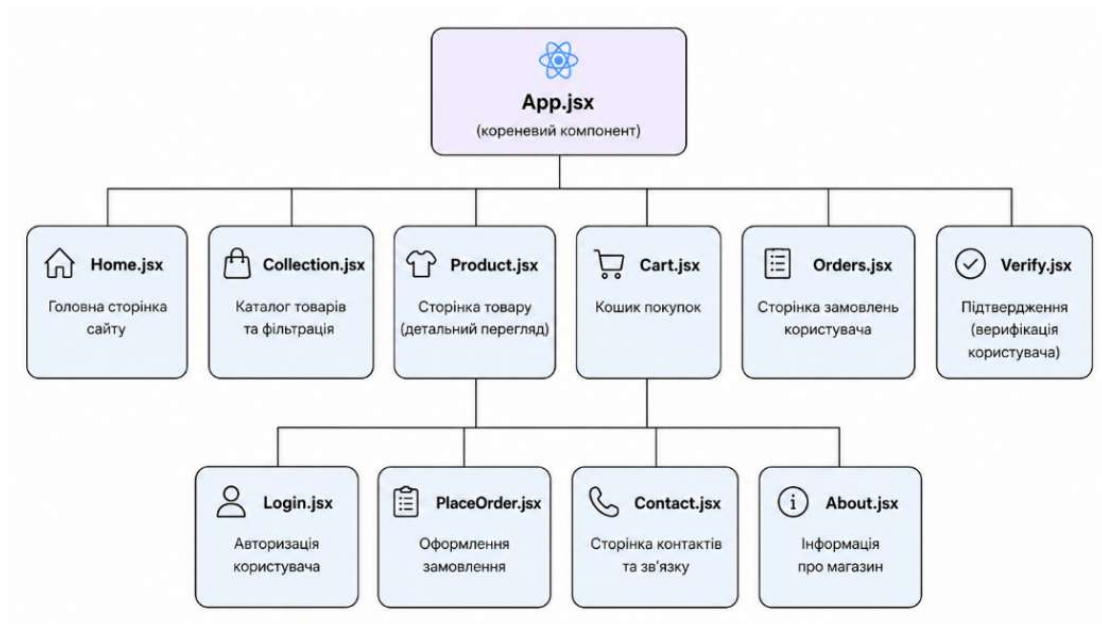


Рис.3.4 Структура клієнтської частини веб-застосунку «YNG-YOUNGSHOP»

### 3.3. Проєктування бази даних

Одним із ключових етапів розроблення веб-застосунку «YNG-YOUNGSHOP» стало створення структури бази даних, яка забезпечує зберігання та обробку інформації про товари, користувачів, замовлення й інші елементи системи.

Для організації бази даних було обрано MongoDB [7]. Використання цієї системи дало змогу працювати з даними у форматі документів, що забезпечує гнучкість структури та спрощує масштабування веб-застосунку.

У межах системи передбачено декілька окремих колекцій, призначених для зберігання різних типів інформації:

- товари;
- користувачі;
- замовлення.

Центральне місце у структурі бази даних займає колекція товарів, яка містить основну інформацію про продукцію інтернет-магазину. Для кожного

товару зберігаються назва, опис, категорія, ціна, набір фотографій та перелік доступних розмірів.

Для роботи із базою даних використовувалася бібліотека Mongoose, яка дозволяє створювати моделі документів та виконувати операції додавання, редагування і видалення інформації. На рисунку 3.5 представлено приклад документа товару у базі даних MongoDB.

```

1  _id: ObjectId('69eb88ebbebcc897eb3b7969')
2  name: "Skinny/"
3  description: "product/"
4  price: 25
5  image: Array (1)
6    0: "https://i.postimg.cc/8cLdZ9vd/hudi.png/"
7  category: "Men/"
8  subCategory: "Topwear/"
9  sizes: Array (2)
10 bestseller: true
11 date: 1777043690171
12 __v: 0

```

Рис.3.5 Приклад документа товару у базі даних MongoDB

У документі товару містяться такі поля:

- name - назва товару;
- description - опис продукції;
- price - вартість;
- image - список зображень;
- category - категорія товару;
- subCategory - підкатегорія;
- sizes - доступні розміри;
- bestseller - статус популярного товару.

Проектування структури бази даних дозволило організувати ефективне зберігання інформації та забезпечити швидку взаємодію серверної частини із даними веб-застосунку.

Окремою частиною бази даних є колекція користувачів, яка використовується для зберігання інформації про акаунти системи. У ній містяться дані, необхідні для авторизації користувачів та роботи кошика покупок.

Кожен документ користувача містить:

- name - ім'я користувача;
- email - електронну пошту;
- password - зашифрований пароль;
- cartData - інформацію про товари у кошику.

Паролі користувачів зберігаються у зашифрованому вигляді за допомогою bcrypt, що забезпечує безпеку персональних даних та захист акаунтів від несанкціонованого доступу.

```
_id: ObjectId('69eb878288f776aa691cca9d')
name: "kama"
email: "petrovad896@gmail.com"
password: "$2b$10$Ui3rLP1mHZ8ja.BR2pPxmuCErdbV9azmElVDA7JiE4s8.Y1qOnEEq"
cartData: Object
__v: 0
```

Рис.3.6 – Приклад структури документа користувача у MongoDB

### 3.4 Організація взаємодії між клієнтською та серверною частинами системи

Для забезпечення коректної роботи веб-застосунку «YNG-YOUNGSHOP» було організовано взаємодію між клієнтською та серверною частинами системи. Передача інформації здійснюється через REST API із використанням HTTP-

запитів та формату JSON [5], що забезпечує швидкий обмін даними між компонентами застосунку.

Клієнтська частина веб-застосунку надсилає запити до сервера для отримання інформації про товари, виконання авторизації користувачів, роботи з кошиком та оформлення замовлень. Після обробки запиту сервер звертається до бази даних MongoDB, виконує необхідні операції та повертає відповідь frontend-частині.

Для реалізації асинхронного обміну даними у React-застосунку використовувалася бібліотека Axios. Її використання дозволило надсилати HTTP-запити без перезавантаження сторінки, що позитивно вплинуло на швидкодію та зручність роботи користувача із системою.

У межах веб-застосунку реалізовано декілька типів HTTP-запитів:

- GET;
- POST;
- передача даних для авторизації та оформлення замовлень;
- DELETE;
- видалення товарів із кошика.

Передача інформації між серверною та клієнтською частинами здійснюється у форматі JSON, що забезпечує зручну обробку даних та сумісність між різними компонентами системи.

Для захисту облікових записів користувачів було реалізовано авторизацію на основі JWT-токенів [11]. Після успішного входу до системи сервер генерує спеціальний токен, який надалі використовується для перевірки прав доступу до захищених функцій веб-застосунку.

У серверній частині були створені API-маршрути для виконання основних функцій системи:

- отримання списку товарів;
- реєстрації користувачів;
- авторизації;
- роботи з кошиком;

- оформлення замовлень;
- перегляду інформації про замовлення.

Окрім цього, сервер забезпечує перевірку правильності введених даних та обробку можливих помилок під час виконання запитів. Значна увага приділялася стабільності роботи API, швидкості передачі інформації та безпечній взаємодії користувачів із системою.

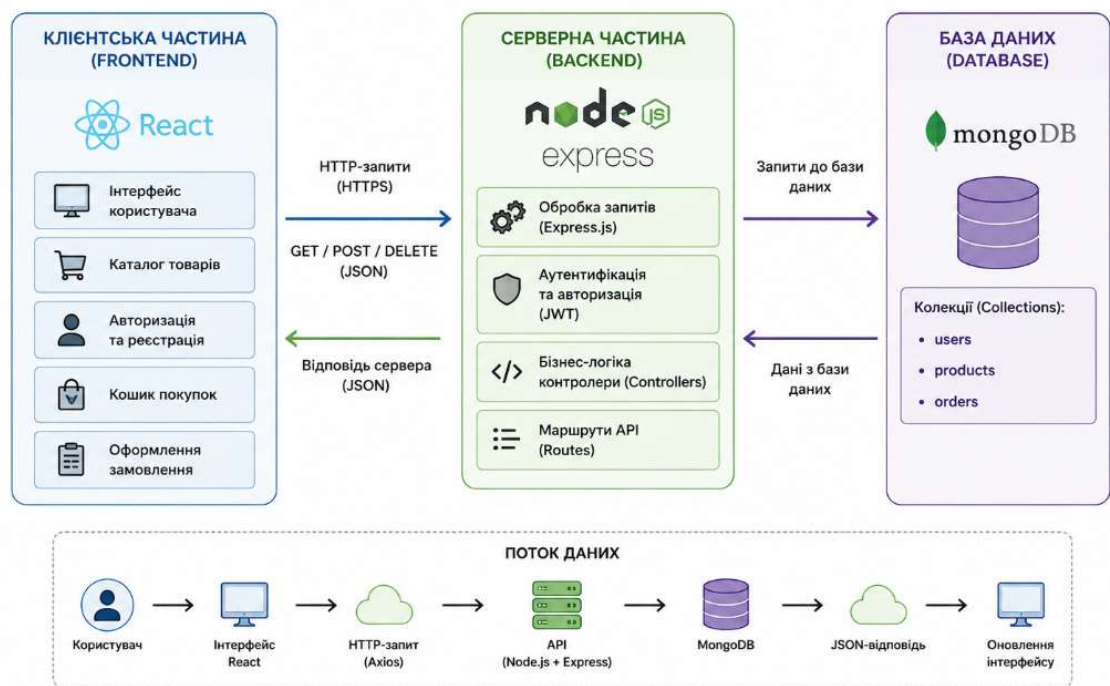


Рис.3.7 Схема взаємодії клієнтської та серверної частин веб-застосунку.

## 4. ПРАКТИЧНА РЕАЛІЗАЦІЯ ВЕБ-ЗАСТОСУНКУ

### 4.1 Створення клієнтської частини системи

Практична реалізація веб-застосунку «YNG-YOUNGSHOP» розпочиналася зі створення клієнтської частини системи, оскільки саме frontend відповідає за взаємодію користувача із онлайн-магазином. Основним завданням на цьому етапі було створення сучасного інтерфейсу, який поєднує адаптивність, швидку роботу та стилістику готичного бренду.

Для розробки клієнтської частини використовувалася бібліотека React [4]. Даний інструмент дозволив організувати структуру веб-застосунку у вигляді окремих компонентів, кожен із яких виконує власну функцію. Завдяки цьому код став більш структурованим та зручним для подальшого редагування.

У процесі реалізації були створені основні сторінки онлайн-магазину:

- головна сторінка;
- каталог продукції;
- сторінка окремого товару;
- кошик покупок;
- авторизація користувача;
- оформлення замовлення;
- сторінка замовлень.

Навігація між сторінками реалізована за допомогою React Router. На практиці це дозволило зробити переходи між сторінками більш плавними без повторного завантаження браузера. Такий підхід позитивно впливає на швидкість роботи веб-застосунку та покращує користувацький досвід.

Особливу увагу під час створення інтерфейсу було приділено зовнішньому вигляду веб-застосунку. Основою дизайну став темний мінімалістичний стиль із використанням готичних елементів, контрастних кольорів та тематичних зображень продукції.

Для стилізації сторінок використовувався Tailwind CSS [8]. Даний фреймворк дозволив швидко створювати адаптивний дизайн та спростив роботу із оформленням компонентів. У результаті інтерфейс коректно відображається як на мобільних пристроях, так і на персональних комп'ютерах.



Рис.4.1 Головна сторінка веб-застосунку «YNG-YOUNGSHOP»

Головна сторінка містить банер бренду, блок із новими колекціями, список популярних товарів та елементи навігації. Також на сторінці розміщені інформаційні блоки із коротким описом переваг онлайн-магазину.

Для роботи із даними у React-застосунку використовувався Context API. За його допомогою було реалізовано централізоване зберігання інформації про товари, кошик та стан авторизації користувача. Це значно спростило передачу даних між компонентами системи.

У frontend-частині реалізовано сторінку каталогу товарів із можливістю фільтрації продукції за категоріями. Користувач може переглядати доступні товари та переходити до детальної сторінки продукції.



Рис.4.2 представлено сторінку каталогу товарів.

У системі реалізовано сторінку кошика покупок, де автоматично відображається список обраних товарів та розраховується загальна вартість замовлення.

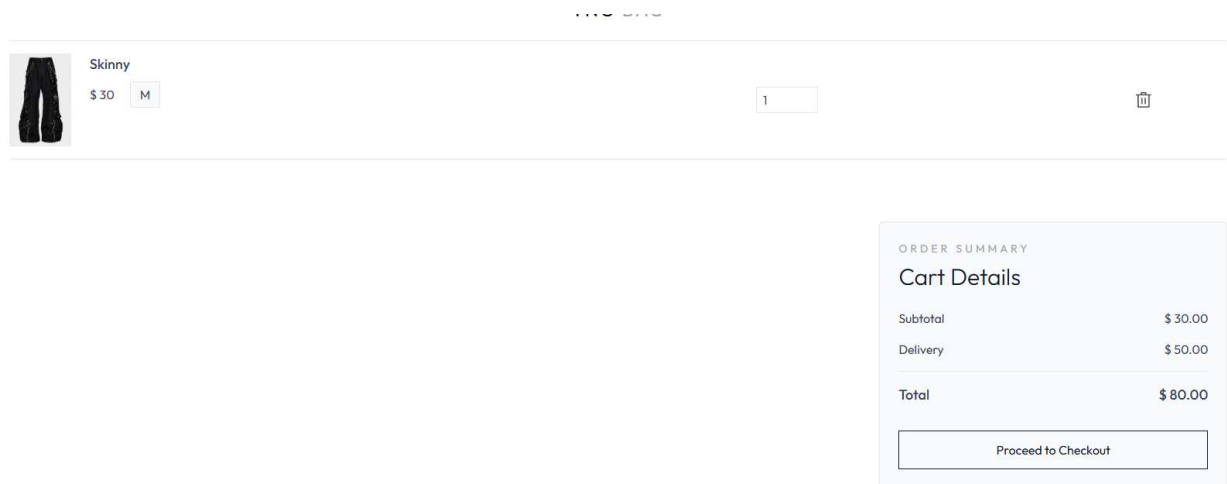


Рис.4.3 Сторінка кошика покупок

Після додавання товарів користувач може перейти до сторінки оформлення замовлення та ввести необхідні дані доставки.

The image shows a checkout page layout. On the left is a form for user details with fields for First name, Last name, Email address, Street, City, State, Zipcode, Country, and Phone. On the right is a 'Cart Details' summary box showing Subtotal (\$30.00), Delivery (\$50.00), and Total (\$80.00), with a 'Proceed to Checkout' button. Below the cart summary is a 'PAYMENT METHOD' section with three options: stripe, Razorpay, and CASH ON DELIVERY (which is selected). A 'PLACE ORDER' button is at the bottom right.

Рис.4.4 Сторінка оформлення доставки

## 4.2 Розробка backend-частини веб-застосунку

Щоб забезпечити повноцінної роботи онлайн-магазину «YNG-YOUNGSHOP» була створена серверна частина системи, яка відповідає за обробку запитів користувачів, взаємодію із базою даних та виконання логіки веб-застосунку.

Backend-частина реалізована за допомогою Node.js [5]. Використання даного середовища дозволило створити швидкий серверний застосунок, який підтримує асинхронну обробку запитів та ефективно працює із великою кількістю даних.

Для організації серверної логіки використовувався Express.js [6]. На його основі були створені API-маршрути, через які клієнтська частина взаємодіє із сервером.

Структура backend-проєкту організована у вигляді окремих модулів. Це дозволило зробити код більш структурованим та спростило роботу із різними частинами системи.

У процесі реалізації були створені:

- маршрути для роботи із товарами;
- система авторизації;
- API для оформлення замовлень;
- логіка роботи кошика;
- взаємодія із MongoDB.

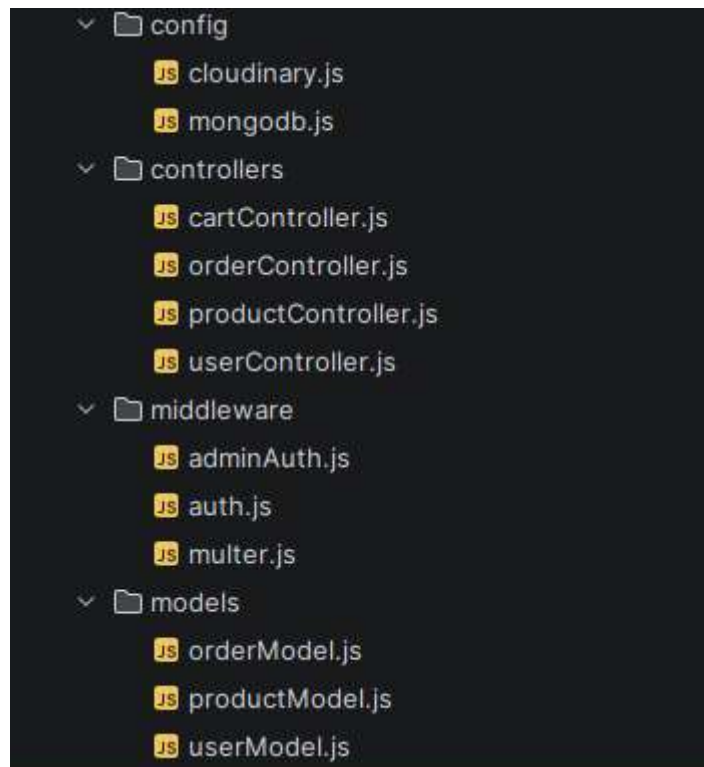


Рис4.3 – Структура backend-проєкту «YNG-YOUNGSHOP»

### 4.3 Реалізація бази даних та роботи з товарами

Зберігання інформації у веб-застосунку «YNG-YOUNGSHOP» використовувалася база даних MongoDB [7]. Дана система дозволяє працювати із документами у форматі BSON та добре підходить для веб-застосунків, створених на основі JavaScript-технологій.

У процесі реалізації було створено окремі колекції для:

- товарів;
- користувачів;

- замовлень.

Основною колекцією системи є products, у якій зберігається інформація про продукцію онлайн-магазину. Для кожного товару у базі даних містяться назва, опис, категорія, вартість, список зображень та доступні розміри.

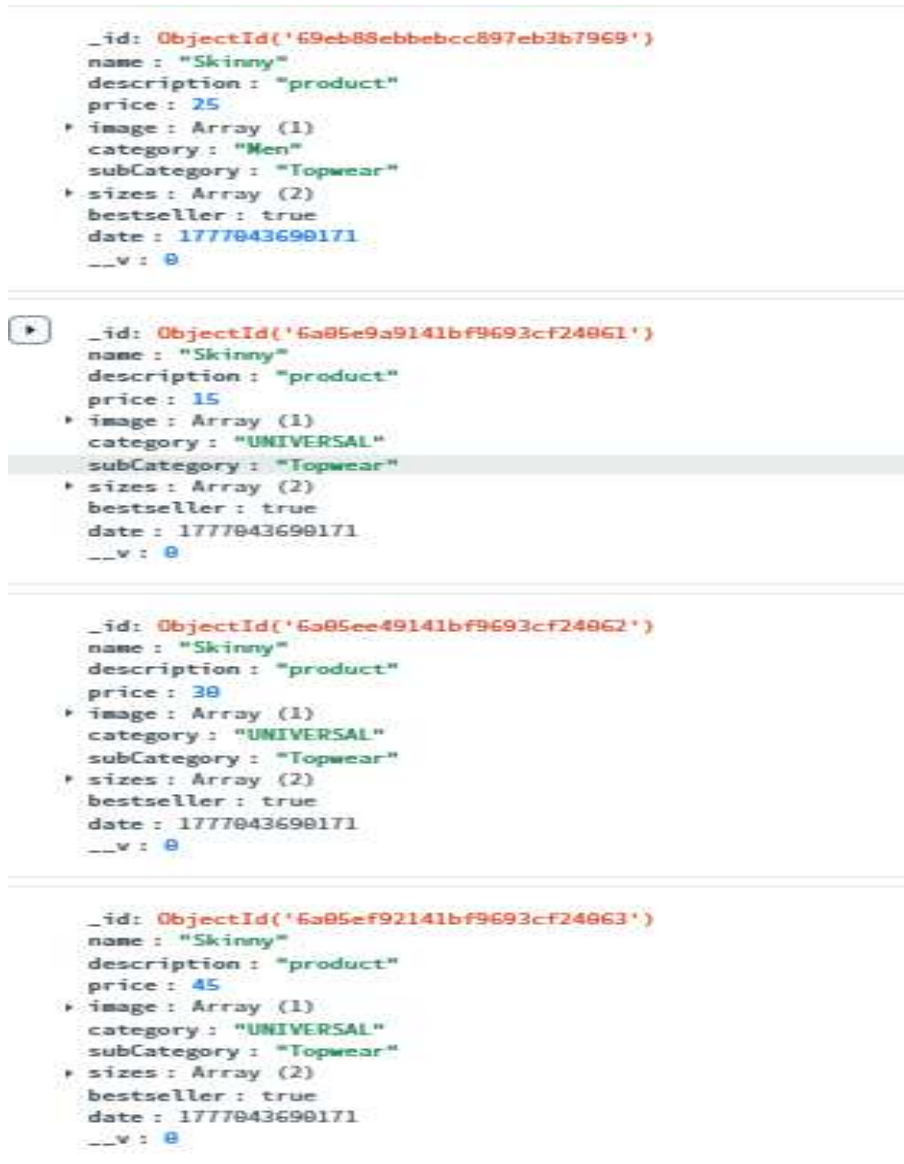


Рис.4.5 – Колекція товарів у MongoDB

Взаємодія серверної частини із базою даних використовувалася бібліотека Mongoose. За допомогою моделей у системі реалізовано створення, редагування та отримання інформації із колекцій MongoDB.

У backend-частині була створена модель товару, яка визначає структуру документа продукції. У моделі описуються всі необхідні поля товару, що використовуються у веб-застосунку.

```
import mongoose from "mongoose";

const productSchema : Schema<any, Model<...>, { ... }, { ... }, { ... }, DefaultSchemaOptions, { ... }> = new mongoose.Schema( definition: {
  name: { type: String, required: true },
  description: { type: String, required: true },
  price: { type: Number, required: true },
  image: { type: Array, required: true },
  category: { type: String, required: true },
  subCategory: { type: String, required: true },
  sizes: { type: Array, required: true },
  bestseller: { type: Boolean },
  date: { type: Number, required: true }
})

const productModel : Model<...> | ... = mongoose.models.product || mongoose.model( name: "product", productSchema);

export default productModel Show usages
```

Рис.4.6 Модель товару у backend-частині веб-застосунку

Даний функціонал є важливою частиною веб-застосунку, оскільки забезпечує доступ користувачів до оформлення замовлень та персональних даних акаунта

Під час реєстрації користувач вводить ім'я, електронну пошту та пароль. Після надсилання форми серверна частина виконує перевірку введених даних та створює новий акаунт у базі даних MongoDB.



Рис.4.7 Сторінка реєстрації користувача

```

_id: ObjectId('69eb878288f776aa691cca9d')
name : "kama"
email : "petrovad896@gmail.com"
password : "$2b$10$Ui3rLPLmHZ8ja.BR2pPxmuCErdbV9azmElVDA7JiE4s8.Y1qOnEEq"
  ▶ cartData : Object
  __v : 0

```

Рис.4.8 Реалізовано додавання користувача в БД

#### 4.4 Аналіз роботи та тестування системи

Після завершення реалізації веб-застосунку «YNG-YOUNGSHOP» було проведено перевірку роботи основних функціональних можливостей системи. Основною метою тестування стало визначення стабільності роботи веб-застосунку, коректності взаємодії між компонентами та перевірка роботи користувацького функціоналу.

У процесі перевірки тестувалися сторінки каталогу продукції, система авторизації, кошик покупок, оформлення замовлення та взаємодія клієнтської частини із сервером. Особлива увага приділялася перевірці передачі даних між frontend та backend-частинами системи.

Під час тестування було встановлено, що веб-застосунок коректно обробляє HTTP-запити та взаємодіє із базою даних MongoDB. Інформація про товари, користувачів та замовлення успішно зберігається у системі та відображається у клієнтській частині веб-застосунку.

Було перевірено роботу авторизації користувачів. Після введення коректних даних система успішно виконує вхід до акаунта та створює JWT-токен для подальшої взаємодії із серверною частиною.

Окремо тестувалася робота кошика покупок. Під час додавання товарів система автоматично оновлює список продукції, виконує розрахунок загальної вартості та передає інформацію до серверної частини.

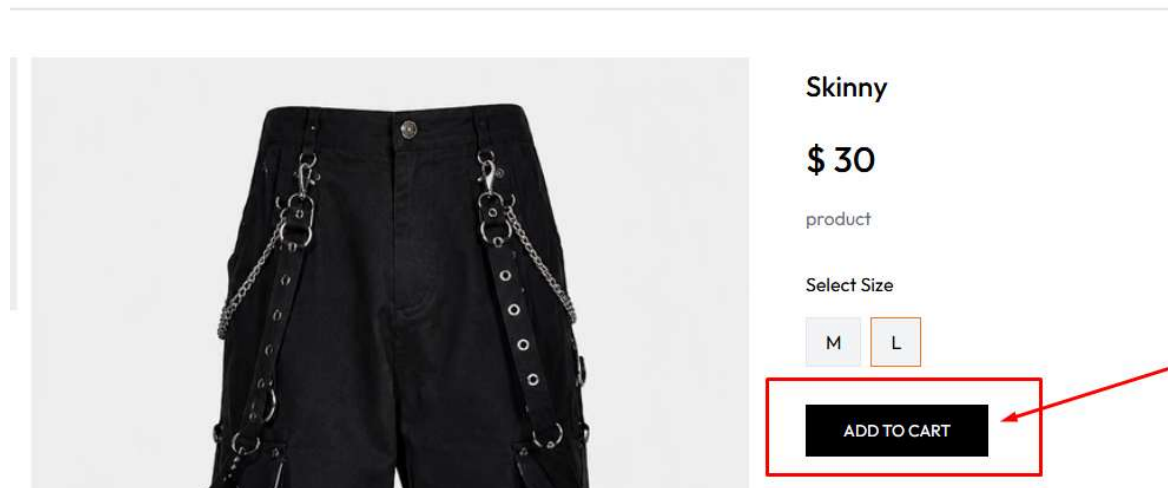


Рис.4.8 Сторінка додавання в кошик

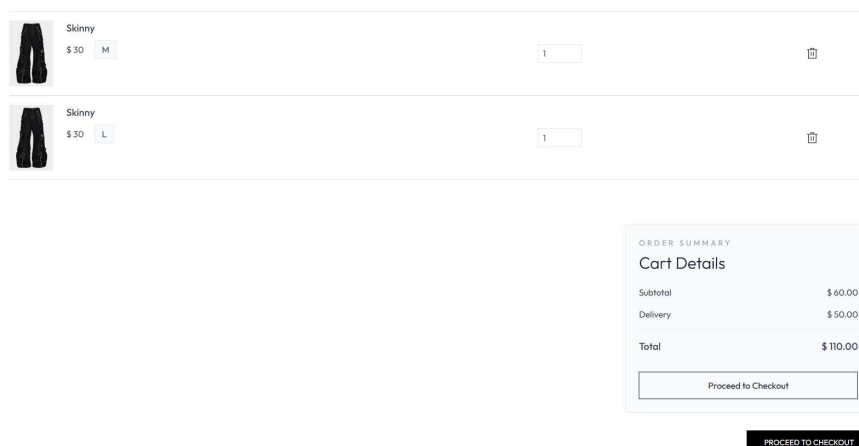


Рис.4.9 Сторінка кошика з товаром

## ВИСНОВКИ

Розвиток електронної комерції суттєво змінив підхід до продажу товарів та взаємодії між продавцем і покупцем. Сьогодні онлайн-магазини є не лише платформою для придбання продукції, а й важливою частиною бренду, яка формує стиль, атмосферу та загальний користувацький досвід. Саме тому під час виконання кваліфікаційної роботи основна увага приділялася створенню сучасного веб-застосунку для продажу одягу, який поєднує функціональність, адаптивність та унікальний дизайн.

У процесі виконання роботи було проведено аналіз предметної області та досліджено особливості сучасних онлайн-магазинів одягу. Під час аналізу аналогів вдалося визначити основні принципи побудови інтерфейсу, організації каталогу продукції та взаємодії користувача із системою. Дослідження існуючих рішень дозволило краще зрозуміти вимоги до сучасних веб-застосунків електронної комерції та врахувати їх під час створення власного проєкту.

У результаті було розроблено веб-застосунок «YNG-YOUNGSHOP», який представляє собою онлайн-магазин одягу у готичному стилі. Під час створення системи було реалізовано повноцінну клієнтську та серверну частини, організовано взаємодію із базою даних та реалізовано REST API для обміну інформацією між компонентами веб-застосунку.

Frontend-частина системи створювалася із використанням React. Компонентний підхід дозволив побудувати структуру веб-застосунку у вигляді незалежних елементів інтерфейсу, що значно спростило розробку та подальшу підтримку програмного коду. Окрему увагу було приділено дизайну веб-застосунку, оскільки саме візуальна складова формує стиль бренду та впливає на сприйняття користувачем онлайн-магазину.

Для стилізації сторінок використовувався Tailwind CSS, завдяки чому вдалося реалізувати адаптивний інтерфейс та забезпечити коректне відображення сторінок на різних типах пристроїв. Усі сторінки веб-застосунку були адаптовані під мобільні телефони, планшети та персональні комп'ютери.

Серверна частина системи була реалізована за допомогою Node.js та Express.js. У межах backend-частини створено API для роботи із товарами, авторизації користувачів, кошиком покупок та оформленням замовлень. Взаємодія між frontend та backend здійснюється через HTTP-запити у форматі JSON.

Для зберігання інформації використовувалася MongoDB. У базі даних реалізовано колекції товарів, користувачів та замовлень.

Після завершення розробки було проведено тестування основного функціоналу системи. Під час перевірки тестувалися сторінки веб-застосунку, взаємодія із базою даних, робота API та коректність обробки запитів користувачів. Результати тестування підтвердили стабільну роботу системи та коректне функціонування основних компонентів веб-застосунку.

Отриманий результат повністю відповідає поставленій меті кваліфікаційної роботи. Розроблений веб-застосунок може бути використаний як основа для подальшого розвитку онлайн-магазину, розширення функціоналу та інтеграції нових можливостей електронної комерції.

Робота пройшла апробація. За її результатами надруковано наступні тези доповідей:

1. Федоренко О.І. Гаманюк І.М. Матеріали Розробка веб-застосунку для інтернет-магазину одягу з використанням сучасних технологій JavaScriptII Всеукраїнської науково-технічної конференції «Виклики та рішення в програмній інженерії», 26 листопада 2025 року, Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. – К.: ДУІКТ, 2025, С.468-471.

2. Федоренко О.І., Гаманюк І.М. Визначення вимог до веб-застосунку інтернет-магазину одягу на основі сучасних технологій електронної комерції. Матеріали Всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях», 24 квітня 2026 року, Київ, ДУІКТ. Збірник тез.

## ПЕРЕЛІК ПОСТЛАНЬ

1. Sommerville I. Software Engineering. 10th ed. Boston: Pearson, 2016. 816 p.
2. Pressman R. S., Maxim B. R. Software Engineering: A Practitioner's Approach. 9th ed. New York: McGraw-Hill, 2019. 784 p.
3. MDN Web Docs. JavaScript Guide. URL: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>.
4. React Documentation. URL: <https://react.dev/> (date of access: 15.03.2026).
5. Node.js Documentation. URL: <https://nodejs.org/en/docs> (date of access: 16.03.2026).
6. Express.js Documentation. URL: <https://expressjs.com/> (date of access: 16.03.2026).
7. MongoDB Documentation. URL: <https://www.mongodb.com/docs/> (date of access: 17.03.2026).
8. Tailwind CSS Documentation. URL: <https://tailwindcss.com/docs> (date of access: 18.03.2026).
9. Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. Boston: Addison-Wesley, 1995. 395 p.
10. Freeman A. Pro React 18. New York: Apress, 2023. 950 p.
11. Flanagan D. JavaScript: The Definitive Guide. 7th ed. Sebastopol: O'Reilly Media, 2020. 704 p.
12. Osmani A. Learning JavaScript Design Patterns. Sebastopol: O'Reilly Media, 2023. 254 p.
13. Wieruch R. The Road to React. Berlin: Self-Published, 2022. 520 p.
14. Nixon R. Learning PHP, MySQL & JavaScript. 6th ed. Sebastopol: O'Reilly Media, 2021. 816 p.
15. Figma Documentation. URL: <https://help.figma.com/> (date of access: 20.03.2026).

16. Git Documentation. URL: <https://git-scm.com/doc> (date of access: 21.03.2026).
17. GitHub Docs. URL: <https://docs.github.com/> (date of access: 21.03.2026).
18. Krug S. Don't Make Me Think. 3rd ed. Berkeley: New Riders, 2014. 216 p.
19. Nielsen J. Usability Engineering. San Francisco: Morgan Kaufmann, 1994. 362 p.
20. Patel D., Patel F., Chauhan U. Recommendation Systems: Types, Applications, and Challenges. International Journal of Computing and Digital Systems. 2023. Vol. 13, no. 1. P. 851–868. URL: <https://doi.org/10.12785/ijcds/130168>
21. Bohdana Muzyka. User Flows. Як ця техніка допомагає в роботі над проектами. URL: <https://dou.ua/lenta/columns/user-flows/> (дата звернення: 22.03.2026).
22. WebAIM. Web Accessibility Principles. URL: <https://webaim.org/> (date of access: 22.03.2026).
23. ZARA Official Website. URL: <https://www.zara.com/> (date of access: 23.03.2026).
24. Bershka Official Website. URL: <https://www.bershka.com/> (date of access: 23.03.2026).
25. ASOS Official Website. URL: <https://www.asos.com/> (date of access: 23.03.2026).

## ДОДАТОК А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ



ДЕРЖАВНИЙ УНІВЕРСИТЕТ  
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ  
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ  
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ



Онлайн-магазин одягу "YNG-YOUNGSHOP" для підлітків

Виконав студент 4 курсу  
групи ПД-42  
Федоренко Олексій  
Керівник роботи

старший викладач кафедри Гаманюк Ігор Михайлович

Київ – 2026

## МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

**Мета роботи:** покращення користувацького досвіду під час взаємодії із системою електронної комерції шляхом створення адаптивного онлайн-магазину одягу для підлітків із сучасним інтерфейсом та зручним функціоналом.

**Об'єкт дослідження:** процес функціонування онлайн-магазину для підлітків

**Предмет дослідження:** методи, засоби та технології створення вебастосунку онлайн-магазину одягу для підлітків, а також особливості реалізації користувацького інтерфейсу, серверної частини та взаємодії із базою даних.

## ЗАДАЧІ КВАЛІФІКАЦІЙНОЇ РОБОТИ

1. Провести аналіз сучасних онлайнмагазинів електронної комерції для продажу одягу.
2. Дослідити тенденції розвитку онлайнмагазинів та особливості організації користувацького інтерфейсу.
3. Визначити функціональні та нефункціональні вимоги до онлайнмагазину «YNG-YOUNGSHOP».
4. Спроектувати архітектуру клієнтської та серверної частин системи.
5. Дослідити сучасні засоби та технології реалізації клієнтської частини онлайнмагазину.
6. Дослідити засоби реалізації серверної частини системи.
7. Реалізувати взаємодію онлайнмагазину із базою даних.
8. Реалізувати систему авторизації та реєстрації користувачів.
9. Провести тестування основного функціоналу онлайнмагазину.

## АНАЛІЗ АНАЛОГІВ

| Характеристика магазинів                       | ZARA  | Bershka | ASOS  | YNG-YOUNGSHOP |
|------------------------------------------------|-------|---------|-------|---------------|
| Зручний мінімалістичний інтерфейс              | -     | -       | -     | +             |
| Готичний стиль дизайну                         | -     | -       | -     | +             |
| SPA-архітектура                                | -     | -       | +     | +             |
| Швидка взаємодія без перезавантаження сторінок | -     | -       | +     | +             |
| REST API                                       | -     | -       | +     | +             |
| Авторизація та реєстрація                      | +     | +       | +     | +             |
| Кошик покупок                                  | +     | +       | +     | +             |
| Сучасний стек MERN                             | -     | -       | -     | +             |
| Відсоток підлітків                             | 40/60 | 45/55   | 35/65 | 70/30         |

## ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

### Функціональні вимоги

1. Управління каталогом товарів
2. Управління даними щодо товарів;
3. Пошук та фільтрація товарів;
4. Забезпечення роботи кошика товарів;
5. Управління даними щодо замовлень;
6. Реєстрація, управління даними щодо Користувачів
7. Авторизація користувачів

### Нефункціональні вимоги

1. Час завантаження сторінок вебзастосунку не повинен перевищувати 3 секунд;
2. Система повинна коректно працювати у браузерях Google Chrome, Safari та Mozilla Firefox;
3. Онлайн-магазин повинен адаптуватися під екрани смартфонів, планшетів та персональних комп'ютерів;
4. Паролі користувачів повинні зберігатися у зашифрованому вигляді;

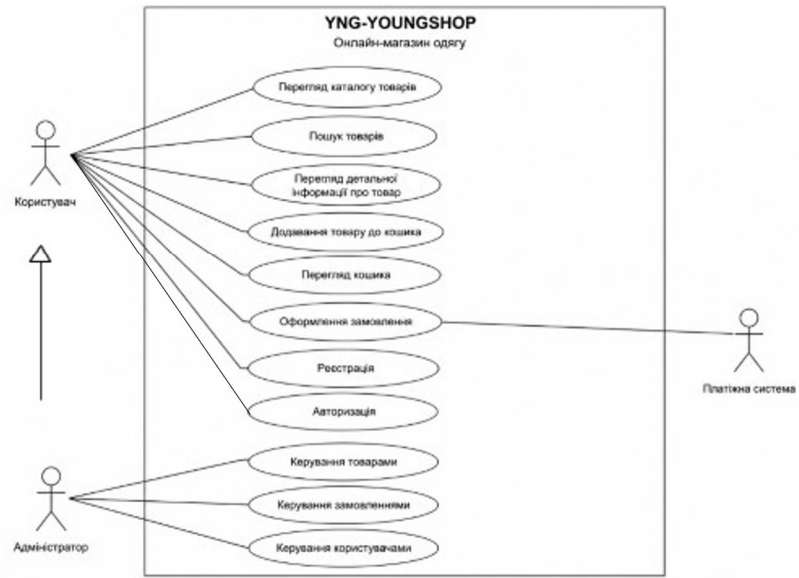
5

## ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ



6

### ДІАГРАМА ПРЕЦЕДЕНТІВ КОРИСТУВАЧА



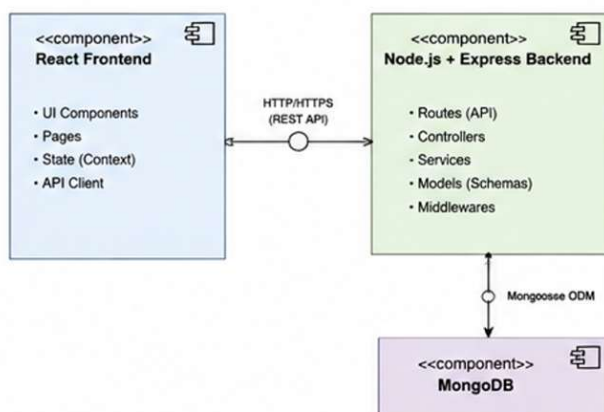
7

### ДІАГРАМА ПРЕДМЕТНОЇ ГАЛУЗІ



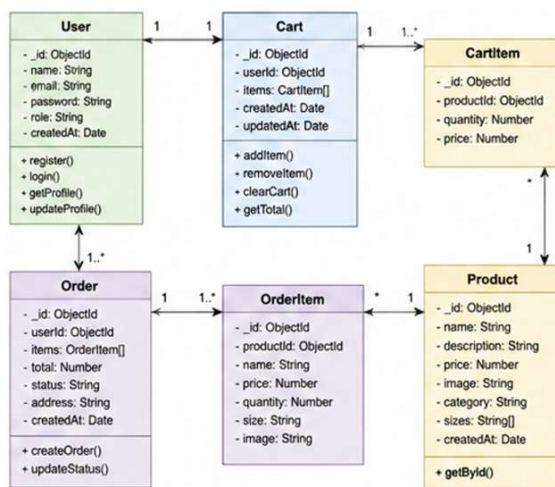
8

## ДІАГРАМА КОМПОНЕНТІВ



9

## ДІАГРАМА КЛАСІВ(ОСНОВНІ СУТНОСТІ)



10

МОДЕЛЮВАННЯ ВЗАЄМОДІЇ КОРИСТУВАЧА ІЗ ЗАСТОСУНОКОМ



ТЕСТУВАННЯ

| Функціональні вимоги            | Тест-кейс 1 | Тест-кейс 2 | Тест-кейс 3 | Тест-кейс 4 | Тест-кейс 5 |
|---------------------------------|-------------|-------------|-------------|-------------|-------------|
| Реєстрація користувача          | +           |             |             |             |             |
| Авторизація користувача         |             | +           |             |             |             |
| Перегляд каталогу товарів       |             | +           | +           |             |             |
| Додавання товару до кошика      |             |             | +           | +           |             |
| Оформлення замовлення           |             |             |             | +           | +           |
| Пошук та фільтрація товарів     |             | +           | +           |             |             |
| Взаємодія із базою даних        | +           | +           | +           | +           | +           |
| Відображення деталей товару     |             |             | +           | +           |             |
| Редагування профілю користувача |             | +           |             | +           |             |
| Вихід із системи                | +           |             |             |             | +           |

## ЕКРАННІ ФОРМИ



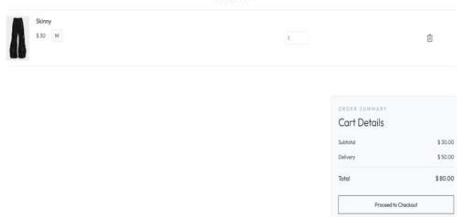
Головна сторінка сайту



Представлено сторінку каталогу товарів

13

## ЕКРАННІ ФОРМИ



Сторінка кошика покупок



Сторінка оформлення доставки



Сторінка реєстрації користувача

14

## АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

### Тези доповідей:

- Федоренко О.І. Гаманюк І.М. Матеріали Розробка веб-застосунку для інтернет-магазину одягу з використанням сучасних технологій JavaScriptlI Всеукраїнської науково-технічної конференції «Виклики та рішення в програмній інженерії», 26 листопада 2025 року, Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. – К.: ДУІКТ, 2025, С.468 -471.
- Федоренко О.І., Гаманюк І.М. Визначення вимог до веб-застосунку інтернет-магазину одягу на основі сучасних технологій електронної комерції. Матеріали Всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях», 24 квітня 2026 року, Київ, ДУІКТ. Збірник тез.

15

## ВИСНОВКИ

1. У ході виконання кваліфікаційної роботи було проведено аналіз сучасних онлайн-магазинів електронної комерції для продажу одягу та досліджено особливості організації користувацького інтерфейсу сучасних систем електронної комерції.
2. На основі проведеного аналізу було визначено функціональні та нефункціональні вимоги до онлайн-магазину «YNG-YOUNGSHOP» та спроектовано архітектуру клієнтської і серверної частин системи.
3. Під час роботи було досліджено сучасні засоби та технології реалізації frontend- і backend-частини онлайн-магазину, а також реалізовано взаємодію системи із базою даних MongoDB.
4. У процесі практичної реалізації було створено систему авторизації та реєстрації користувачів, реалізовано основний функціонал онлайн-магазину та забезпечено коректну взаємодію між компонентами системи.
5. Після завершення розробки проведено тестування основного функціоналу онлайн-магазину, результати якого підтвердили стабільну роботу системи та коректне виконання основних операцій користувача.

16

## ДОДАТОК Б. ЛІСТИНГИ ПРОГРАМНИХ МОДУЛІВ

```

import React from 'react'
import { Routes, Route } from 'react-router-dom'
import Home from './pages/Home'
import Collection from './pages/Collection'
import About from './pages/About'
import Contact from './pages/Contact'
import Product from './pages/Product'
import Cart from './pages/Cart'
import Login from './pages/Login'
import PlaceOrder from './pages/PlaceOrder'
import Orders from './pages/Orders'
import Navbar from './components/Navbar'
import Footer from './components/Footer'
import SearchBar from './components/SearchBar'
import { ToastContainer, toast } from 'react-
toastify';
import 'react-toastify/dist/ReactToastify.css';
import Verify from './pages/Verify'

const App = () => {
  return (
    <div className='px-4 sm:px-[5vw] md:px-[7vw]
lg:px-[9vw]'>
      <ToastContainer />
      <Navbar />
      <SearchBar />
      <Routes>
        <Route path='/' element={<Home />} />
        <Route
          element={<Collection />} />
          path='/collection'
        <Route path='/about' element={<About />} />
        <Route path='/contact' element={<Contact />} />
        <Route
          element={<Product />} />
          path='/product/:productId'
        <Route path='/cart' element={<Cart />} />
        <Route path='/login' element={<Login />} />
        <Route
          element={<PlaceOrder />} />
          path='/place-order'
        <Route path='/orders' element={<Orders />} />
        <Route path='/verify' element={<Verify />} />
      </Routes>
      <Footer />
    </div>
  )
}

export default App
import { createContext, useEffect, useState } from
"react";
import { toast } from "react-toastify";
import { useNavigate } from "react-router-dom";
import axios from 'axios'

export const ShopContext = createContext();

const ShopContextProvider = (props) => {

  const currency = '$ ';
  const delivery_fee = 50;
  const
    backendUrl
    =
import.meta.env.VITE_BACKEND_URL
  const [search, setSearch] = useState("");
  const [showSearch, setShowSearch] =
useState(false);
  const [cartItems, setCartItems] = useState({});
  const [products, setProducts] = useState([]);

```

```

const [token, setToken] = useState("")
const navigate = useNavigate();

const addToCart = async (itemId, size) => {

  if (!size) {
    toast.error('Select Product Size');
    return;
  }

  let cartData = structuredClone(cartItems);

  if (cartData[itemId]) {
    if (cartData[itemId][size]) {
      cartData[itemId][size] += 1;
    }
    else {
      cartData[itemId][size] = 1;
    }
  }
  else {
    cartData[itemId] = {};
    cartData[itemId][size] = 1;
  }
  setCartItems(cartData);

  if (token) {
    try {
      await
        axios.post(backendUrl
          +
          '/api/cart/add', { itemId, size }, { headers: { token } })

    } catch (error) {
      console.log(error)
      toast.error(error.message)
    }
  }

  const getCartCount = () => {
    let totalCount = 0;
    for (const items in cartItems) {
      for (const item in cartItems[items]) {
        try {
          if (cartItems[items][item] > 0) {
            totalCount
              +=
              cartItems[items][item];
          }
        } catch (error) {
        }
      }
    }
    return totalCount;
  }

  const updateQuantity = async (itemId, size,
quantity) => {

    let cartData = structuredClone(cartItems);

    cartData[itemId][size] = quantity;

    setCartItems(cartData)

    if (token) {

```

```

    try {
      await axios.post(backendUrl +
'/api/cart/update', { itemId, size, quantity }, { headers: { token
}})

    } catch (error) {
      console.log(error)
      toast.error(error.message)
    }
  }

const getCartAmount = () => {
  let totalAmount = 0;
  for (const items in cartItems) {
    let itemInfo = products.find((product) =>
product._id === items);
    for (const item in cartItems[items]) {
      try {
        if (cartItems[items][item] > 0) {
          totalAmount += itemInfo.price *
cartItems[items][item];
        }
      } catch (error) {
      }
    }
  }
  return totalAmount;
}

const getProductsData = async () => {
  try {

    const response = await axios.get(backendUrl
+ '/api/product/list')
    if (response.data.success) {
      setProducts(response.data.products.reverse())
    } else {
      toast.error(response.data.message)
    }

  } catch (error) {
    console.log(error)
    toast.error(error.message)
  }

  const getUserCart = async ( token ) => {
    try {

      const response = await
axios.post(backendUrl + '/api/cart/get', {}, {headers: {token}})
      if (response.data.success) {
        setCartItems(response.data.cartData)
      }
    } catch (error) {
      console.log(error)
      toast.error(error.message)
    }
  }

  useEffect(() => {
    getProductsData()
  }, [])

  useEffect(() => {

```

```

    if (!token && localStorage.getItem('token')) {
      setToken(localStorage.getItem('token'))
      getUserCart(localStorage.getItem('token'))
    }
    if (token) {
      getUserCart(token)
    }
  }, [token])

  const value = {
    products, currency, delivery_fee,
    search, setSearch, showSearch,
    setShowSearch,
    cartItems, addToCart, setCartItems,
    getCartCount, updateQuantity,
    getCartAmount, navigate, backendUrl,
    setToken, token
  }

  return (
    <ShopContext.Provider value={value}>
      {props.children}
    </ShopContext.Provider>
  )
}

export default ShopContextProvider;
import React, { useContext, useEffect, useState }
from 'react'
import { ShopContext } from
'../context/ShopContext'
import Title from '../components/Title';
import { assets } from '../assets/assets';
import CartTotal from '../components/CartTotal';

const Cart = () => {

  const { products, currency, cartItems,
updateQuantity, navigate } = useContext(ShopContext);

  const [cartData, setCartData] = useState([]);

  useEffect(() => {

    if (products.length > 0) {
      const tempData = [];
      for (const items in cartItems) {
        for (const item in cartItems[items]) {
          if (cartItems[items][item] > 0) {
            tempData.push({
              _id: items,
              size: item,
              quantity: cartItems[items][item]
            })
          }
        }
      }
      setCartData(tempData);
    }
  }, [cartItems, products])

  return (
    <div className="border-t pt-14">

      <div className="text-2xl mb-6 text-center">
        <h1 className="tracking-widest">
          YNG <span className="text-gray-
400">BAG</span>
        </h1>

```



```

        <div className="relative z-10 flex flex-
col justify-center items-center p-10 text-white w-full">
        </div>

    </div>

    { /* RIGHT SIDE */ }
    <div className="w-full md:w-1/2 flex
items-start justify-center pt-20">

        <form
            onSubmit={handleSubmit}
            className="w-full max-w-sm p-8"
        >
            <h2 className="text-2xl font-
semibold mb-6">
                {mode === "login" ? "Sign In" :
"Create Account"}
            </h2>

            {mode === "signup" && (
                <input
                    type="text"
                    name="name"
                    placeholder="Name"
                    value={data.name}
                    onChange={handleChange}
                    className="w-full mb-4 px-3 py-
2 border border-gray-300"
                    required
                />
            )}

            <input
                type="email"
                name="email"
                placeholder="Email"
                value={data.email}
                onChange={handleChange}
                className="w-full mb-4 px-3 py-2
border border-gray-300"
                required
            />

            <input
                type="password"
                name="password"
                placeholder="Password"
                value={data.password}
                onChange={handleChange}
                className="w-full mb-6 px-3 py-2
border border-gray-300"
                required
            />

            <button className="w-full bg-black
text-white py-2 mb-4">
                {mode === "login" ? "Login" :
"Register"}
            </button>

            <div className="text-sm text-gray-
500 flex justify-between">
                <span
                    className="cursor-
pointer">Forgot?</span>

                {mode === "login" ? (
                    <span
                        onClick={() =>
setMode("signup")} className="cursor-pointer">

```

```

Create account
</span>
        ) : (
            <span
                onClick={() =>
setMode("login")} className="cursor-pointer">
                Back to login
            </span>

```