

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

«Сайт з продажу еко товарів»

Виконав студент групи ЗПЗк-21-1
ОСТАПЧУК Богдан Володимирович

Керівник роботи:
ВЛАСЕНКО Олег Васильович

Рецензент:
ТОЛСТОЙ Ігор Анатолійович

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії
програмного забезпечення
Тетяна ВАКАЛЮК
«26» лютого 2024 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ОСТАПЧУК Богдан Володимирович**

Керівник роботи: **ВЛАСЕНКО Олег Васильович**

Тема роботи: **«Сайт з продажу еко товарів»**, затверджена Наказом закладу вищої освіти наказ №74/с від 26.02.2024р.

Вихідні дані для роботи: **розробка магазину з продажу еко товарів**

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Власенко О.В.	27.02.2024р.	27.02.2024р.
2	Власенко О.В.	27.02.2024р.	27.02.2024р.
3	Власенко О.В.	27.02.2024р.	27.02.2024р.

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з веб-сайту для продажу еко товарів та пояснювальної записки. Пояснювальна записка обсягом 70 сторінок включає 41 ілюстрації та 7 таблиць.

Метою цієї роботи є створення веб-додатка для продажу еко товарів з використанням сучасних технологій та ефективним зберіганням даних.

Архітектура сайту для продажу еко товарів побудована для створення динамічного сайту, а також реляційної бази даних для зберігання інформації про товари, користувачів і замовлення. Інтеграція платіжної забезпечує безпечні онлайн-платежі. Основні компоненти включають головну сторінку, каталог товарів, сторінку продукту, кошик, оформлення замовлення, кабінет користувача та панель адміністратора. Система автентифікації та авторизації гарантує безпеку доступу до сайту. Ця архітектура забезпечує прозорість, надійність і зручність використання для користувачів.

Веб-додаток має інтуїтивно зрозумілий інтерфейс, що дозволяє користувачам з легкістю переглядати продукти, здійснювати покупки, реєструватись та зв'язуватися з експертами у сфері еко товарів.

КЛЮЧОВІ СЛОВА: ВЕБ-ДОДАТОК, АДМІН ПАНЕЛЬ, САЙТ, ЕКО ТОВАРИ, REACT, STRIPE.

					ІПЗ.КР.Б – 121 – 24 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Остапчук Б.В.			Сайт з продажу еко товарів			
Керівник		Власенко О.В.						
Н. контр.		О.Тичина						
Зав. каф.		Т. Вакалюк						
					Літ.	Арк.	Аркушів	
							3	70
					Житомирська політехніка, група ІПЗк-21-1			

ABSTRACT

The bachelor's thesis consists of a website for the sale of eco-products and an explanatory note. The 70-page explanatory note includes 41 illustrations and 7 tables.

The purpose of this work is to create a web application for the sale of eco goods using modern technologies and efficient data storage.

The architecture of the site for the sale of eco goods is built to create a dynamic site, as well as a relational database for storing information about goods, users and orders. Payment integration ensures secure online payments. The main components include the main page, product catalog, product page, cart, checkout, user account and admin panel. The authentication and authorization system guarantees the security of access to the site. This architecture ensures transparency, reliability and ease of use for users.

The web application has an intuitive interface that allows users to easily browse products, make purchases, register and connect with experts in the field of eco goods.

KEYWORDS: WEB APPLICATION, ADMIN PANEL, SITE, ECO GOODS, REACT, STRIPE.

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	6
ВСТУП.....	7
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ЗАСТОСУВАННЯ МАГАЗИНУ З ПРОДАЖУ ЕКО ТОВАРІВ	9
1.1 Постановка задачі.....	9
1.2 Аналіз аналогічних програмних продуктів.....	11
1.3 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення	16
Висновки до першого розділу	17
РОЗДІЛ 2. ПРОЕКТУВАННЯ САЙТУ З ПРОДАЖУ ЕКО ТОВАРІВ	19
2.1 Визначення варіантів використання та об'єктно-орієнтованої структури систем.....	19
2.2 Розробка бази даних системи	24
2.3 Проектування та реалізація алгоритмів роботи системи	29
2.4 Реалізація функціоналу інтернет магазину еко товарів.....	31
Висновки до другого розділу	39
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З САЙТОМ З ПРОДАЖУ ЕКО ТОВАРІВ	41
3.1 Порядок встановлення та налаштування параметрів сайту.....	41
3.2 Структура інтерфейсу сайту з продажу еко товарів	42
3.3 Тестування сайту	54
Висновки до третього розділу	55
ВИСНОВКИ.....	57
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	59
ДОДАТКИ.....	61

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

UX/UI – User Experience/User Interface (користувацький досвід/користувацький інтерфейс)

БД – База даних

ВКР – Випускна кваліфікаційна робота

ЕКО – Екологічні (товари, продукти)

Інтерфейс – Користувацький інтерфейс

ОС – Операційна система

ПЗ – Програмне забезпечення

СУБД – Система управління базами даних

ВСТУП

Актуальність теми полягає у тому що, у сучасному світі екологічні проблеми стають все більш актуальними. Зміна клімату, забруднення навколишнього середовища та виснаження природних ресурсів змушують суспільство зосереджуватися на сталому розвитку та використанні органічних продуктів. Все більше і більше людей визнають необхідність змінити свої звички споживання на користь екологічно чистих продуктів, які сприяють збереженню навколишнього середовища.

Веб-сайти, що продають еко-товари, задовольняють цей попит ринку. Це не тільки поширює інформацію про органічні продукти, але й робить органічні продукти доступними для ширшого кола споживачів. Завдяки зручному інтерфейсу користувачі можуть легко знайти потрібний товар, отримати консультацію експертів і прийняти зважені рішення щодо екологічно чистих продуктів.

Крім того, розвиток Інтернет-технологій дозволив створити сучасні та ефективні сайти. Веб-додатки не лише надають можливість купувати товари онлайн, а й забезпечують прозорість та надійність, що є важливим фактором для споживачів.

Тому створення сайту для продажу еко товарів є актуальною темою, яка відповідає сучасним тенденціям у сфері екології та інформаційних технологій. Це сприяє розвитку сталого споживання та сприяє формуванню екологічної свідомості суспільства. Це важливий крок до захисту нашої планети для майбутніх поколінь.

Метою кваліфікаційної роботи є розробка веб-додатку для продажу еко-продуктів, який забезпечує зручність використання, прозорість і надійність.

Основними завданнями робіт є:

Створення інтуїтивно зрозумілого інтерфейсу користувача та адміністратора – розробка зручного і функціонального інтерфейсу для користувача та адміністратора, що спрощує перегляд, пошук і купівлю

екологічних продуктів для користувачів, а також управління замовленнями, товарами тощо для адміністратора.

Інтеграція бази даних – використання надійної системи керування базами даних для ефективного керування інформацією про продукти, користувачів, експертів та замовлення.

Зв'язок з експертами – надання користувачам можливості отримувати поради від експертів у сфері екологічних товарів, що підвищує обізнаність і допомагає приймати обґрунтовані рішення.

Об'єктом дослідження є веб-додаток, спрямований на створення сайту для торгівлі екологічно чистими товарами. Цей веб-додаток має на меті забезпечити зручний та ефективний механізм продажу еко-продуктів, які пропагують здоровий та екологічно свідомий спосіб життя.

Предметом дослідження є використання технології Node.js для розвитку ефективної торгівлі екологічно чистими товарами. Це включає аналіз потреб користувачів і ринкових тенденцій, розробку інтерфейсу, функціональне програмування та інтеграцію бази даних [1].

У цій роботі використовувався системний аналіз, функціональні підходи, та методи проектування інтерфейсу користувача. Розроблений веб-додаток може використовуватися як малим, так і великим бізнесом для продажу екологічних товарів.

Випускна кваліфікаційна робота включає в себе вступ, три розділи, висновки, список використаних джерел із 23 джерел (2 сторінки), 4 додатки (11 сторінки), 7 таблиць та 41 ілюстрацій в тексті. Загальний обсяг роботи становить 70 сторінок, з них основний текст займає 60 сторінок.

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ЗАСТОСУВАННЯ МАГАЗИНУ З ПРОДАЖУ ЕКО ТОВАРІВ

1.1 Постановка задачі

Метою кваліфікаційної роботи є створення веб-сайту для продажу еко-продуктів з використанням сучасних технологій, які можна легко використовувати та розширювати в майбутньому.

Сайт з продажу еко товарів має включати наступні розділи: головна сторінка, каталог товарів, сторінка товару, можливість входу та реєстрації в системі, перегляд історії замовлень, розділ, присвячений благодійності, а також сторінку з експертами у сфері еко товарів. Крім того, адміністратор повинен мати можливість додавати, редагувати та видаляти товари, користувачів, замовлення та експертів. Також важливо вести моніторинг продажів та користувачів, а також збирати загальну статистику.

Для реалізації цього програмного продукту, необхідно використовувати локальний сервер, наприклад локальний сервер, вбудований у VS Code [2]. У даній кваліфікаційній роботі краще використовувати такі мови програмування як:

- React.js для створення інтерактивних компонентів на веб-сторінках, які дозволяють користувачам взаємодіяти з сайтом [3].
- Next.js для розробки серверної частини програми. Це не тільки забезпечує швидке та ефективне відтворення сторінок, але також підтримує відтворення та маршрутизацію на стороні сервера [4].
- Typescript для підвищення надійності та стабільності коду, щоб допомогти усунути помилки на етапі розробки та полегшити роботу з великими проектами [5].
- Tailwind CSS для швидкого та зручного стилізації компонентів за допомогою попередньо створених класів [6].

На підставі перерахованих переваг і міркувань загального спрощення визначено перелік завдань, які необхідно вирішити для досягнення цього результату і ось що потрібно для реалізації цієї задачі:

- Провести дослідження ринку еко товарів
- Визначити цільову аудиторію сайту
- Визначити конкурентні переваги
- Створити загальну концепцію сайту
- Забезпечити дизайн для сайту
- Розробити прототип сайту
- Реалізувати функціональні можливості
- Забезпечити адаптивний дизайн
- Провести комплексне тестування сайту
- Оптимізувати продуктивність сайту

Для реалізації описаних завдань у вашій системі необхідні такі об'єкти:

1) Каталог еко товарів:

- i) Сортування за категорією
- ii) Пошук за назвою

2) Кошик для покупок і замовлення:

- i) Додати або видалити товари з кошика для покупок
- ii) Оформити замовлення

3) Особистий обліковий запис користувача:

- i) Реєструватися/Вхід в обліковий запис
- ii) Показ історії замовлень

4) Система оглядів і оцінок:

- i) Можливість для користувачів залишати відгуки та оцінки продуктів, які вони вже придбали.
- ii) Відображає середній рейтинг еко продукту на сторінці продукту.

5) Підтримка платіжної системи:

- i) Інтеграція з платіжною системою.

6) Область керування:

					ІПЗ.КР.Б – 121 – 24 – ІЗ	Арк.
						10

- i) Керування каталогом продуктів (додавання, редагування, видалення).
 - ii) Управління замовленнями та базою клієнтів.
 - iii) Управління та керування експертами.
 - iv) Статистика продажів, загальний аналіз даних магазину.
- 7) Мобільна підтримка:
- i) Адаптивний дизайн для всіх мобільних пристроїв.
- 8) Безпека та конфіденційність:
- i) Захист персональних даних користувачів, шифрування персональних і платіжних даних.

1.2 Аналіз аналогічних програмних продуктів

Для порівняльного аналізу було обрано три сайти для продажу еко продукції: КАНЦКАПІТАЛ, ECOVILL, Vita Natura, Glossary, та ФІТОДАР. Аналіз цих веб-сайтів дозволяє нам зрозуміти практики та підходи, які там використовуються, що може позитивно вплинути на розробку програмного забезпечення в майбутньому.

Одним з перших додатків є сайт КАНЦКАПІТАЛ (рис. 1.1).

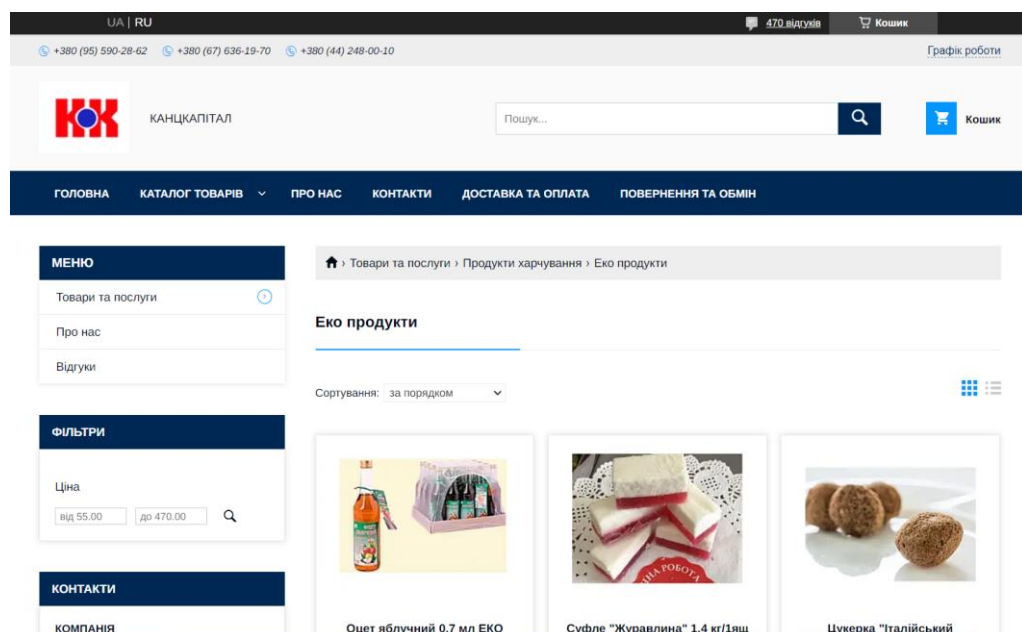


Рис. 1.1 Інтернет магазин з продажу еко товарів КАНЦКАПІТАЛ [7]

Сайт "КАНЦКАПІТАЛ" – це спеціалізований сайт, яка займається

					ІПЗ.КР.Б – 121 – 24 – ІПЗ	Арк.
						11

продажом в більшості домашніх еко товарів для цінувальників цієї сфери. Сайт забезпечує зручний доступ до широкого асортименту домашньої еко продукції.

Основними перевагами КАНЦКАПІТАЛ є:

- велика кількість фільтрів та сортувань;
- інтуїтивно зрозуміла система у використанні;
- прозорі тарифи;
- ексклюзивні пропозиції (асортимент домашньої продукції);

Недоліками є:

- застарілий дизайн;
- немає адаптивності для мобільних пристроїв.
- обмежений асортимент

ECOVILL – інноваційне, якісне рішення для людей, які хочуть зберегти та покращити своє здоров'я. У магазині представлений широкий асортимент екологічно чистих та органічних продуктів, які сприяють збалансованому способу життя. Призначений для тих, хто цінує якість продукції з натуральних компонентів без шкідливих домішок і хімічних добавок (рис. 1.2).

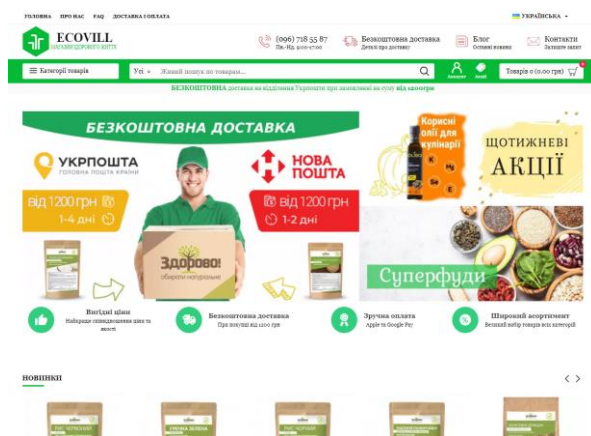


Рис. 1.2 Інтернет магазин з продажу еко товарів ECOVILL [8]

Основними перевагами ECOVILL є:

- велика кількість фільтрів та сортувань;
- можливість вибрати мову сайту;
- зручність оплати;
- знижки на першу оплату;

Недоліками є:

					ПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
						12

- дизайн вже застарілий і не дуже зрозумілий;
- немає адаптивності для мобільних пристроїв.
- недостатня інформація про товар

Сайт «Фітодар» — інтернет-магазин, що спеціалізується на натуральних, здорових продуктах харчування та екологічно чистих товарах для догляду за собою. На сайті представлений широкий асортимент товарів, включаючи органічні продукти харчування, вітаміни та добавки, спортивне харчування, натуральну косметику та засоби гігієни. Крім того, магазин пропонує різноманітні аксесуари для здорового способу життя, зокрема соковижималки, та контейнери для зберігання. Фітодар надає нашим клієнтам продукцію найвищої якості, яка відповідає найвищим стандартам безпеки та сертифікована за міжнародними стандартами. Завдяки зручному інтерфейсу веб-сайту та швидкому процесу замовлення користувачі можуть легко знаходити потрібні товари та доставляти їх прямо додому (рис. 1.3).



Рис. 1.3 Інтернет магазин з продажу еко товарів Фітодар [9]

Основними перевагами Фітодар є:

- великий вибір товарів, не тільки їжа;
- краща якість на ринку з сертифікатами;
- зручність замовлення;
- доставка прямо до дому;
- можливість змінити мову сайту;

Недоліками є:

- неінтуїтивний пошук;

- високі ціни;
- відсутність регулярних акцій та знижок;
- недостатня взаємодія з клієнтами;

Сайт Glossary — є важливим онлайн-ресурсом для тих, хто цікавиться здоровим способом життя та правильним харчуванням. Основною метою цього сайту є надання користувачам доступу до широкого асортименту продуктів харчування, а також інформації про них (рис. 1.4).

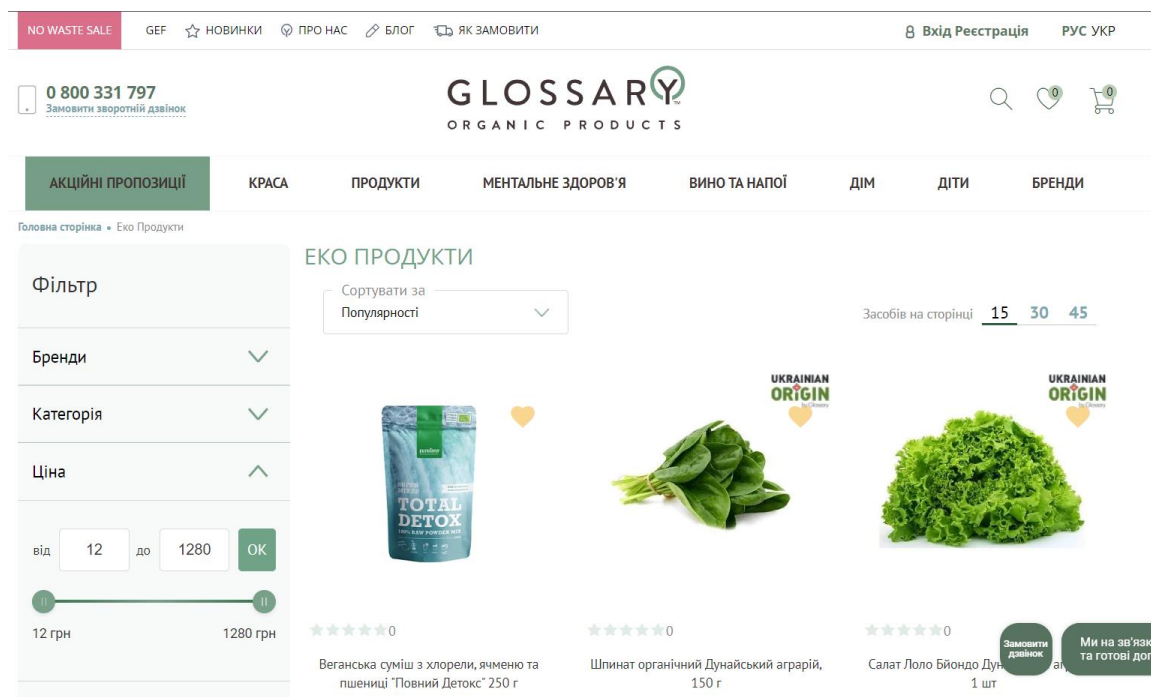


Рис. 1.4 Інтернет магазин з продажу еко товарів Glossary [10]

Основними перевагами Glossary є:

- різноманітність продуктів;
- інформативність;
- хороша взаємодія з клієнтами;
- зручний дизайн;
- велика кількість фільтрів та сортувань;

Недоліками є:

- недостатня інформація про товар;
- високі ціни.
- відсутність регулярних акцій та знижок;

Сайт Vita Natura — інтернет-магазин, що спеціалізується на продажу натуральних та органічних продуктів харчування. Основною метою цього веб-сайту є надання користувачам доступу до широкого асортименту харчових продуктів та інформації про них (рис. 1.5).

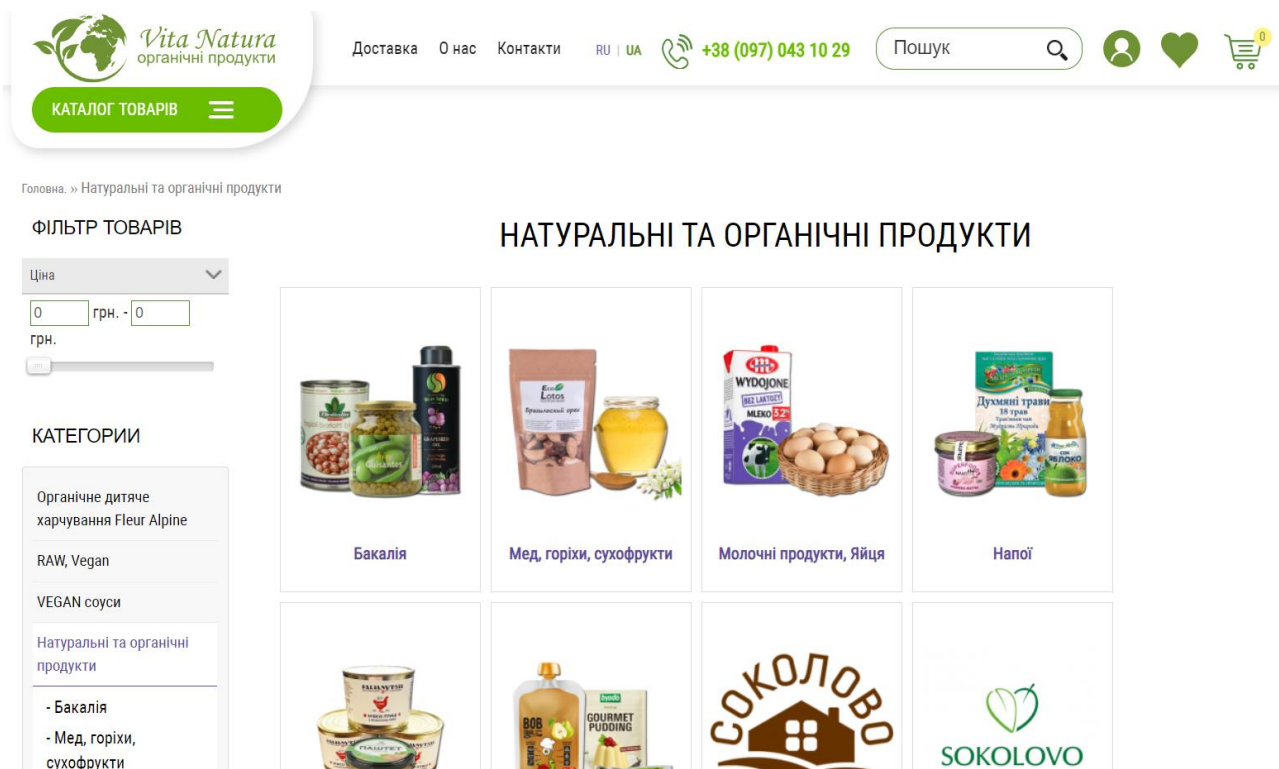


Рис. 1.5 Інтернет магазин з продажу еко товарів Vita Natura [11]

Основними перевагами Vita Natura є:

- велика кількість фільтрів та сортувань;
- наявність чатів з консультантами;
- прозорі тарифи;
- наявність історії покупок;

Недоліками є:

- застарілий дизайн;
- немає адаптивності для мобільних пристроїв.
- реклама та спам;

Порівняння характеристик інтернет-магазинів: КАНЦКАПІТАЛ, Ековіль, Vita Natura, Glossary та ФІТОДАР, а також розроблюваного власного проекту в межах кваліфікаційної роботи (табл. 1.1).

Таблиця 1.1

	Ціна	Функціональність	Інтерфейс	Підтримка	Адаптивність
Kanckapital	Помірні	Основна	Простий	Email, телефон	Мобільна
ECOVILL	Високі	Розширена	Сучасний	Форма, телефон	Повна
Fitodar	Низькі	Основна	Простий	Телефон, email	Мобільна
Vita Natura	Помірні	Основна	Простий	Email, телефон	Адаптована
Glossary	Високі	Розширена	Стильний	Чат, email, телефон	Повна
Додаток в розробці	Помірні	Розширена	Сучасний	Email, телефон	Повна

1.3 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

Під час вибору інструментальних засобів для розробки інтернет-магазину екологічних товарів було враховано низку технічних, функціональних та естетичних аспектів.

Для створення веб-сайту, що базуються на React.js, використовувався Next.js, який забезпечує переваги, такі як підтримка серверного та статичного відтворення, маршрутизація, оптимізація швидкодії та можливість побудови високопродуктивних додатків з великою кількістю взаємодій. React.js, як бібліотека JavaScript для створення користувацьких інтерфейсів, дозволяє створювати компоненти інтерфейсу, якими легко управляти та маніпулювати, що робить його ідеальним для створення інтерактивних елементів магазину [12].

Ключових розширень JavaScript, вибраних для проекту, став TypeScript, який додає строгу типізацію до мови. TypeScript сприяє виявленню помилок під

час розробки на етапі компіляції, забезпечуючи більшу надійність та зручність управління кодом, особливо у великих проектах.

Щоб стилізувати веб-елементи було використано сучасний CSS-фреймворк Tailwind CSS, який пропонує набір готових компонентів та утиліт класів. Завдяки Tailwind CSS розробники можуть швидко створювати стилізовані компоненти без необхідності написання власних CSS-стилів.

Щоб забезпечення простої та ефективної взаємодії з базами даних було застосовано Prisma DB. Prisma автоматизує багато рутинних операцій, таких як створення, читання, оновлення та видалення записів у базі даних [13].

Для прийому онлайн-платежів використовувався Stripe, відомий своєю безпекою та зручністю інтеграції з різноманітними платіжними методами, що робить його ідеальним вибором для інтернет-магазинів [14].

Кожен із вибраних інструментів був обраний з метою забезпечення високої якості, ефективності та надійності інтернет-магазину екологічних товарів, а також для спрощення процесу розробки та підтримки проекту.

Висновки до першого розділу

У першому розділі здійснено комплексний аналіз напрямків застосування інтернет-магазину з продажу екологічних товарів, сформульовано постановку задачі та розглянуто аналоги існуючих програмних продуктів. Зокрема, окреслено основні цілі створення інтернет-магазину.

Аналіз аналогічних програмних продуктів дозволив виявити їхні сильні та слабкі сторони, що стало основою для формування вимог до власного рішення. Визначено ключові функції та особливості, які необхідно реалізувати для забезпечення конкурентоспроможності та задоволення потреб користувачів.

Також були обрані інструментальні засоби для побудови сучасного, швидкого та інтерактивного інтернет-магазину, що забезпечить надійність та зручність управління кодом і дозволить швидко та ефективно стилізувати компоненти. Вибрано засоби для простої та ефективної взаємодії з базою даних

і технологію обробки платежів, яка надасть безпечний та зручний механізм прийому платежів.

Таким чином, обрані технології та інструменти сприятимуть створенню інтернет-магазину, що відповідатиме сучасним вимогам щодо якості, ефективності та надійності, а також сприятиме розвитку ринку екологічних товарів.

РОЗДІЛ 2. ПРОЕКТУВАННЯ САЙТУ З ПРОДАЖУ ЕКО ТОВАРІВ

2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи

Веб-сайт з продажу екологічних товарів надає користувачам можливість переглядати товари, здійснювати покупки, переглядати історію своїх замовлень, залишати відгуки та взаємодіяти з експертами в цій галузі. Функціональні можливості системи залежать від активності користувачів. Користувачі можуть бути класифіковані на три основні групи: звичайні покупці, які шукають екологічні товари; експерти, які надають консультації покупцям, які бажають отримати детальнішу інформацію, та адміністратор, який управляє загальними можливостями сайту. Для візуалізації взаємодії між елементами додатку було розроблено діаграму варіантів використання (рис. 2.1).

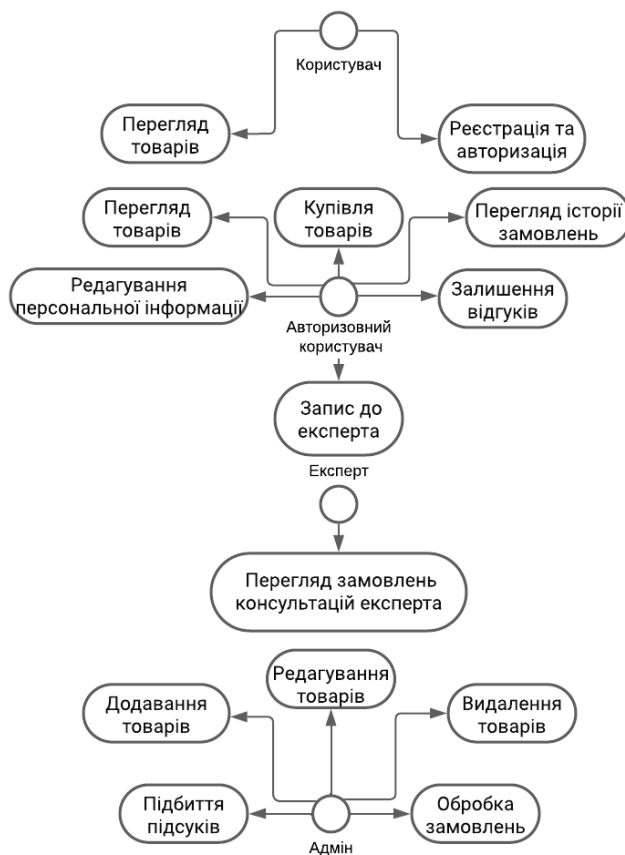


Рис. 2.1 Діаграма варіантів використання

Основна ціль створення сайту з продажу еко-товарів полягає в розробці онлайн-сайту, де можна продавати екологічно чисті товари. Сайт спрямований забезпечити ефективну комунікацію та взаємодію звичайних покупців, експертів та адміністратора і також забезпечити зручний та надійний механізм продажу.

Основна мета сайту, з продажу екологічно чистих продуктів, полягає в просуванні та продажу продуктів, які сприяють збереженню навколишнього середовища, підтримують сталий розвиток і покращують якість життя. Цей сайт має на меті підвищити обізнаність споживачів щодо важливості вибору екологічно чистих продуктів, пропонуючи високоякісні та екологічно чисті альтернативи звичайним продуктам. Підтримуючи виробників, які дотримуються принципів екологічної та соціальної відповідальності, ми заохочуємо свідоме споживання та зменшуємо екологічний слід кожного клієнта. Крім того, це місце також сприяє формуванню спільноти однодумців, зацікавлених у сталому способі життя та збереженні природи для майбутніх поколінь.

Нижче наведено перелік основних вимог до користувачів:

Користувач:

- Перегляд товарів;
- Реєстрація та авторизація;

Авторизований користувач:

- Перегляд товарів;
- Купівля товарів;
- Перегляд історії замовлень;
- Залишення відгуків;
- Редагування персональних даних;

Адмін:

- Підбиття підсумків;
- Обробка замовлень;
- Додавання/видалення товару;
- Редагування товару;

Експерт:

- Перегляд замовлень консультацій експертів;

Функціональні вимоги:

1. Користувачі повинні мати можливість створювати обліковий запис та авторизуватися на сайті.
2. Сайт повинен забезпечувати зручний перегляд, пошук товарів та фільтрація за категоріями.
3. Користувачі повинні мати змогу додавати товари до кошика, редагувати його вміст та оформляти замовлення.
4. Користувачі повинні мати можливість переглядати історію замовлень, відстежувати статус поточних замовлень та отримувати сповіщення про їхній стан.
5. Адміністратори сайту повинні мати інструменти для управління товарами, включаючи додавання, редагування та видалення товарів.
6. Сайт повинен підтримувати функції відгуків та оцінок товарів, дозволяючи користувачам ділитися своїми враженнями про продукцію.
7. Інтерфейс сайту має бути адаптивним, забезпечуючи зручне користування на різних пристроях, включаючи комп'ютери, планшети та смартфони.
8. Користувач повинен мати змогу записатись до експертів в сфері екологічних товарів.
9. Експерт повинен бачити список людей які відправили запит до нього на консультацію для змоги зв'язатися.

Не функціональні вимоги:

1. Продуктивність: Сайт повинен забезпечувати швидке завантаження сторінок та оперативну обробку запитів користувачів, навіть при високих навантаженнях.
2. Надійність: Система повинна працювати стабільно без збоїв, забезпечуючи безперебійний доступ до всіх функцій та даних.
3. Безпека: Сайт повинен мати надійні механізми захисту від несанкціонованого доступу, зломів та інших кібератак. Особисті дані

					ІІЗ.КР.Б – 121 – 24 – ІІЗ	Арк.
						21

користувачів мають бути захищені за допомогою сучасних методів шифрування.

4. Масштабованість: Архітектура сайту повинна дозволяти легке масштабування для обробки зростаючої кількості користувачів та даних без втрати продуктивності.
5. Зручність користування: Інтерфейс сайту повинен бути інтуїтивно зрозумілим і зручним для користувачів, забезпечуючи легкий доступ до всіх функцій та розділів.
6. Сумісність: Сайт має коректно відображатись та функціонувати у різних веб-браузерах та на різних операційних системах.
7. Підтримка мобільних пристроїв: Сайт повинен бути оптимізований для роботи на мобільних пристроях, забезпечуючи коректне відображення та функціональність на смартфонах і планшетах.
8. Підтримка та обслуговування: Система повинна бути спроектована таким чином, щоб спрощувати процес оновлення та обслуговування, мінімізуючи час простою під час проведення технічних робіт.

Системні вимоги:

Вимоги до середовища виконання:

1. Мінімальні апаратні вимоги для клієнтських пристроїв:
 - ГБ оперативної пам'яті.
 - Процесор з тактовою частотою не нижче 1.4 GHz.
 - Доступ до мережі Інтернет.

Вимоги до сервера інформаційної системи:

1. Веб-сайт:
 - Сайт повинен бути інтегрований з Web 3.0.
2. База даних:
 - Використання бази даних, орієнтованої на документи, для забезпечення гнучкого та ефективного зберігання даних.
3. Підтримка Node.js:

- Сервер повинен підтримувати технології Node.js для забезпечення сучасних функцій та можливостей веб сайту [15].

Аналіз функціональних вимог дозволив визначити наступні сутності, необхідні для програмної реалізації додатку. Нижче представлена діаграма класів, яка використовується для його функціонування (рис. 2.2).

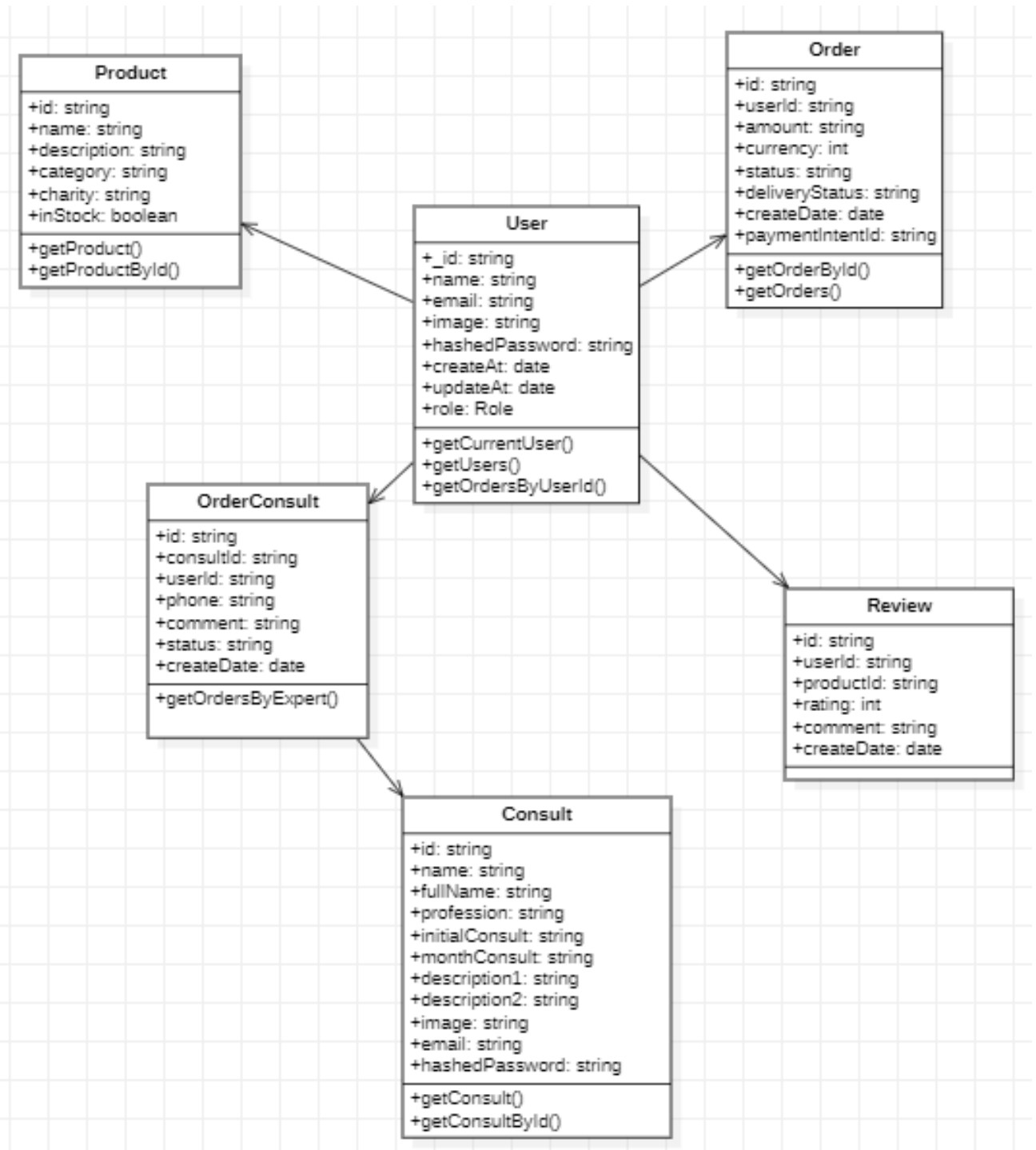


Рис. 2.2 Діаграма класів.

User – клас який відповідає за користувача:

- `getCurrentUser()` – відповідає за отримання поточного користувача.
- `getUsers()` – відповідає за відображення всіх користувачів.

Product – клас який відповідає за товару:

- `getProduct()` – відповідає за отримання продуктів з можливістю фільтрації за категорією та пошуком за назвою або описом.
- `getProductById()` – відповідає за відображення продукту за його ідентифікатором разом з відгуками користувачів.

Order – клас який відповідає за замовлення:

- `getOrders()` – відповідає за отримання всіх замовлень.
- `getOrderById()` – відповідає за отримання замовлення за його ідентифікатором.
- `getOrdersByUserId()` – відповідає за отримання замовлень користувача.

Consult – клас який відповідає за експерта:

- `getConsult()` – відповідає за отримання вибраного експерта.
- `getConsultById()` – відповідає за отримання експерта за її ідентифікатором.

OrderConsult – клас який відповідає за записи до експерта:

- `getOrdersByExpert()` – відповідає за отримання замовлень експертів.

`getGraphData()` – відповідає за отримання даних для графіка, що показує замовлення протягом останніх семи днів, і сумує їхні суми.

2.2 Розробка бази даних системи

Для збереження файлів та інформації у додатку використовується MongoDB, обрано для зберігання файлів та інформації у додатку через її гнучку орієнтовану на документи модель, яка дозволяє ефективно керувати великою кількістю різномірних даних [16].

База даних системи складається з шести колекцій:

- **User** - колекція, яка зберігає дані користувача.
- **Order** – колекція, яка зберігає замовлення.
- **Consult** – колекція, яка зберігає дані експертів.

- OrderConsult – колекція, яка зберігає замовлення до експертів.
- Reviews – колекція для зберігання відгуків.
- Product — колекція, яка зберігає інформацію про всі еко-продукти.

Описи окремих колекцій баз даних можна знайти в таблицях.

Таблиця 2.1

Опис таблиці Users

Users		
Назва поля	Тип поля	Призначення поля
id	string	Унікальний ідентифікатор користувача
name	string	Ім'я користувача
email	string	Електронна пошта користувача
emailVerified	date	Статус верифікації електронної пошти користувача
image	string	URL зображення профілю користувача
hashedPassword	string	Зашифрований пароль користувача
createAt	date	Дата та час створення запису користувача
updateAt	date	Дата та час останнього оновлення запису користувача
role	enum	Роль користувача в системі

Таблиця 2.2

Опис таблиці Order

Order		
id	string	Унікальний ідентифікатор замовлення
userId	string	Ідентифікатор користувача, який здійснив замовлення
amount	float	Сума замовлення
currency	string	Валюта, в якій здійснено оплату
status	string	Статус замовлення
deliveryStatus	string	Статус доставки замовлення
createDate	date	Дата та час створення замовлення
paymentIntentId	string	Ідентифікатор платіжного наміру

Таблиця 2.3

Опис таблиці Consult

Consult		
id	string	Унікальний ідентифікатор експерта
name	string	Ім'я експерта
fullName	string	повне ім'я експерта
profession	string	Професія або спеціалізація експерта

initialConsult	string	Інформація про первинну консультацію
monthConsult	string	Інформація про місячну консультацію
description1	string	Перший опис або деталі про експерта
description2	string	Другий опис або додаткові деталі про експерта
image	string	URL експерта
email	string	Електронна пошта експерта
hashedPassword	string	Зашифрований пароль консультанта

Таблиця 2.4

Опис таблиці OrderConsult

OrderConsult		
id	string	унікальний ідентифікатор замовленої консультації
consultId	string	ідентифікатор консультації
userId	string	ідентифікатор користувача, який замовив консультацію
phone	string	телефон користувача для зв'язку
comment	string	коментар користувача щодо консультації

status	string	статус зв'язку між замовленням та консультацією
createDate	date	дата та час створення цього запису

Таблиця 2.5

Опис таблиці Reviews

Reviews		
id	string	Унікальний ідентифікатор відгуку
userId	string	Ідентифікатор користувача, який залишив відгук
productId	string	Ідентифікатор продукту, до якого відноситься відгук
rating	int	Оцінка продукту
comment	string	Текстовий коментар користувача щодо продукту
createDate	date	Дата та час створення відгуку

Таблиця 2.6

Опис таблиці Product

Product		
id	string	Унікальний ідентифікатор продукту
name	string	Назва продукту
description	string	Опис продукту

category	string	Категорія, до якої належить продукт
charity	string	Інформація про те, з якою благодійністю пов'язаний продукт
inStock	boolean	Наявність продукту

2.3 Проектування та реалізація алгоритмів роботи системи

Сайт є простим у використанні, має всі основні функції сайту, що дозволяє користувачам легко взаємодіяти з ним. Алгоритм роботи з сайту зображено та описано детально всі його пункти (рис. 2.3).

Якщо користувач неавторизований, він може переглядати список доступних еко товарів, що пропонуються на сайті, ознайомлюватися з їх описами, характеристиками, цінами та відгуками інших користувачів. Окрім того, неавторизований користувач має можливість зареєструватися на сайті, створивши свій обліковий запис, або увійти в існуючий аккаунт, що відкриє додаткові функціональні можливості.

Авторизований користувач отримує розширений доступ до функціоналу сайту. Він може не лише переглядати список еко товарів, але й купувати їх, додаючи товари до кошика та оформлюючи замовлення. Крім того, авторизований користувач має доступ до особистого кабінету, де може переглядати історію своїх замовлень, відстежувати статус поточних замовлень, редагувати свої персональні дані, такі як адреса доставки чи контактна інформація. Після успішного отримання товару, користувач може залишати відгуки, допомагаючи іншим покупцям зробити правильний вибір.

Експерт, авторизувавшись на сайті, має доступ до списку консультацій, замовлених у нього. Він може переглядати деталі цих консультацій, зв'язуватися з користувачами, які замовили консультації, для уточнення деталей або проведення самих консультацій. Це дозволяє експертам ефективно керувати своїм часом і надавати якісні консультаційні послуги.

Адміністратор сайту має найбільший обсяг повноважень. Він може переглядати загальну статистику по замовленнях та користувачах, що дозволяє аналізувати активність на сайті та приймати рішення для покращення роботи ресурсу. Адміністратор також обробляє замовлення, забезпечуючи їх своєчасне виконання, додає нові товари до каталогу, редагує або видаляє існуючі. Крім того, адміністратор може створювати акаунти для експертів, забезпечуючи їх доступ до необхідних інструментів для проведення консультацій.

Таким чином, залежно від рівня авторизації, користувачі сайту отримують різні можливості, що дозволяє кожному з них ефективно використовувати ресурси сайту для досягнення своїх цілей, будь то покупки, надання консультацій або адміністрування сайту.

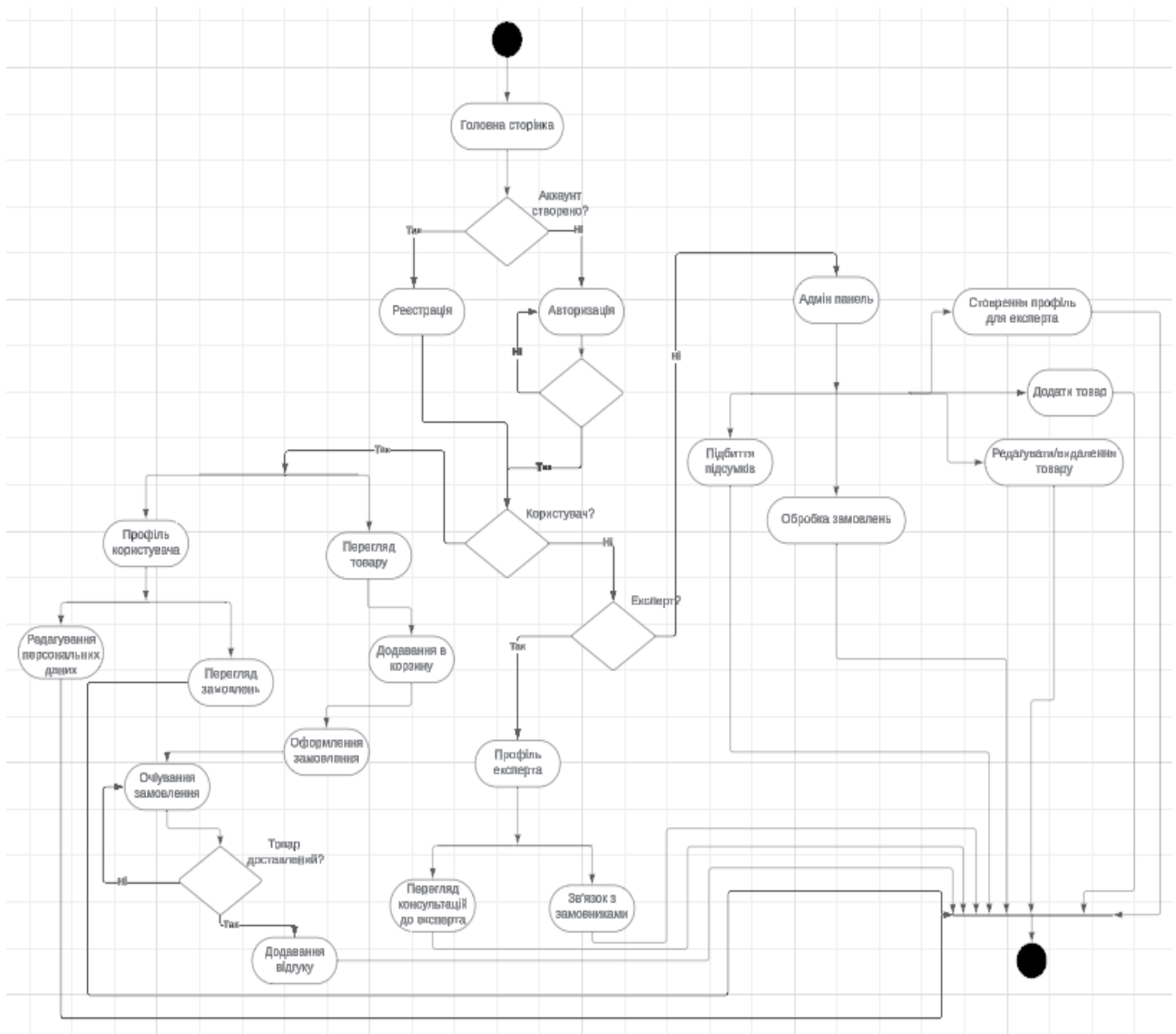


Рис. 2.3 Діаграма активності

2.4 Реалізація функціоналу інтернет магазину еко товарів

Розглянемо основні етапи створення інтернет магазину еко товарів. Для цього користувач повинен здійснити авторизацію або реєстрацію нового аккаунту після чого він має можливість здійснити покупки, переглянути історію замовлень та змінити свої персональні дані і також записатись на консультацію до експерта, нижче буде наведено декілька цікавих функцій. Функція обробки форми авторизації (рис. 2.4).


```

const onSubmit: SubmitHandler<FieldValues> = (data) => {
  setIsLoading(true)
  signIn('credentials', {
    ...data,
    redirect: false
  }).then((callback) => {
    setIsLoading(false)

    if (callback?.ok) {
      router.push("/");
      router.refresh();
    }
    if (callback?.error) {
      console.error("Помилка:", callback.error);
      setErrorMessage("Некоректний email або пароль");
    }
  })
}

```

Рис. 2.4 Обробки форми авторизації

Цей код є функцією обробки форми, яка встановлює стан завантаження, викликає функцію аутентифікації `signIn` з обліковими даними користувача, після отримання відповіді знімає стан завантаження, у випадку успішної аутентифікації перенаправляє користувача на головну сторінку та оновлює її, а у випадку помилки виводить повідомлення про помилку в консоль і встановлює повідомлення про некоректний email або пароль.

Також нижче описано функцію, яка реєструє нового користувача і авторизує його після успішної реєстрації (рис. 2.5).

```

setIsLoading(true);
data.role = "USER";
axios.post("/api/register", data)
  .then(() => {
    signIn("credentials", {
      email: data.email,
      password: data.password,
      redirect: false
    })
  })
  .then((callback) => {
    if (callback?.ok) {
      router.push("/");
      router.refresh();
    }
    if (callback?.error) {
      setError(callback.error);
    }
  })
  .catch((error) => {
    setError(error.message);
  })
  .finally(() => {
    setIsLoading(false);
  });
});

```

Рис. 2.5 Реєстрація нового користувача

Цей код реєструє нового користувача через POST-запит до API і, якщо реєстрація успішна, виконує аутентифікацію користувача за допомогою `signIn`. У разі успішної аутентифікації користувача перенаправляють на головну сторінку і сторінку оновлюють. У разі помилки під час аутентифікації або реєстрації встановлюється повідомлення про помилку. Завершується зняттям стану завантаження, незалежно від результату операцій [17].

Щоб записатись до експерта в даній сфері звичайному користувачу спочатку треба авторизуватись і вибрати експерта до якого є бажання записатись та обговорити всі деталі. Нижче показано створення замовлення консультації до експерта та запис його в базу даних(рис. 2.6).

```

const onSubmit: SubmitHandler<FieldValues> = async (data) => {
  setIsLoading(true);
  if (!data.phone.trim()) {
    setIsLoading(false);
    console.log("Додайте хоч якийсь текст");
    return;
  }

  if (!data.comment.trim()) {
    setIsLoading(false);
    console.log("Додайте хоч якийсь текст");
    return;
  }
  const ratingData = { ...data, userId: user?.id, consult: consult };

  axios.post("/api/appointment", ratingData)
    .then(() => {
      console.log("Записано на консультацію");
      router.refresh();
      reset();
    })
    .catch((error) => {
      console.log("Упс", error);
    })
    .finally(() => {
      setIsLoading(false);
    });
};

```

Рис. 2.6 Запис до експерта на консультацію

Код визначає асинхронну функцію `onSubmit` для обробки форми. Вона встановлює стан завантаження, перевіряє наявність тексту в полях `phone` і `comment`, виводить повідомлення в консоль і завершує роботу, якщо будь-яке поле порожнє. Створюється об'єкт `ratingData` з даними форми, ID користувача та інформацією про консультацію. Потім відправляється POST-запит до API для запису на консультацію. У разі успіху виводиться повідомлення в консоль, оновлюється сторінка і скидається форма [18].

Коли користувач обрав потрібні товари та відповідну кількість, то його спочатку перевіряє на авторизацію, далі якщо він авторизувався, створюється унікальний секретний ключ після натискання на кнопку оформити замовлення для даного платежу, нижче показана функція (рис. 2.7).

```

useEffect(() => {
  if (cartProducts) {
    setLoading(true);
    setError(false);

    fetch('/api/create-payment-intent', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({
        items: cartProducts,
        payment_intent_id: paymentIntent
      })
    }).then((res)=>{
      setLoading(false);
      if (res.status === 401) {
        return router.push('/login');
      }
      return res.json();
    }).then ((data)=>{
      setClientSecret(data.paymentIntent.client_secret);
      handleSetPaymentIntent(data.paymentIntent.id);
    }).catch ((error)=> {
      setError(true);
      console.log("error", error);
    })
  }
}, [cartProducts, paymentIntent]);

```

Рис. 2.7 Створення секретного ключа для платежу та ID платіжного наміру

Цей код виконує певні дії автоматично, коли змінюються дані про товари в кошику або інформація про платіж. Якщо є інформація про товари в кошику, він встановлює стан завантаження для показу індикатора завантаження на екрані і скидає стан помилки. Далі надсилається запит до сервера для створення платіжного наміру, необхідного для проведення платежу. Після отримання відповіді завантаження завершується. Якщо сервер відповідає, що користувач не авторизований (статус 401), користувача перенаправляють на сторінку входу. У разі успішної відповіді від сервера зберігаються секретний ключ для платежу та ID платіжного наміру. Якщо сталася помилка під час запиту, встановлюється стан помилки і виводиться повідомлення про помилку. Таким чином, цей код обробляє процес створення платіжного наміру для товарів у кошику, включаючи обробку можливих помилок і перенаправлення користувача на сторінку входу в разі необхідності.

Більше детально розглянемо функціонал адмін панелі. В ній передбачено створення товарів, експертів, перегляд загальної статистики, редагування та видалення товарів і також оформлення замовлень. Далі розглянемо більш детально декілька цікавих функцій.

Спочатку розглянемо створення в адмін панелі аккаунту для експерта (рис. 2.8).

```
await new Promise<void>((resolve, reject) => {
  uploadTask.on(
    'state_changed',
    (snapshot) => {
      const progress = (snapshot.bytesTransferred / snapshot.totalBytes) * 100;
      console.log('Завантаження ' + progress + '% виконано');
      switch (snapshot.state) {
        case 'paused':
          console.log('Завантаження призупинено');
          break;
        case 'running':
          console.log('Виконується завантаження');
          break;
      }
    },
    (error) => {
      console.log("Помилка завантаження", error);
      reject(error);
    },
    () => {
      getDownloadURL(uploadTask.snapshot.ref)
        .then((downloadURL) => {
          uploadedImages.push(downloadURL);
          console.log('Файл доступний за посилання', downloadURL);
          resolve();
        })
        .catch((error) => {
          console.log('Помилка отримання URL-адреси завантаження', error);
          reject(error);
        });
    }
  );
});
```

Рис. 2.8 Завантаження зображення експерта

Цей код завантажує файл і показує прогрес завантаження, після завершення отримує URL файлу. Якщо виникає помилка під час завантаження або отримання URL, він повідомляє про це. Після успішного завантаження URL додається до списку і показується в консолі (рис. 2.9).

```

    axios.post('/api/consult', consultData).then(()=>{
      setIsConsultCreated(true);
      router.refresh();
    }).catch((error)=>{
      console.log("Помилка щодо збереження бази даних");
    }).finally(()=>{
      reset();
      setIsLoading(false);
    })
    const { password, name, email, role } = data;

    const newData = {
      password,
      name,
      email,
      role,
    };
    newData.role = 'EXPERT';
    axios.post('/api/register', newData)
      .then(() => {
        router.refresh();
      })
      .catch((error) => {
        console.log("Помилка збереження експерта", error);
      })
      .finally(() => {
        reset();
        setIsLoading(false);
      });
  });

```

Рис. 2.9 Створення аккаунта для експерта

Цей код виконує дві операції: створює аккаунт і реєструє експерта. Спочатку він об'єднує дані зображення з іншими даними і надсилає їх на сервер для створення консультації. Після успішного завершення встановлює прапорець про успішне створення, оновлює сторінку, скидає форму і вимикає індикатор завантаження. Якщо виникає помилка, вона фіксується в консолі. Далі він створює, змінює роль на 'EXPERT' і відправляє ці дані на сервер для реєстрації. Після успішної реєстрації оновлюється сторінка, скидається форма і вимикається індикатор завантаження. Помилки, якщо виникають, також фіксуються в консолі.

Далі розглянемо головну сторінку адмін панелі, на неї відображено всі загальні дані про користувачі, про замовлення, оплачені вони чи ні і так далі, нижче показана функція де беруться ці всі дані (рис. 2.10).

```

setSummaryData((prev)=>{
  let tempData = {...prev}

  const totalSale = orders.reduce((acc, item)=>{
    if(item.status == "complete"){
      return acc + item.amount
    }else return acc
  }, 0)
  const paidOrders = orders.filter((order)=>{
    return order.status == "complete"
  })
  const unpaidOrders = orders.filter((order)=>{
    return order.status == "pending"
  })
  tempData.sale.digit = totalSale/100;
  tempData.orders.digit = uniqueOrdersCopy.length;
  tempData.paidOrders.digit = paidOrders.length;
  tempData.unpaidOrders.digit = uniqueOrdersCopy.length-paidOrders.length;
  tempData.products.digit = products.length;
  tempData.users.digit = users.length;

  return tempData
})

```

Рис. 2.10 Підбиття підсумків на головній сторінці адмін панелі

Цей код оновлює підсумкові дані, базуючись на інформації про замовлення, продукти та користувачів. Він розраховує загальну суму продажів для завершених замовлень, кількість завершених та незавершених замовлень, а також загальну кількість замовлень, продуктів та користувачів. Потім ці значення зберігаються у стані summaryData.

Нижче описано ще дві функції для редагування товару, такі як видалення товару, та зміна статусу наявності даного товару(рис. 2.11).

```

const handleToggleStock = useCallback((id: string, inStock: boolean)=>{
  axios.put('/api/product',{
    id,
    inStock: !inStock
  }).then((res)=>{
    router.refresh()
  }).catch((error)=>{
    console.log(error);
  });
},[]);

```

Рис. 2.11 Зміна статусу наявності продукту

Функція handleToggleStock приймає ідентифікатор продукту і поточний статус наявності. Вона надсилає запит на сервер, щоб змінити статус на

протилежний. Після успішного виконання запиту функція оновлює сторінку, щоб відобразити новий статус продукту. Якщо виникає помилка, вона виводиться в консоль (рис. 2.12).

```
const handleDelete = useCallback(async(id: string, images: any[])=>{
  const handleImageDelete = async()=>{
    try{
      for(const item of images){
        if(item.image){
          const imageRef = ref(storage, item.image)
          await deleteObject(imageRef)
          console.log("img deleted");
        }
      }
    }catch(error){
      return console.log(error);
    }
  }
  await handleImageDelete()
  axios.delete(`/api/product/${id}`).then((res)=>{
    router.refresh()
  }).catch((error)=>{
    console.log(error);
  });
},[])
```

Рис. 2.12 Видалення продукту

Та функція `handleDelete` яка приймає ідентифікатор продукту і масив зображень. Вона починає з видалення кожного зображення зі сховища. Для цього вона перевіряє, чи є зображення, і видаляє його. Якщо виникає помилка під час видалення, вона виводиться в консоль. Після видалення зображень функція надсилає запит на сервер для видалення продукту за його ідентифікатором. Після успішного видалення продукту функція оновлює сторінку. Якщо виникає помилка під час видалення продукту, вона також виводиться в консоль.

Висновки до другого розділу

У другому розділі була детально розглянута розробка сайту з продажу еко товарів. Початково були визначені варіанти використання та об'єктно-орієнтована структура системи, що забезпечило чітке розуміння взаємодії між різними компонентами сайту. Потім була розроблена база даних системи, яка гарантує ефективне зберігання та управління інформацією про користувачів,

товари, замовлення та платежі.

На наступному етапі було спроектовано та реалізовано алгоритми роботи системи, що включає обробку замовлень, управління користувачами, розрахунок вартості замовлень, а також інші критичні для функціонування сайту процеси. Завершальною частиною розділу стала реалізація функціоналу інтернет-магазину еко товарів, який надає користувачам можливість переглядати, вибирати та купувати товари, та записуватись до експертів на консультації.

Таким чином, у цьому розділі було створено фундамент для повноцінного функціонування сайту з продажу еко товарів, що забезпечує зручність і ефективність для всіх учасників процесу.

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З САЙТОМ З ПРОДАЖУ ЕКО ТОВАРІВ

3.1 Порядок встановлення та налаштування параметрів сайту

Налаштування сайту вимагає розгортання не тільки всієї системи, а й усіх її компонентів та налаштування окремих елементів. Щоб проілюструвати архітектуру сайту, було створено розгорнуту діаграму (рис. 3.1).



Рис. 3.1 Діаграма розгортання сайту з продажу еко товарів

Опис процесу розгортання:

Клієнтська частина (Frontend): Клієнти взаємодіють із сайтом через веб-браузер. Інтерфейс користувача розроблений за допомогою Next.js 13, React.js, TypeScript та Tailwind CSS [19].

Серверна частина (Backend): Серверна частина використовує API Routes в Next.js 13 для обробки запитів від клієнтів. Сервер виконує бізнес-логіку, обробляє запити на аутентифікацію, замовлення товарів та інші функції [20].

Prisma: Prisma використовується для доступу до бази даних MongoDB. Це ORM, який допомагає взаємодіяти з базою даних більш зручним та типізованим способом [21].

MongoDB: Це реляційна база даних, в якій зберігається інформація про користувачів, товари, замовлення та інші дані, необхідні для функціонування сайту [22].

Stripe: Використовується для обробки платежів. Інтеграція з Stripe дозволяє приймати платежі, керувати підписками та виконувати інші фінансові операції [23].

Ця діаграма показує, як компоненти взаємодіють один з одним та забезпечують функціонування сайту з продажу еко товарів.

3.2 Структура інтерфейсу сайту з продажу еко товарів

У цьому розділі буде представлено огляд сайту, призначеного для продажу екологічних товарів. Аналіз базується на скріншотах інтерфейсу користувача, які демонструють ключові елементи та функціональні можливості програмного продукту. Кожен скріншот супроводжується коротким описом, що підкреслює основні аспекти взаємодії користувачів з сайтом.

Під час першого відкриття сайту представлено головну сторінку сайту, яку користувач бачить при його відкритті на комп'ютері. Ця сторінка містить кілька важливих елементів (рис. 3.2), (див. рис. 3.3).

Головний банер розташований у верхній частині сторінки і виконує інформаційно-рекламу благодійності, привертаючи увагу користувачів до залучення аудиторії.

Меню сайту забезпечує зручну навігацію між різними розділами. У ньому містяться посилання на основні категорії, а також на додаткові сторінки, такі як Головна, Каталог, Експерти та Про нас.

І також кнопка входу/реєстрації розташована у верхньому правому куті сторінки, надаючи користувачам можливість увійти до свого облікового запису або зареєструватися на сайті.



Далі розглянуто каталог товарів є центральним елементом сайту, де користувачі можуть ознайомитися з усіма доступними екологічними товарами. Він представлений у вигляді списку із зображеннями, назвами, коротким описом та цінами, що робить процес вибору зручним та інтуїтивно зрозумілим (рис. 3.4).

На сторінці каталогу також реалізовано функцію пошуку, що дозволяє користувачам швидко знаходити необхідні товари за ключовими словами. Пошуковий рядок розташований у верхній частині сторінки, забезпечуючи швидкий доступ до нього.

Фільтрація по категоріям є ще однією важливою функцією каталогу, що сприяє полегшенню процесу вибору товарів. Це дозволяє користувачам звужити пошук до конкретних груп товарів, що відповідають їхнім потребам та вподобанням.

Загалом, каталог товарів забезпечує зручність та ефективність взаємодії користувачів із сайтом, надаючи їм можливість швидко та легко знаходити та обирати необхідні екологічні товари.

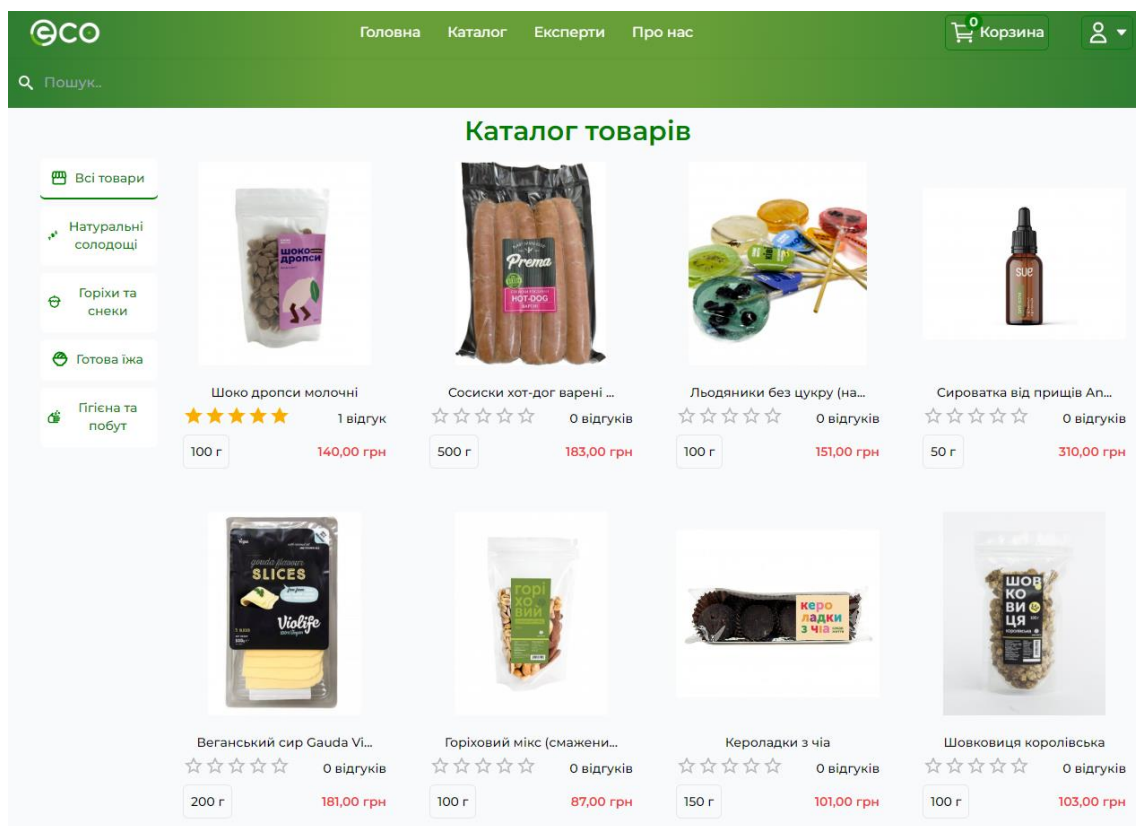


Рис. 3.4 Каталог товарів

Кожен товар має окрему сторінку з детальною інформацією, яка включає опис, характеристики, відгуки користувачів та додаткову інформацію (рис. 3.5).



Рис. 3.5 Сторінка товару

Після того як користувач вибрав кількість товарів та його вагу додавши до корзини користувач може або видалити або переглянути сторінку (рис. 3.6).

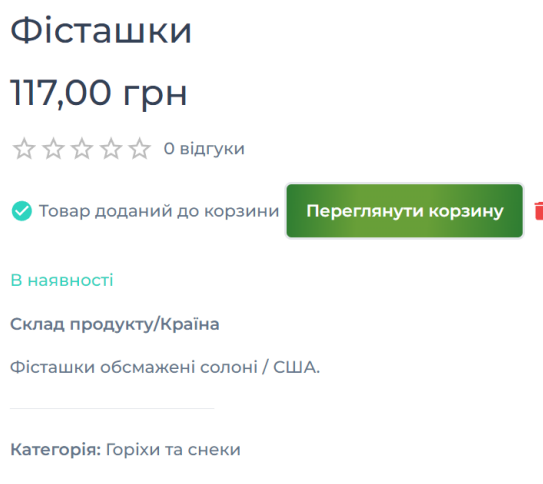


Рис. 3.6 Сторінка товару після додавання товару в корзину

Натиснувши переглянути сторінку користувач попадає на сторінку корзини де він може ще раз перевірити всю інформацію, підправити та перейти до наступного пункту оформлення замовлення, сторінка корзини (рис. 3.7).

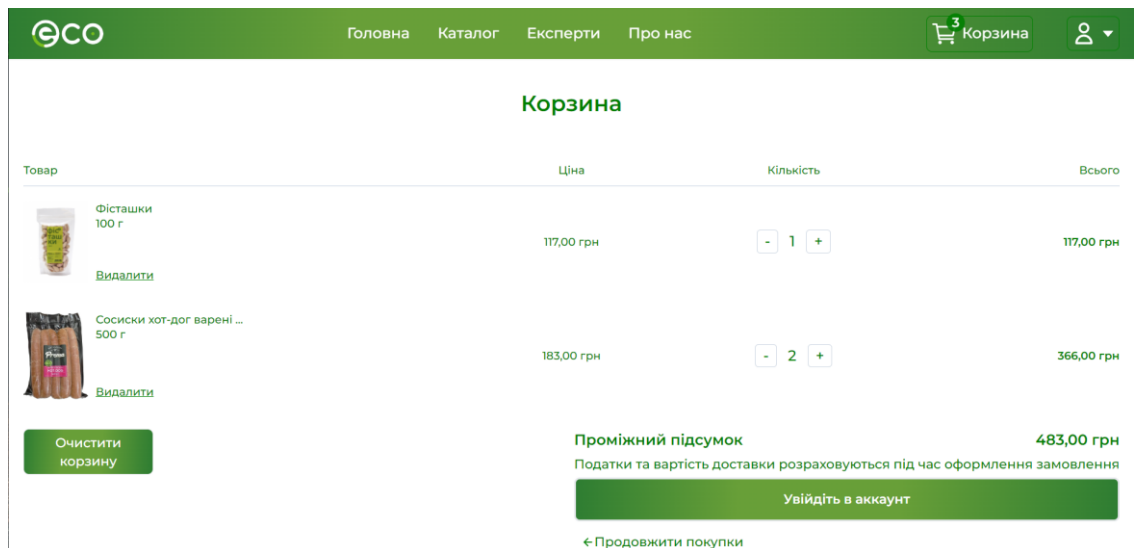


Рис. 3.7 Сторінка корзини

Так як користувач не був ще авторизований його попросить перейти на сторінку авторизації для реєстрації або для входу в аккаунт, після успішної авторизації користувач може повернутись до корзини для подальшого оформлення замовлення. Нижче показані сторінки авторизації та реєстрації (рис. 3.8).

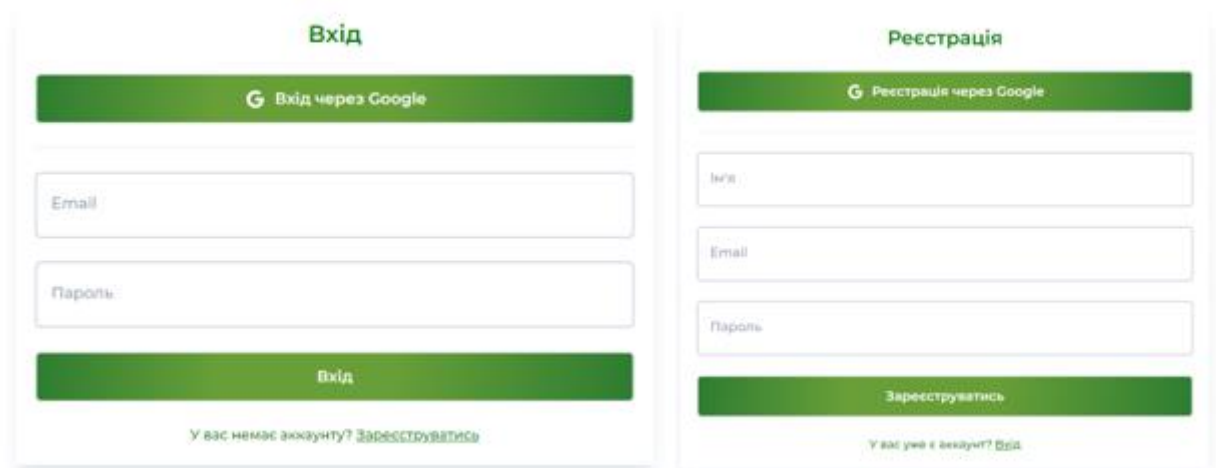


Рис. 3.8 Сторінка реєстрації та авторизації

Нижче показано фінальне оформлення замовлення де замовник вводить свої персональні дані та натискає Оплата (рис. 3.9).

Рис. 3.9 Оформлення замовлення

Якщо всі дані заповнені правильно після натискання на кнопку оплата проходить успішно і користувача перекидає на сторінку успішної оплати або помилки (рис. 3.10).

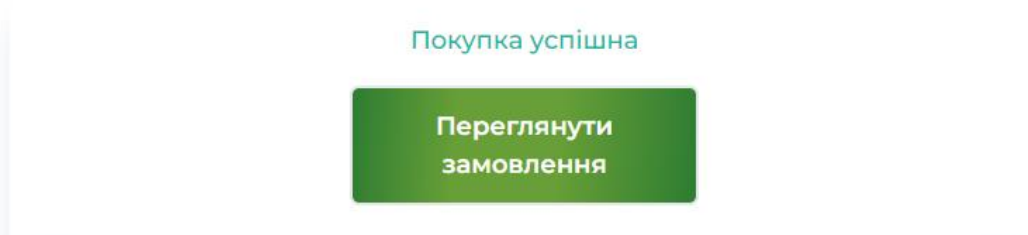


Рис. 3.10 Сторінка успішного замовлення

Після замовлення користувач може переглянути всі свої замовлення, базова інформація, статуси оплати, процес відправки а також може переглянути детальну інформацію натиснувши на додаткові дії сторінка замовлень (рис. 3.11). І також показано більш детальну інформацію нижче (рис. 3.12).

Мої замовлень							
☐	ID	Ім'я замовника	Сума	Статус оплати	Стан замовлення	Дата замовлення	Додаткові дії
☐	6655ab38da6a37f5b1b69...	Богдан Остапчук	483,00 грн	Завершено ✓	В процесі ⌚	14 хвилин тому	
☐	66156ef1da7c71e53be5a...	Богдан Остапчук	87,00 грн	В очікуванні ⌚	Відправлено 📦	2 місяці тому	
☐	661110e0568da026566e0...	Богдан Остапчук	140,00 грн	Завершено ✓	Доставлено ✓	2 місяці тому	
Rows per page: 9 ▾ 1–3 of 3 < >							

Рис. 3.11 Сторінка моїх замовлень

Деталі замовлення

ID замовлення: 6655ab38da6a37f5b1b6930f

Загальна сума: 483,00 грн

Статус оплати: **Завершено** ✓

Статус замовлення: **В очікуванні** ⌚

Дата замовлення: 20 хвилин тому

Замовлені товари

Товар	Ціна	Кількість	Всього
 Фісташки 100 г	117,00 грн	1	117,00 грн
 Сосиски хот-дог варені ... 500 г	183,00 грн	2	366,00 грн

Рис. 3.12 Сторінка детальної інформації

Також користувач може зайти на сторінку Ваші дані де він зможе змінити свої дані, видалити аккаунт, та переглянути невелику статистику про свій аккаунт (рис. 3.13).

Персональні дані

Зміні персональних даних

ipzk211_obv@student.ztu.edu.ua

Богдан Остапчук

Змінити

Видалити аккаунт

Персональні дані замовлень

Кількість замовлень:
6

Сума замовлень:
623,00 грн

Благодійність:
155,75 грн

Завдяки вам ми посадимо 4 🌳.

Рис. 3.13 Сторінка персональних даних

Якщо замовлення вже позначене як доставлено, користувачу відкривається можливість під товаром який замовив залишити відгук, коли користувач вибрав рейтинг, написав коментар і натиснув додати відгук через пару секунд відгук завантажиться на сторінці (рис. 3.14), (див. рис. 3.15).

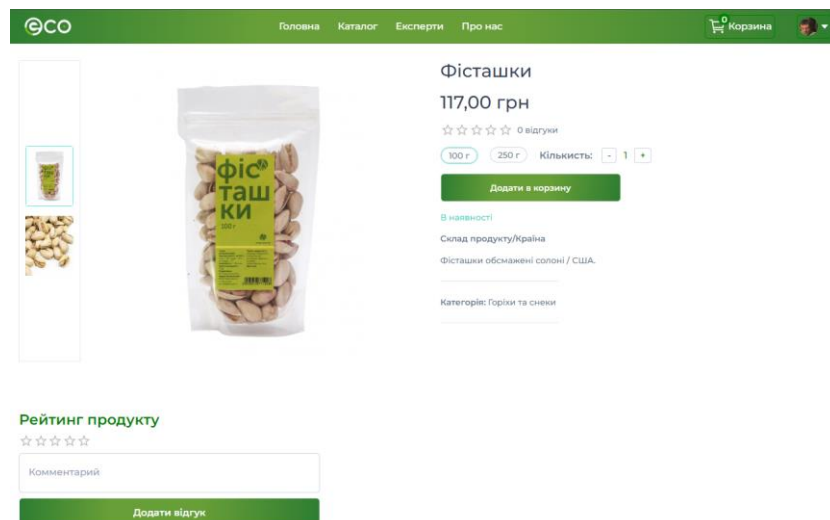


Рис. 3.14 Сторінка з додаванням коментарю до товару

Відгуки



Богдан Остапчук a few seconds ago



Кращі фісташки які я пробував!

Рис. 3.15 Доданий коментар після успішної перевірки

У користувачів сайту є ще одна можливість, а саме запис на консультацію до експертів, щоб замовити консультацію до експерта потрібно вибрати його та під інформацією про експерта залишити свій номер телефона для зв'язку та коментар (рис. 3.16), (див. рис. 3.17).

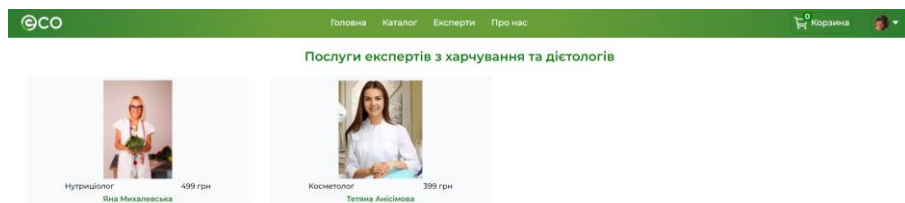


Рис. 3.16 Сторінка з експертами

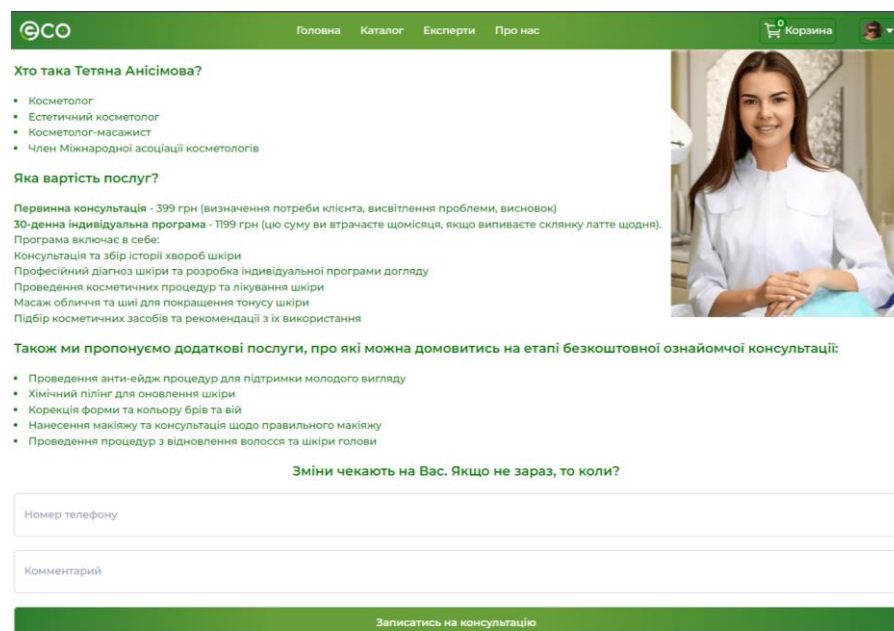


Рис. 3.17 Сторінка експерта з формою для зв'язку

Після успішного запису на консультацію експерту приходить в його так би мовити БД клієнтів і вже може зв'язатись з своїми клієнтами, в кожного експерта відображаються тільки свої замовники (рис. 3.18), (див. рис. 3.19).

Зміни чекають на Вас. Якщо не зараз, то коли?

Номер телефону
+380674107509

Коментарий

Хочу записатись на первинну консультацію, подзвоніть для уточнення всіх деталей, дякую.

Записатись на консультацію

Рис. 3.18 Заповнена форма для зв’язку

Ваші клієнти				
ID	Номер телефона	Коментар	Статус	Дата
662691d8e4fbd1df36486375	380674107509	Потрібна консультація	В очікуванні	22.04.2024, 19:35:36
6627eff95ea85bab658744f1	380674107509	Можна до вас записатись?	В очікуванні	23.04.2024, 20:29:29
6628df645a634d8f422a9ef8	380666666666	test	В очікуванні	24.04.2024, 13:31:00
6628e2475a634d8f422a9efc	380455345345	test test	В очікуванні	24.04.2024, 13:43:19
6655b71fda6a37f5b1b69313	380674107509	Хочу записатись на первинну консультацію, подзвоніть для уточнення всіх деталей, дякую.	В очікуванні	28.05.2024, 13:51:11

Рис. 3.19 БД експерта для зв’язку з клієнтами

Адміністративна панель є ключовим інструментом для управління сайтом, забезпечуючи адміністратора необхідними функціональними можливостями для ефективного керування товарами та користувачами. Вона включає декілька важливих сторінок, що сприяють забезпеченню належного функціонування сайту.

Сторінка зі статистикою містить детальну інформацію про замовлення, статуси замовлень та кількість користувачів. Вона надає адміністраторам можливість відстежувати загальну кількість зроблених замовлень за останні 7 днів, їхній стан (оплачені та неоплачені), що дозволяє контролювати процес обробки замовлень та виявляти можливі затримки або проблеми. Крім того, сторінка надає інформацію про загальну кількість зареєстрованих користувачів (рис. 3.20).

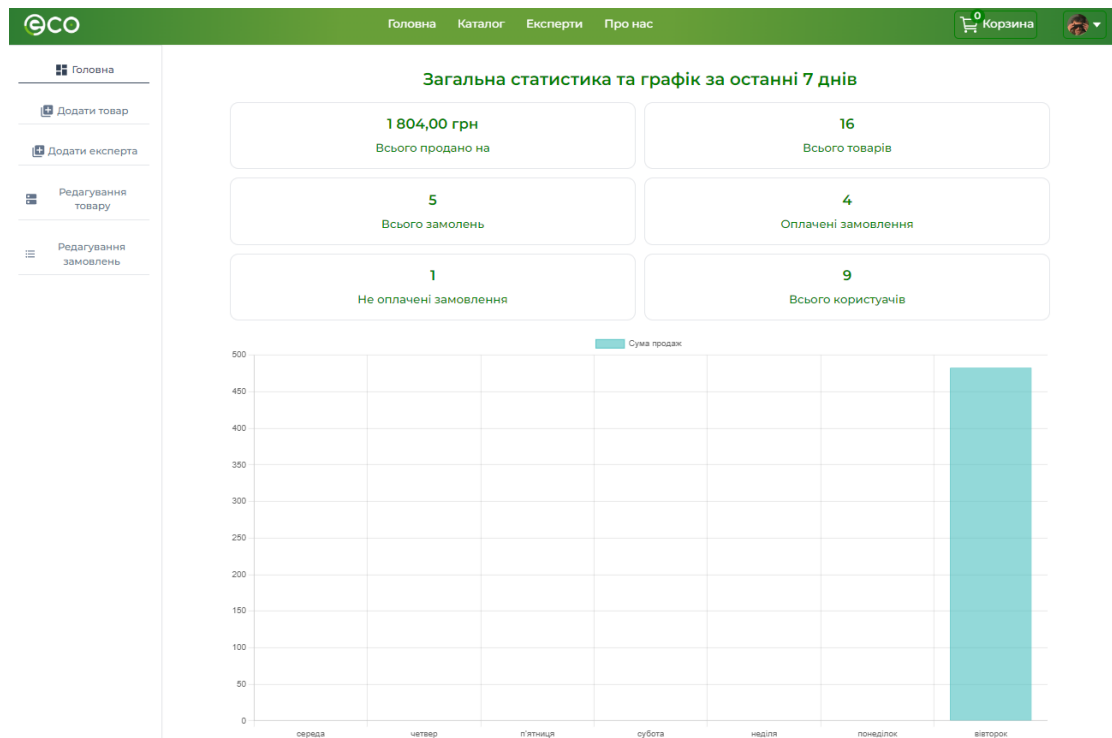


Рис. 3.20 Головна сторінка адміністратора з статистикою

Сторінка додавання товарів дозволяє адміністраторам легко додавати нові товари до каталогу. Вона включає форми для введення назви товару, опису, зображень, цін, кількості на складі та інших характеристик. Адміністратори можуть також визначати категорії, до яких належить товар, і також фонд в який буде йти 25% на благодійність від цього товару.

Сторінка створення профілів для експертів дозволяє адміністраторам створювати та керувати профілями експертів, які можуть надавати консультації користувачам сайту. Вона включає поля для введення особистої інформації експертів, їхніх кваліфікацій, досвіду роботи та цін на консультації. Це забезпечує можливість ефективного управління інформацією про експертів, що сприяє покращенню обслуговування користувачів сайту (рис. 3.21).

клієнтами у випадку необхідності. Адміністратори можуть переглядати деталі кожного замовлення, включаючи товари, що були замовлені, контактну інформацію клієнта та історію змін статусу замовлення (рис. 3.23).

eco

ГоловнаКаталогЕкспертиПро нас

Корзина

Головна

Додати товар

Додати експерта

Редагування товару

Редагування замовлень

Редагування замовлень

☐	ID	Ім'я замовника	Сума	Статус оплати	Стан замовлення	Дата замовлення	Додаткові дії
☐	6655ab38...	Богдан Оста...	483,00 грн	Завершено ✓	Доставлено ✓	годину тому	
☐	6628e282...	qwerty	362,00 грн	Завершено ✓	В процесі ⌚	місяць тому	
☐	6628df8d...	testtest	819,00 грн	Завершено ✓	В процесі ⌚	місяць тому	
☐	66156ef1...	Богдан Оста...	87,00 грн	В очікуванні ⌚	Відправлено ✖	2 місяці тому	
☐	661110e0...	Богдан Оста...	140,00 грн	Завершено ✓	Доставлено ✓	2 місяці тому	

Рис. 3.23 Сторінки редагування товарів

Таким чином, адміністративна панель забезпечує комплексне управління сайтом, включаючи моніторинг статистики, додавання та редагування товарів, управління замовленнями та профілями експертів, що сприяє ефективній роботі та розвитку сайту.

3.3 Тестування сайту

Тестування сайту, призначеного для продажу екологічних товарів, є важливим етапом у забезпеченні його надійності, функціональності та зручності для користувачів. Процес тестування включає кілька основних аспектів, таких як функціональне тестування, тестування продуктивності та зручності використання.

Функціональне тестування перевірено коректність роботи всіх функцій сайту відповідно до заданих вимог. Перевірено завантаження головного банера, робота меню, кошика, та кнопок входу/реєстрації на головній сторінці. Проаналізовано відображення всіх товарів у каталозі, робота пошуку та фільтрації по категоріях, а також коректність відображення товарів відповідно до застосованих фільтрів та ключових слів пошуку. Додатково, перевірено відображення детальної інформації про кожен товар, включаючи

опис, характеристики, зображення, а також можливість додавання товару до кошика. Оцінено функціональність кошика, зокрема можливість додавання та видалення товарів, зміна кількості товарів, відображення загальної вартості замовлення та процес оформлення замовлення. Особлива увага приділено перевірці процесу реєстрації нового користувача, входу до існуючого облікового запису та відновлення паролю. У адміністративній панелі перевірено функціональність додавання, редагування та видалення товарів, управління замовленнями, перегляд статистики та створення профілів для експертів.

Перевірено зручності використання, наскільки зручно та інтуїтивно зрозуміло користувачам взаємодіяти з сайтом. Визначено легкість навігації по сайту, зручність розташування меню, кошика та кнопок входу/реєстрації. Проаналізовано коректність відображення всіх елементів сайту на різних пристроях та браузерях, а також зрозумілість та доступність інформації про товари. Оцінено зручність процесу вибору товарів, додавання їх до кошика та оформлення замовлення.

Проведення комплексного тестування сайту з продажу екологічних товарів є необхідним для забезпечення його стабільної роботи, високого рівня безпеки та користувацького досвіду. Це дозволяє виявити та усунути можливі проблеми на ранніх етапах, забезпечуючи ефективну та безперебійну роботу сайту.

Висновки до третього розділу

У третьому розділі було детально розглянуто інтерфейс та порядок роботи з сайтом з продажу екологічних товарів. Зокрема, було проаналізовано процедуру встановлення та налаштування параметрів сайту, структуру його інтерфейсу, а також проведено комплексне тестування сайту.

У підрозділі, присвяченому порядку встановлення та налаштування параметрів сайту, було описано всі необхідні кроки для правильної інсталяції системи, включаючи вимоги до програмного забезпечення та конфігурацію

серверного оточення. Належна увага була приділена налаштуванню базових параметрів, що забезпечують стабільну та безперебійну роботу сайту.

Аналіз структури інтерфейсу сайту з продажу екологічних товарів показав, що він включає ключові елементи, які сприяють зручності та ефективності використання. Було досліджено головну сторінку, каталог товарів, кошик, сторінку входу/реєстрації, сторінку експертів, а також адміністративну панель. Кожен з цих елементів відіграє важливу роль у забезпеченні позитивного користувацького досвіду.

Комплексне тестування сайту підтвердило її функціональність, надійність та зручність використання. Було проведено функціональне тестування, яке перевірило коректність роботи всіх основних функцій сайту. Тестування зручності використання продемонструвало, що інтерфейс сайту є інтуїтивно зрозумілим та зручним для користувачів.

Таким чином, результати досліджень, проведених у цьому розділі, підтверджують, що сайт з продажу екологічних товарів є добре структурованим, функціонально повним та готовим до експлуатації. Застосовані методи тестування дозволили виявити та усунути можливі проблеми на ранніх етапах, забезпечуючи високу якість та надійність роботи сайту.

ВИСНОВКИ

У цій кваліфікаційній роботі було здійснено всебічний аналіз, проектування та тестування сайту для продажу екологічних товарів, що дозволяє зробити кілька важливих висновків щодо його ефективності, функціональності та доцільності використання.

Перший розділ досліджує основні напрямки застосування магазину з продажу еко товарів. Постановка задачі включала визначення основних цілей та вимог до системи, зокрема забезпечення зручності користувачів та екологічності продуктів. Проведений аналіз аналогічних програмних продуктів дозволив виявити їхні переваги та недоліки, що було враховано при розробці власного рішення. Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення підтвердили доцільність використання сучасних технологій для створення ефективної та стабільної системи.

Другий розділ було присвячено проектуванню сайту з продажу екологічних товарів. Визначення варіантів використання та об'єктно-орієнтованої структури системи забезпечило логічну та ефективну організацію роботи сайту. Розробка бази даних системи забезпечила належне зберігання та обробку інформації. Проектування та реалізація алгоритмів роботи системи дозволили створити функціонал, який задовольняє потреби користувачів і адміністраторів. Реалізація функціоналу інтернет-магазину еко товарів показала його відповідність сучасним вимогам та стандартам.

У Третьому розділі проаналізовано інтерфейс та порядок роботи з сайтом з продажу екологічних товарів. Детально описано порядок встановлення та налаштування параметрів сайту, що забезпечує його стабільну роботу. Проаналізовано структуру інтерфейсу, яка сприяє зручності та ефективності використання сайту. Проведене комплексне тестування сайту підтвердило її функціональність, надійність та зручність використання, а також дозволило виявити та усунути можливі проблеми на ранніх етапах.

Таким чином, результати проведеного дослідження свідчать про успішну розробку, проектування та тестування сайту для продажу

					ІІЗ.КР.Б – 121 – 24 – ІІЗ	Арк.
						57

екологічних товарів. Сайт відповідає всім заявленим вимогам, забезпечує високу якість та надійність роботи, а також сприяє популяризації екологічно чистих продуктів серед користувачів. Реалізація даного проекту має важливе практичне значення та може бути використана як основа для подальших досліджень та розробок у сфері електронної комерції та екологічних ініціатив.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Node JS Documentation [Електронний ресурс] - Режим доступу до ресурсу:
<https://nodejs.org/en>
2. VS Code [Електронний ресурс] - Режим доступу до ресурсу:
<https://code.visualstudio.com/>
3. ReactJS Documentation [Електронний ресурс] - Режим доступу до ресурсу:
<https://reactjs.org/docs/getting-started.html>
4. TypeScript Documentation [Електронний ресурс] - Режим доступу до ресурсу: <https://www.typescriptlang.org/docs/>
5. Tailwind CSS Documentation [Електронний ресурс] - Режим доступу до ресурсу: <https://tailwindcss.com/docs>
6. Kanckapital [Електронний ресурс] - Режим доступу до ресурсу:
<https://kanckapital.com.ua/ua/>
7. Ecovill [Електронний ресурс] - Режим доступу до ресурсу:
<https://ecovill.com.ua/>
8. Ecovill [Електронний ресурс] - Режим доступу до ресурсу:
<https://ecovill.com.ua/>
9. Vita Natura [Електронний ресурс] - Режим доступу до ресурсу:
<https://organichni-producty.com.ua/vita-natura.html>
10. Glossary [Електронний ресурс] - Режим доступу до ресурсу:
<https://glossary.ua/ua/>
11. Fitodar [Електронний ресурс] - Режим доступу до ресурсу:
<https://fitodar.com.ua/uk/>
12. JavaScript Info [Електронний ресурс] - Режим доступу до ресурсу:
<https://javascript.info/>
13. Prisma Documentation [Електронний ресурс] - Режим доступу до ресурсу:
<https://www.prisma.io/docs>
14. Stripe API Documentation [Електронний ресурс] - Режим доступу до ресурсу: <https://stripe.com/docs/api>

15. Node.js server [Електронний ресурс] - Режим доступу до ресурсу:
https://developer.mozilla.org/en-US/docs/Learn/Server-side/Node_server_without_framework
16. Brights [Електронний ресурс] - Режим доступу до ресурсу:
<https://brights.io/ua/technology/mongodb>
17. POST-запит до API [Електронний ресурс] - Режим доступу до ресурсу:
<https://code.tutsplus.com/uk/code-your-first-api-with-nodejs-and-express-understanding-rest-apis--cms-31697t>
18. HTTP у JavaScript [Електронний ресурс] - Режим доступу до ресурсу:
<https://www.freecodecamp.org/ukrainian/news/naypopulyarnishi-sposoby-zrobyty-zapyt-http-u-javascript/>
19. Frontend [Електронний ресурс] - Режим доступу до ресурсу:
<https://itvdn.com/ua/specialities/frontend-developer>
20. Backend [Електронний ресурс] - Режим доступу до ресурсу:
<https://mc.today/что-такое-backend-razrabotka/>
21. Mongodb [Електронний ресурс] - Режим доступу до ресурсу:
<https://www.npmjs.com/package/mongodb>
22. Mongodb Bink [Електронний ресурс] - Режим доступу до ресурсу: <https://www.linkedin.com/company/mongodbinc>
23. Stripe [Електронний ресурс] - Режим доступу до ресурсу:
<https://docs.stripe.com/api>

ДОДАТКИ

					ПЗ.КР.Б – 121 – 24 – ПЗ	Арк.
						61

Реалізація класу Appointment.

Лістинг Appointment.tsx:

```
"use client"
import Button from "@app/components/Button";
import Input from "@app/components/inputs/Input";
import { SafeUser } from "@types";
import { Consult, OrderConsult } from "@prisma/client";
import axios from "axios";
import { useRouter } from "next/navigation";
import React, { useState } from "react";
import { FieldValues, SubmitHandler, useForm } from "react-hook-form";

interface AddAppointment{
  consult: Consult & {
    orderConsult: OrderConsult[]
  } | null;
  user:(SafeUser &{
  }) | null;
}

const Appointment:React.FC<AddAppointment> = ({consult, user}) => {
  const [isLoading, setIsLoading] = useState(false);
  const router = useRouter()

  const { register, handleSubmit, setValue, reset, formState: { errors } } =
    useForm<FieldValues>({
      defaultValues: {
        comment: "",
        phone: "",
        status: "В очікуванні",
      }
    });

  const onSubmit: SubmitHandler<FieldValues> = async (data) => {
    setIsLoading(true);
    if (!data.phone.trim()) {
      setIsLoading(false);
      console.log("Додайте хоч якийсь текст");
      return;
    }

    if (!data.comment.trim()) {
      setIsLoading(false);
      console.log("Додайте хоч якийсь текст");
      return;
    }

    const ratingData = { ...data, userId: user?.id, consult: consult };

    axios.post("/api/appointment", ratingData)
      .then(() => {
        console.log("Записано на консультацію");
      })
  }
}
```

```

        router.refresh();
        reset();
    })
    .catch((error) => {
        console.log("Упс", error);
    })
    .finally(() => {
        setIsLoading(false);
    });
});

if (!user || !consult) return (
    <h1 className="w-full items-center text-center">
        <a href="/login" className="w-full items-center text-center">{isLoading
? "Загрузка" : "Увійти в аккаунт"}</a>
    </h1>
);

return ( <div>
    <div className="mb-5">
        <Input id="phone" type="number" label="Номер телефону"
disabled={isLoading} register={register} errors={errors} required/>
    </div>
    <div className="mb-5">
        <Input id="comment" label="Коментарий" disabled={isLoading}
register={register} errors={errors} required/>
    </div>
    {errors.comment && <span className="text-red-500">Додайте хоч якийсь
текст</span>}
    <Button label={isLoading ? "Загрузка" : "Записатись на консультацію"}
onClick={handleSubmit(onSubmit)}/></div> );
}

export default Appointment;

```

Реалізація базової кнопки Button.

Лістинг Button.tsx:

```

"use client"
import { on } from "events";
import React from "react";
import { IconType } from "react-icons";

interface ButtonProps{
    label: string,
    disabled?: boolean,
    outline?: boolean,
    small?: boolean,
    custom?: string,
    icon?: IconType,

```

```

    onClick: (e: React.MouseEvent<HTMLButtonElement>) => void;
  }
const Button: React.FC<ButtonProps> = ({label, disabled, outline, small, custom, icon:
Icon, onClick}) => {
  return (
    <button onClick={onClick} disabled={disabled} className={`disabled:opacity-70
disabled:cursor-not-allowed
rounded-md
hover:opacity-80
transition
w-full
text-white
flex
iteam-center
justify-center
gap-2
bg-bg
${outline? "bg-white":"bg-bg"}
${outline? "text-slate-700": "text-white"}
${small? "text-sm font-light":"text-md font-semibold"}
${small? "py-1 px-2 border-[1px]":"py-3 px-4 border-2"}
${custom? custom : ''}
`}>{Icon && <Icon size={24}/>}{label}</button> );
  }
}

export default Button;

```

Реалізація створення платіжного наміру.

Лістинг route.ts:

```

import Stripe from "stripe";
import prisma from "@/libs/prismadb";
import { NextResponse } from "next/server";
import { CartProductType } from "@/app/product/[productId]/ProductDetails";
import { getCurrentUser } from "@/actions/getCurrentUser";

const stripe = new Stripe(process.env.STRIPE_SECRET_KEY as string,
  {
    apiVersion: "2023-10-16"
  })

const calculateOrderAmount = (items: CartProductType[]) =>{
  const totalPrice = items.reduce((acc: any, item) => {
    const itemTotal = item.selectImg.price * item.quantity;
    return acc + itemTotal;
  }, 0);

  const price: any = totalPrice.toFixed(2)
  return price;
}

```

					ПЗ.КР.Б – 121 – 24 – ПЗ

```

        amount: total,
        currency: "usd",
        automatic_payment_methods: {enabled: true},
    });

    orderData.paymentIntentId = paymentIntent.id

    await prisma.order.create({
        data: orderData,
    })

    return NextResponse.json({paymentIntent});
}
}

```

Реалізація створення консультанта(експерта).

Лістинг route.ts:

```

import prisma from "@/libs/prismadb";
import { NextResponse } from "next/server";
import { getCurrentUser } from "@/actions/getCurrentUser";

export async function POST(request: Request) {

    const currentUser = await getCurrentUser();

    if(!currentUser){
        return NextResponse.error()
    }

    if(currentUser.role !== 'ADMIN'){
        return NextResponse.error()
    }

    const body = await request.json();
    const {name, fullName, profession, initialConsult, monthConsult, description1,
description2, image, hashedPassword, email} = body;

    const consult = await prisma.consult.create({
        data:{
            name,
            fullName,
            profession,
            initialConsult,
            monthConsult,
            description1,
            description2,
            image,
            hashedPassword,
            email,
        }
    })
}

```

```

    })
    return NextResponse.json(consult);
}

```

Лістинг AddConsult.tsx:

```

"use client"

import Input from "@app/components/inputs/Input";
import TextArea from "@app/components/inputs/TextArea";
import Heading from "@app/components/products/Heading";
import { useCallback, useEffect, useState } from "react";
import { FieldValues, SubmitHandler, useForm } from "react-hook-form";
import { useRouter } from "next/navigation";
import SelectImageConsult from "@app/components/inputs/SelectImageConsult";
import Button from "@app/components/Button";
import { getDownloadURL, getStorage, ref, uploadBytesResumable } from
"firebase/storage";
import firebaseApp from "@libs/firebase";
import axios from "axios";
import { MdPassword } from "react-icons/md";

export type ImageConsultType = {
  image: File | null,
}
export type UploadedImageConsultType = {
  image: string,
}

const AddConsult = () => {
  const [isLoading, setIsLoading] = useState(false);
  const [image, setImage] = useState<ImageConsultType>({ image: null });
  const [isConsultCreated, setIsConsultCreated] = useState(false);
  const router = useRouter()

  const {register, handleSubmit, setValue, watch, reset, formState:{errors}} =
useForm<FieldValues>({
  defaultValues:{
    name: "",
    fullName: "",
    profession: "",
    initialConsult: "",
    monthConsult: "",
    description1: "",
    description2: "",
    image: "",
    email: "",
    password: "",
  }
});

```

```

useEffect(()=>{
  if(isConsultCreated){
    reset();
    setImage({ image: null });
    setIsConsultCreated(false);
  }

}, [isConsultCreated])
const onSubmit: SubmitHandler<FieldValues> = async(data) => {
  setIsLoading(true);
  let uploadedImages: string[] = [];
  if (!image || !image.image) {
    setIsLoading(false);
    console.log("No selected image");
    return;
  }
  console.log("Створюється продукт, зачекайте...");
  try {
    const fileName = new Date().getTime() + "-" + image.image.name;
    const storage = getStorage(firebaseApp);
    const storageRef = ref(storage, `consult/${fileName}`);
    const uploadTask = uploadBytesResumable(storageRef, image.image);

    await new Promise<void>((resolve, reject) => {
      uploadTask.on(
        'state_changed',
        (snapshot) => {
          const progress = (snapshot.bytesTransferred /
snapshot.totalBytes) * 100;
          console.log('Завантаження ' + progress + '% виконано');
          switch (snapshot.state) {
            case 'paused':
              console.log('Завантаження призупинено');
              break;
            case 'running':
              console.log('Виконується завантаження');
              break;
          }
        },
        (error) => {
          console.log("Помилка завантаження", error);
          reject(error);
        },
        () => {
          getDownloadURL(uploadTask.snapshot.ref)
            .then((downloadURL) => {
              uploadedImages.push(downloadURL);
              console.log('Файл доступний за посилання',
downloadURL);

              resolve();
            })
        }
      )
    })
  }
}

```

```

        })
        .catch((error) => {
            console.log('Помилка отримання URL-адреси
завантаження', error);
            reject(error);
        });
    }

    );
});

const consultData = { ...data, image: uploadedImages[0] };

axios.post('/api/consult', consultData).then(()=>{
    setIsConsultCreated(true);
    router.refresh();
}).catch((error)=>{
    console.log("помилка щодо збереження бази даних");
}).finally(()=>{
    reset();
    setIsLoading(false);

})
const { password, name, email, role } = data;

const newData = {
    password,
    name,
    email,
    role,
};
newData.role = 'EXPERT';
axios.post('/api/register', newData)
    .then(() => {
        router.refresh();
    })
    .catch((error) => {
        console.log("Помилка збереження експерта", error);
    })
    .finally(() => {
        reset();
        setIsLoading(false);
    });
} catch(error) {
    setIsLoading(false);
    console.log("Помилка обробки завантаження", error);
    return;
}

};

```

```

return (
  <>
    <Heading title="Додати експерта" center/>
    <Input id="name"
      label="Ім'я англійською" disabled={isLoading} register={register}
errors={errors} required/>
    <Input id="email"
      label="Email" disabled={isLoading} register={register} errors={errors}
required/>
    <Input id="password"
      label="Пароль" disabled={isLoading} register={register} errors={errors}
required/>
    <Input id="fullName"
      label="Повне ім'я" disabled={isLoading} register={register} errors={errors}
required/>
    <TextArea id="profession"
      label="Спеціальність" disabled={isLoading} register={register} errors={errors}
required/>
    <Input id="initialConsult" type="number"
      label="Первинна консультація в грн" disabled={isLoading} register={register}
errors={errors} required/>
    <Input id="monthConsult" type="number"
      label="30-денна індивідуальна програма в грн" disabled={isLoading}
register={register} errors={errors} required/>
    <TextArea id="description1"
      label="Опис індивідуальної програми" disabled={isLoading} register={register}
errors={errors} required/>
    <TextArea id="description2"
      label="Додаткові послуги" disabled={isLoading} register={register}
errors={errors} required/>

    <div>
      <SelectImageConsult item={image} setImage={setImage}
isConsultCreated={isConsultCreated}/>
    </div>

    <Button label={isLoading ? "Завантаження": "Додати експерта"}
onClick={handleSubmit(onSubmit)}/>
  </>
);
export default AddConsult;

```