Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії і управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти перший (бакалаврський)

Веб-застосунок інтернет-магазину побутової техніки

(тема)

Виконав:
здобувач 3 року навчання,
групи КІУКІу-22-2
Даниїл ШУМ
(власне ім'я, прізвище)

Спеціальність
123 «Комп'ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма
Комп'ютерна інженерія
(повна назва освітньої програми)

Керівник: ас. Артем ГУК
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ЕОМ _____ Андрій КОВАЛЕНКО
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти перший (бакалаврський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп’ютерна інженерія
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Шуму Даниїлу Олексійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Веб-застосунок інтернет-магазину побутової техніки

затверджена наказом по університету від “ 26 ” травня 2025 р. № 425 Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 17 червня 2025 р.

3. Вхідні дані до роботи Java
Spring Boot
Spring Security
jQuery
Thymeleaf
Maven
Bootstrap
Hibernate

4. Перелік питань, що потрібно опрацювати у роботі
Вибір та обґрунтування сучасного технологічного стеку
Розробка структури бази даних і побудова ER-схеми
Створення REST API та реалізація логіки на серверній стороні
Проєктування та налаштування реляційної бази даних з використанням MySQL
Розроблення адаптивного та зручного інтерфейсу для користувача
Підключення платіжної системи та реалізація обробки замовлен
Підготовка технічної та супровідної документації

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій цф13 ілюстрацій

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Аналіз проблеми та огляд існуючих рішень	27.05.2025-30.05.2025	
2	Обґрунтування вибору технологічного стеку та	31.05.2025-02.06.2025	
	інструментальних засобів розробки		
3	Проєктування структури бази даних,	03.06.2025-05.06.2025	
	основних сутностей та розробка алгоритмів		
4	Розробка та відлагодження програмного	06.06.2025-09.06.2025	
5	Оформлення матеріалів кваліфікаційної роботи	10.06.2025-11.06.2025	
6	Подання кваліфікаційної роботи керівникові	12.06.2025-13.06.2025	
	та попередній захист		
7	Подання кваліфікаційної роботи на	14.06.2025-16.06.2025	
	рецензування		

Дата видачі завдання " 27 " травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ ас. Гук А. С.
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 62 с., 14 рис., 0 табл., 1 дод., 11 джерел.

ІНТЕРНЕТ-МАГАЗИН, ПОБУТОВА ТЕХНІКА, SPRING BOOT, SPRING SECURITY, JSP/JSTL, BOOTSTRAP, MYSQL, REST API.

Метою кваліфікаційної роботи є розробка веб-застосунку для організації інтернет-магазину з продажу побутової техніки, що забезпечує автоматизовану взаємодію між покупцем і продавцем. У ході виконання роботи розроблено веб-додаток, що дозволяє здійснювати реєстрацію та авторизацію користувачів, перегляд каталогу товарів, пошук і фільтрацію продукції, оформлення замовлень, управління кошиком, перегляд історії покупок.

Функціональність реалізовано на основі Java Spring Boot з використанням шаблонів JSP і бібліотеки JSTL. Для захисту даних і організації доступу до різних функцій застосовано Spring Security. Робота з базою даних реалізована за допомогою ORM-технології JPA/Hibernate у зв’язці з реляційною СУБД MySQL. Інтерфейс користувача створено з використанням CSS-фреймворку Bootstrap, що забезпечує адаптивність і зручність на різних пристроях.

У роботі детально розглянуто архітектуру системи, побудовану за шаблоном MVC, реалізацію бізнес-логіки, проєктування структури бази даних та ER-діаграму. Впроваджено модулі захисту персональних даних, автентифікації користувачів, а також інтеграцію з платіжною системою LiqPay для реалізації онлайн-оплати. Застосунок підтримує розмежування ролей між адміністраторами та звичайними користувачами.

ABSTRACT

Bachelor’s thesis: 62 pages, 14 figures, 0 tables, 1 appendix, 11 sources.

E-COMMERCE, HOUSEHOLD APPLIANCES, WEB APPLICATION, SPRING BOOT, SPRING SECURITY, JSP/JSTL, BOOTSTRAP, MYSQL, REST API.

The aim of this bachelor’s thesis is to develop a modern web application for an online household appliance store, enabling users to browse products, search and filter items, place orders, and manage their personal accounts. The application is built using Java Spring Boot, providing a modular architecture and RESTful API communication between client and server components. Spring Security ensures secure user authentication and data protection, while JPA/Hibernate is used to interact with a relational MySQL database.

The user interface is implemented using JSP, JSTL, and the Bootstrap framework, ensuring responsiveness and user-friendly design across devices. The thesis includes the design of the system architecture based on the MVC pattern, the development of an entity-relationship model, and the implementation of access control and role-based permissions. Integration with the LiqPay payment system has also been implemented for handling online transactions.

The platform supports separate functionalities for administrators and regular users, including a shopping cart, order management, and order history. The developed web application is scalable, secure, and aligned with modern web development practices, making it a viable solution for real-world commercial use and further enhancement.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

API – Application Programming Interface (програмний інтерфейс прикладного програмування)

BCrypt – алгоритм хешування для зберігання паролів

CRUD – Create, Read, Update, Delete (операції створення, читання, оновлення, видалення даних)

DB – Database (база даних)

ER-діаграма – діаграма зв’язків сутностей (Entity-Relationship)

HTML – HyperText Markup Language (мова розмітки гіпертексту)

HTTP – HyperText Transfer Protocol (протокол передачі гіпертексту)

JPA – Java Persistence API (інтерфейс збереження даних у Java)

JSP – JavaServer Pages (серверні сторінки на Java)

JSTL – JavaServer Pages Standard Tag Library (стандартна бібліотека тегів для JSP)

LiqPay – українська платіжна система для онлайн-розрахунків

MVC – Model-View-Controller (архітектурна модель “модель-подання-контролер”)

MySQL – система управління реляційними базами даних

ORM – Object-Relational Mapping (об’єктно-реляційне відображення)

REST API – Representational State Transfer Application Programming Interface (архітектурний стиль та інтерфейс для побудови веб-сервісів)

SQL – Structured Query Language (мова структурованих запитів)

Spring Boot – фреймворк для спрощеної розробки Java-додатків

Spring Security – модуль Spring для забезпечення безпеки веб-застосунків

UI – User Interface (інтерфейс користувача)

UX – User Experience (досвід користувача)

ВСТУП

У сучасну епоху стрімкого розвитку цифрових технологій вторинний ринок товарів набуває дедалі більшого значення в структурі електронної комерції. Підвищення інтересу до вживаних речей пояснюється низкою факторів – економічною доцільністю, екологічною відповідальністю та соціальними змінами в поведінці споживачів. Все більше користувачів надають перевагу свідомому споживанню, що створює попит на онлайн-платформи, які забезпечують прямий обмін товарами між продавцями й покупцями без посередників.

Веб-застосунки, призначені для організації маркетплейсів, є ключовим інструментом цифровізації цього ринку. Завдяки використанню сучасного технологічного стеку, до якого входять Spring Boot, Spring Security, JPA/Hibernate, Thymeleaf, Maven та MySQL, стає можливим створення надійних, масштабованих і безпечних веб-платформ. Такі системи дозволяють повністю автоматизувати процеси реєстрації, розміщення оголошень, обробки замовлень, інтеграції з платіжними сервісами та захисту персональних даних.

Розробка подібних рішень не лише сприяє створенню зручного користувацького середовища, а й підтримує тенденцію до свідомого споживання, що набуває дедалі більшої популярності у суспільстві.

Дана кваліфікаційна робота спрямована на створення функціонального веб-застосунку для маркетплейсу з продажу вживаних товарів, з урахуванням особливостей вторинного ринку. У процесі реалізації проєкту розглядаються всі основні етапи життєвого циклу програмного забезпечення — від дослідження предметної області до побудови архітектури, розробки, тестування та впровадження готового рішення.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА АКТУАЛЬНІСТЬ ПРОЄКТУ

1.1 Ринок побутової техніки: аналіз стану та тенденції розвитку

Ринок побутової техніки посідає важливе місце у структурі споживчого ринку України та світу. Його розвиток визначається низкою економічних, соціальних та технологічних чинників. Побутова техніка охоплює широкий спектр товарів – від великої техніки (холодильники, пральні машини, плити, кондиціонери) до дрібної (пилососи, міксери, кавоварки, електрочайники), і ця номенклатура постійно оновлюється відповідно до запитів споживачів та технологічних інновацій.

Стан ринку на сьогодні характеризується стабільним попитом, зумовленим зростанням урбанізації, підвищенням рівня життя населення та поширенням тенденцій до автоматизації побутових процесів. За даними аналітичних компаній, протягом останніх років спостерігається поступове зростання обсягів продажу побутової техніки, причому найбільший приріст демонструють сегменти дрібної побутової техніки та "розумної" (smart) техніки, оснащеної функціями дистанційного керування, енергозбереження та інтеграції з мобільними пристроями.

Однією з основних тенденцій ринку є активна цифровізація каналів збуту: все більша частина споживачів віддає перевагу онлайн-покупкам побутової техніки. За статистикою, понад 40% продажів цієї категорії в Україні вже припадає на електронну комерцію, і цей показник продовжує зростати щороку (Statista, 2023). Це пояснюється не лише зручністю, а й можливістю порівнювати широкий асортимент, швидко знаходити потрібні моделі, а також користуватися системами знижок та програмами лояльності.

Ще однією характерною рисою сучасного ринку є підвищення вимог споживачів до якості продукції, сервісного обслуговування та наявності розширених гарантій. Значний вплив має також екологічний фактор: попит

на енергоефективну, екологічно безпечну та багатофункціональну техніку поступово зростає, а виробники дедалі частіше впроваджують технології, що відповідають принципам сталого розвитку.

У структурі ринку спостерігається зростання конкуренції як серед глобальних брендів (Samsung, LG, Bosch, Electrolux тощо), так і серед локальних виробників. Водночас споживачі все більше орієнтуються не лише на бренд, а й на поєднання ціни, якості, функціональності та сервісної підтримки.

У перспективі, розвиток ринку побутової техніки визначатиметься подальшою цифровізацією, інтеграцією інтернету речей (IoT), поширенням smart-пристроїв і зростанням вимог до адаптації продуктів під індивідуальні потреби користувачів. В умовах швидкої трансформації ринку компанії, що впроваджують сучасні IT-рішення, мають значну конкурентну перевагу.

1.2 Недоліки та слабкі місця популярних онлайн-магазинів

Попри швидкий розвиток електронної комерції в Україні, більшість популярних онлайн-магазинів побутової техніки стикаються з низкою недоліків і проблем, що впливають на якість обслуговування клієнтів, знижують конкурентоспроможність та формують негативний споживчий досвід.

Однією з основних проблем є недостатня зручність користувацького інтерфейсу. Багато сайтів мають складну структуру навігації, перевантажені головні сторінки або незрозуміле розташування основних функцій. Це ускладнює пошук необхідного товару, оформлення замовлення, порівняння характеристик чи перегляд історії покупок. Особливо відчутними ці недоліки стають для користувачів мобільних пристроїв, оскільки не всі онлайн-магазини оптимізовані для смартфонів та планшетів.

Ще однією поширеною проблемою є обмежена функціональність пошуку та фільтрації. Деякі онлайн-магазини не надають достатньо гнучких

інструментів для підбору товарів за різними параметрами (ціна, виробник, технічні характеристики), що ускладнює процес вибору й знижує лояльність користувачів.

Значний вплив на якість роботи мають також затримки та помилки у відображенні актуальної інформації про наявність товарів і ціни. Через недостатню інтеграцію із складськими системами на сайтах часто трапляються ситуації, коли товар, зазначений як доступний, фактично відсутній на складі. Це призводить до зриву замовлень, необхідності повернення коштів і негативних відгуків.

Проблеми із сервісною підтримкою та комунікацією з клієнтами також залишаються актуальними. Довгі терміни обробки звернень, відсутність швидкої онлайн-консультації чи неоперативне вирішення питань щодо гарантійного обслуговування призводять до втрати довіри та переходу клієнтів до конкурентів.

Ще одним суттєвим недоліком є складність процесу оплати й доставки. Не всі сайти мають зручні платіжні інструменти чи забезпечують прозорі умови доставки, що створює додаткові бар’єри для завершення покупки.

Окремо варто відзначити питання захисту персональних даних і безпеки транзакцій. У ряді випадків онлайн-магазини не впроваджують сучасних механізмів шифрування та автентифікації, що підвищує ризики шахрайства та витоку інформації.

Таким чином, навіть найпопулярніші онлайн-магазини побутової техніки мають цілу низку слабких місць, які суттєво впливають на задоволеність користувачів та ефективність бізнесу. Усунення цих недоліків є важливою передумовою для успішного розвитку нових веб-застосунків і підвищення конкурентоспроможності на ринку електронної комерції.

1.3 Огляд сайтів з продажу побутової техніки

В інтернеті безліч онлайн магазинів. Для аналізу було обрано декілько

магазинів побутової техніки, так як ROZETKA та FOXTROT .Я обрав саме ці магазини тому що перший – інтернет рітейлор без офлайн магазину і сайт це основна його частина. А FOXTROT навпаки – офлайн магазин де сайт це додаткова підтримка. Почну з магазина ROZETKA.

ROZETKA – чудовий приклад інтернет магазина-рітейлера та маркетплейсу. Магазин працює як торгівельна площадка для постачальників. Перейшовши на сайт у розділ побутової техніки перше що бачить користувач це слайдер з інформацією про знижки та акції. А також перелік популярних серед інших покупців категорій. (рисунок 1.1)

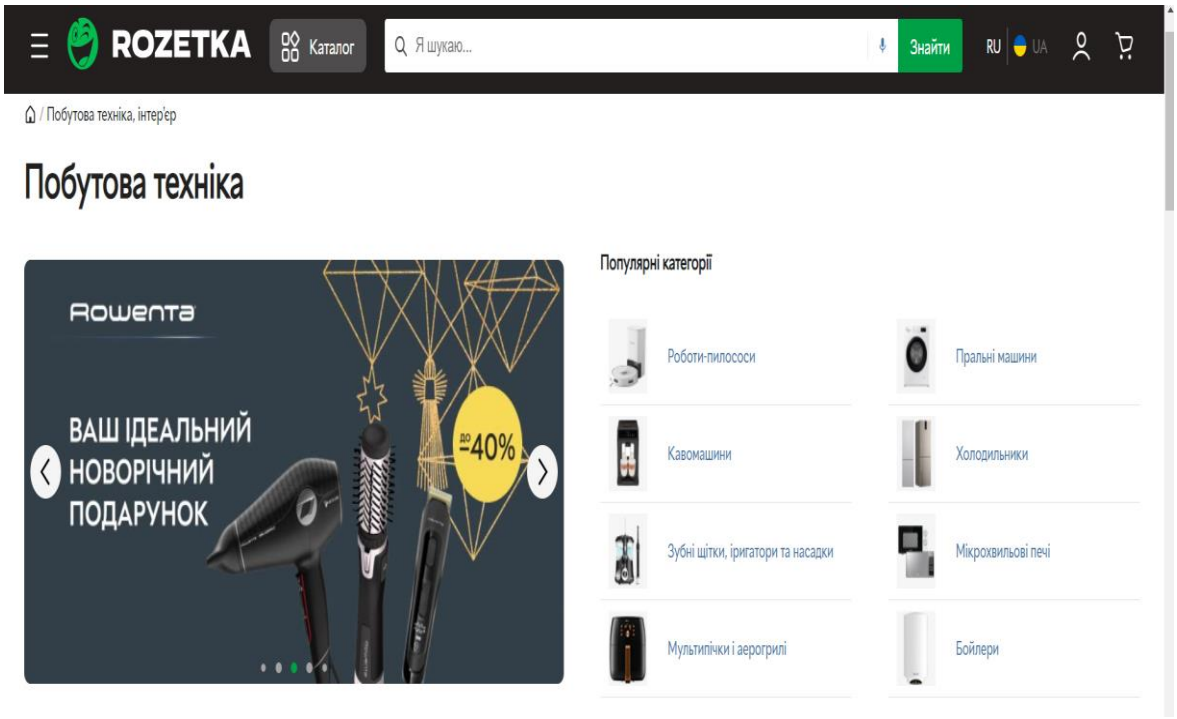

Рисунок 1.1 – головна сторінка розділу

2

Також тут є зручне поле для пошуку необхідних товарів та кнопка «Каталог». У каталозі перелік усіх категорій товарів побутової техніки. Я вважаю що реалізація каталогу тут одночасно як і зручна, так і не дуже. Так як магазин пропонує дуже великий вибір товарів, потрібно на одному єкрані розмістити багато інформації. Тобто потрібно зменщити або розмір шрифту, або робити категорії більш загальними. У ROZETKA це виглядаэ ось так(рисунок 1.2).

1.4 Обґрунтування необхідності створення нового веб-застосунку

Сучасний ринок побутової техніки, попри велику кількість онлайн-магазинів, все ще характеризується низкою проблем, які безпосередньо впливають на досвід користувачів та ефективність електронної комерції. З огляду на результати попереднього аналізу можна зробити висновок, що існуючі онлайн-платформи не завжди відповідають сучасним вимогам до зручності, функціональності, швидкості та безпеки.

По-перше, актуальність створення нового веб-застосунку зумовлена зростанням попиту на прості та інтуїтивно зрозумілі інтерфейси. Багато існуючих сайтів не оптимізовані для мобільних пристроїв, мають заплутану навігацію або незручні форми замовлення, що створює бар’єри для користувачів різного віку та рівня цифрової грамотності.

По-друге, користувачі очікують розширених можливостей персоналізації, швидкого пошуку товарів, гнучкої системи фільтрів і якісної взаємодії з сервісною підтримкою. Більшість поточних рішень на ринку не забезпечують комплексного вирішення цих завдань або реалізують їх лише частково[1].

По-третє, із зростанням обсягів онлайн-покупок надзвичайно актуальними стають питання безпеки персональних даних і захисту транзакцій. Часто вразливі місця в реалізації захисту призводять до зловживань або втрати довіри з боку клієнтів. Саме тому впровадження сучасних технологій аутентифікації, валідації даних і захисту інформації є одним із ключових пріоритетів для нового веб-застосунку.

Крім того, стрімке зростання конкуренції вимагає від бізнесу впровадження додаткових сервісів – наприклад, системи відгуків і рейтингів, онлайн-консультацій, швидкої доставки, гнучких варіантів оплати та багатомовної підтримки. Це дозволяє не лише залучати нових клієнтів, а й утримувати лояльність існуючих.

Отже, розробка нового веб-застосунку для інтернет-магазину побутової техніки є обґрунтованою з точки зору сучасних ринкових вимог, технологічних тенденцій та очікувань споживачів. Такий підхід дозволить не лише усунути основні недоліки існуючих рішень, а й підвищити рівень конкурентоспроможності, забезпечивши якісний сервіс та позитивний користувацький досвід.

1.5 Мета та задачі дипломного проєкту

Мета дипломного проєкту полягає у розробці сучасного, зручного та безпечного веб-додатку для інтернет-магазину побутової техніки, який забезпечить ефективну взаємодію між покупцями та продавцями, розширить можливості електронної комерції та врахує актуальні вимоги ринку.

Для досягнення поставленої мети визначено такі основні задачі дослідження та розробки:

Провести аналіз сучасного стану ринку побутової техніки та визначити особливості організації онлайн-продажів цієї категорії товарів;

Дослідити недоліки та проблеми функціонування існуючих інтернет-магазинів і виявити ключові чинники, що впливають на якість користувацького досвіду;

Обґрунтувати вибір технологічного стеку для створення веб-додатку, з урахуванням вимог до масштабованості, безпеки, зручності та швидкості розробки;

Спроєктувати архітектуру системи, розробити структуру бази даних і описати модель взаємодії основних компонентів;

Реалізувати клієнтську та серверну частини веб-додатку із застосуванням обраних технологій (Spring Boot, JSP/JSTL, Bootstrap, MySQL);

Забезпечити реалізацію основних функцій інтернет-магазину: реєстрація користувачів, перегляд і пошук товарів, керування кошиком і

замовленнями, підтримка онлайн-оплати, зберігання історії покупок;

Інтегрувати систему захисту даних, механізми валідації та безпечної аутентифікації користувачів;

Провести тестування, аналіз працездатності та оцінку якості розробленого застосунку;

Надати рекомендації щодо подальшого розвитку веб-додатку та його масштабування.

Таким чином, дипломний проєкт спрямований на створення комплексного рішення, яке відповідатиме сучасним вимогам ринку й забезпечить високий рівень задоволеності користувачів.

2 АНАЛІЗ ТА ОБҐРУНТУВАННЯ ВИБОРУ ТЕХНОЛОГІЙ

2.1 Аналіз технологій Front-end розробки

Front-end розробка відіграє ключову роль у створенні сучасних веб-додатків, оскільки саме від якості клієнтської частини залежить перше враження користувача, зручність взаємодії та загальний рівень задоволеності сервісом. Сьогодні ринок веб-розробки пропонує широкий спектр технологій та інструментів для створення ефективних, адаптивних і функціональних інтерфейсів[2].

У традиційних Java-проєктах для генерації динамічних сторінок часто застосовують JavaServer Pages (JSP) у поєднанні з бібліотекою JSTL. Таке рішення дозволяє формувати HTML-розмітку на сервері, динамічно підставляти дані та реалізовувати базову інтерактивність через стандартні теги та просту бізнес-логіку. JSP/JSTL залишається актуальним вибором для багатьох корпоративних рішень завдяки простоті інтеграції зі Spring MVC, невисоким вимогам до ресурсів та невеликому порогу входу для розробників.

Водночас стрімкий розвиток веб-технологій призвів до появи потужних клієнтських фреймворків, таких як React, Angular, Vue.js, що дозволяють створювати SPA (Single Page Application) з багатою інтерактивністю та асинхронним обміном даними через REST API. Їхня головна перевага – висока продуктивність, можливість реалізувати складні користувацькі сценарії та створити дійсно сучасний UI. Однак використання таких фреймворків вимагає окремої інфраструктури для підтримки front-end і додаткових витрат на навчання й налаштування проєкту.

Серед серверних шаблонізаторів, окрім JSP, слід виділити Thymeleaf, який набуває популярності завдяки простоті розробки, натуральному синтаксису HTML та гнучкості налаштувань. Thymeleaf добре інтегрується зі Spring Boot, підтримує сучасні підходи до верстання й полегшує командну роботу над інтерфейсом.

Окрему роль у формуванні вигляду та адаптивності веб-додатків відіграють CSS-фреймворки. Зокрема, Bootstrap – один із найпоширеніших інструментів для швидкого створення адаптивних, візуально привабливих інтерфейсів. Його готові компоненти та шаблони значно пришвидшують верстку й забезпечують кросбраузерність і коректне відображення на різних пристроях.

Таким чином, сучасна front-end розробка базується на поєднанні серверних та клієнтських технологій, що дає змогу обирати оптимальне рішення для кожного проєкту залежно від цілей, ресурсів і очікуваної аудиторії. Для реалізації інтернет-магазину побутової техніки доцільно використовувати перевірені технології JSP/JSTL у зв'язці з Bootstrap, що дозволяє досягти балансу між швидкістю розробки, якістю інтерфейсу та легкістю підтримки.

2.1.1 Порівняння JSP/JSTL з альтернативними технологіями

У сучасній веб-розробці для реалізації динамічного відображення даних та побудови інтерфейсу користувача застосовується широкий спектр технологій. В контексті Java-проєктів традиційними рішеннями є JavaServer Pages (JSP) у поєднанні з бібліотекою JavaServer Pages Standard Tag Library (JSTL). Проте на ринку також широко представлені альтернативні підходи, зокрема, сучасні фронтенд-фреймворки та інші серверні шаблонізатори.

JSP/JSTL дозволяють інтегрувати Java-код із HTML-сторінками, а JSTL спрощує відображення інформації завдяки готовим тегам для ітерації, умов, форматування тощо. Таке рішення традиційно використовується у багатьох корпоративних та навчальних проєктах завдяки простоті інтеграції з екосистемою Java EE та підтримці з боку Spring MVC. Перевагою JSP/JSTL є невеликий поріг входу для розробників, можливість швидко створювати прототипи сторінок та відсутність необхідності додаткового налаштування для роботи у звичних середовищах Java.

Однак поряд із JSP/JSTL активно використовуються й інші підходи:

Thymeleaf – сучасний серверний шаблонізатор, який набув популярності завдяки гнучкості, підтримці натурального HTML і кращій інтеграції зі Spring Boot. Thymeleaf забезпечує просте читання шаблонів навіть у вихідному HTML-редакторі та підтримує складніші сценарії динамічного формування сторінок.

React, Angular, Vue.js – це потужні клієнтські фреймворки, які дозволяють створювати SPA (Single Page Application) із використанням RESTful API для обміну даними з сервером. Вони забезпечують високу інтерактивність, миттєве оновлення контенту без повного перезавантаження сторінки та багатий набір UI-компонентів.

Freemarker, Velocity – альтернативні серверні шаблонізатори, що також застосовуються у Java-проєктах. Вони надають розвинуті механізми для створення гнучких шаблонів, однак поступаються за поширеністю JSP та Thymeleaf.

Головні переваги JSP/JSTL – простота та класична інтеграція у Java-проєктах. Недоліками є застаріла концепція в порівнянні з новими фреймворками, складність підтримки великих проектів та обмеженість у створенні сучасних інтерактивних інтерфейсів. У той час як React, Angular чи Vue.js дозволяють реалізовувати інноваційні рішення, їх впровадження вимагає значно більшої кваліфікації та часу на налаштування взаємодії із серверною частиною.

Враховуючи цілі та ресурси проєкту інтернет-магазину побутової техніки, вибір на користь JSP/JSTL є доцільним для забезпечення швидкої реалізації, підтримки класичної архітектури MVC та ефективної інтеграції з технологіями Java EE і Spring Boot. Однак у перспективі розширення функціоналу або інтеграції з мобільними чи SPA-клієнтами можливий перехід до сучасних клієнтських фреймворків або шаблонізаторів нового покоління.

2.1.2 Використання фреймворку Bootstrap для адаптивного інтерфейсу

Однією з ключових вимог до сучасних веб-додатків є забезпечення адаптивності інтерфейсу, тобто його коректного відображення та функціонування на пристроях із різними розмірами екранів – від настільних комп’ютерів до смартфонів. Саме для вирішення цього завдання у веб-розробці широко застосовується фреймворк Bootstrap, який є одним із найпопулярніших інструментів для створення адаптивних і привабливих інтерфейсів.

Bootstrap – це open-source CSS-фреймворк, що містить набір готових стилів, компонентів та інструментів для побудови сучасних веб-інтерфейсів. Він базується на гнучкій системі сіток (grid system), яка дозволяє зручно компонувати елементи сторінки та автоматично підлаштовувати їх до різних розмірів екранів. Завдяки використанню медіа-запитів (media queries) Bootstrap забезпечує коректне масштабування елементів, зміну розташування блоків та оптимізацію навігації для мобільних пристроїв.

Використання Bootstrap у проєкті інтернет-магазину побутової техніки має низку переваг:

Швидкість розробки. Завдяки готовим стилям, класам та компонентам (форми, кнопки, панелі навігації, таблиці тощо) значно скорочується час на верстку сторінок і їх подальшу підтримку.

Єдність стилю. Фреймворк дозволяє підтримувати уніфікований зовнішній вигляд усіх сторінок сайту, що позитивно впливає на сприйняття бренду та створює відчуття цілісності продукту.

Адаптивність. Система сіток і медіа-запитів забезпечують коректну роботу інтерфейсу на будь-яких пристроях без необхідності додаткового налаштування.

Широка підтримка та розширюваність. Завдяки активній спільноті та

великій кількості доступних розширень, Bootstrap легко інтегрується з іншими бібліотеками та дозволяє швидко додавати нові елементи до проєкту.

В контексті даного дипломного проєкту використання Bootstrap є доцільним вибором для створення привабливого, адаптивного та функціонального інтерфейсу інтернет-магазину побутової техніки. Це дозволяє забезпечити позитивний досвід користувача та відповідати сучасним стандартам веб-розробки.

2.2 Аналіз технологій Back-end розробки

Back-end розробка є основою для побудови функціональної та надійної серверної частини веб-додатку. Саме цей рівень відповідає за обробку бізнес-логіки, взаємодію з базою даних, реалізацію авторизації, забезпечення безпеки, обробку запитів від клієнтської частини та формування відповідей. Вибір технологій для back-end напряму впливає на масштабованість, продуктивність і надійність всієї системи.

Для створення сучасних веб-додатків одним із найпоширеніших підходів є використання екосистеми Java та, зокрема, фреймворку Spring Boot. Spring Boot забезпечує швидкий старт розробки за рахунок автоматичної конфігурації та великої кількості готових модулів, які охоплюють всі аспекти back-end: від побудови REST API до організації рівня доступу до даних.

Класичним архітектурним підходом для організації серверної частини є шаблон MVC (Model-View-Controller). Spring MVC, як частина Spring Framework, дозволяє чітко розмежувати логіку обробки запитів (контролери), бізнес-логіку (сервіси) та доступ до даних (репозиторії). Така архітектура спрощує підтримку і розширення проєкту, полегшує тестування окремих компонентів та підвищує зрозумілість коду.

Для реалізації зберігання та обробки даних активно використовуються ORM-технології, зокрема Spring Data JPA, що базується на стандарті Java

Persistence API та інтегрується із сучасними реляційними СУБД. Це дозволяє розробникам працювати із сутностями у вигляді Java-об’єктів, уникати ручного написання SQL-запитів і забезпечувати цілісність та безпечність даних.

Окрім Spring Boot, на ринку представлені й інші популярні back-end фреймворки та платформи, такі як Node.js (з використанням Express), ASP.NET Core (на основі C#), Django (Python), Laravel (PHP) тощо. Кожне з цих рішень має власні переваги, проте екосистема Spring вирізняється потужною інтеграцією з корпоративними рішеннями, гнучкістю налаштувань, широкою підтримкою спільноти та великою кількістю інструментів для забезпечення безпеки та розширюваності.

У контексті створення інтернет-магазину побутової техніки, вибір Spring Boot як основної технології back-end є обґрунтованим завдяки його масштабованості, зручності, підтримці сучасних стандартів розробки та глибокої інтеграції з іншими Java-технологіями. Це дозволяє швидко створювати стабільні, безпечні й продуктивні серверні рішення, що відповідають високим вимогам ринку електронної комерції.

2.2.1 Особливості використання Spring Framework і Spring Boot

Spring Framework є одним із найбільш популярних та гнучких фреймворків для розробки корпоративних Java-додатків. Його ключова ідея полягає у спрощенні створення масштабованих, підтримуваних і модульних програм за допомогою інверсії керування (IoC), аспектно-орієнтованого програмування (AOP) та низки вбудованих сервісів для вирішення типових завдань. Spring дозволяє ефективно управляти залежностями між компонентами, забезпечує прозору конфігурацію, зручну організацію шарів додатку (контролери, сервіси, репозиторії) та легку інтеграцію з іншими бібліотеками.

Однак класичний Spring Framework вимагав значних зусиль для

первинного налаштування, конфігурування та підтримки проєкту, особливо для невеликих або середніх веб-застосунків. З метою спрощення цього процесу була розроблена надбудова – Spring Boot, яка автоматизує більшість рутинних налаштувань, мінімізує необхідність використання XML-конфігурацій і дозволяє сконцентруватися на бізнес-логіці.

Spring Boot забезпечує автоматичну конфігурацію основних компонентів програми, вбудований веб-сервер (наприклад, Tomcat), підтримку hot reload, інтеграцію з популярними базами даних, спрощену роботу з безпекою (Spring Security), RESTful API, email-сервісами, системами тестування тощо. Це значно скорочує час розробки та дозволяє швидко створити робочий прототип чи повноцінний застосунок із сучасною архітектурою.

Окремою перевагою Spring Boot є велика кількість стартерів (starters) – готових бібліотек для підключення часто використовуваних функцій, таких як робота з JPA, безпека, шаблонізація, моніторинг, документування REST API тощо. Крім того, Spring Boot чудово інтегрується з іншими модулями Spring (Spring MVC, Spring Data, Spring Security, Spring Cloud) і дозволяє розробляти як монолітні, так і мікросервісні архітектури.

У рамках створення інтернет-магазину побутової техніки використання Spring Boot дозволяє:

- швидко запускати й масштабувати проєкт;
- гарантувати структурованість та підтримуваність коду;
- легко реалізувати сучасний рівень безпеки та управління доступом;
- забезпечити просту інтеграцію з базою даних, платіжними сервісами, сторонніми API тощо.

Таким чином, Spring Framework у поєднанні зі Spring Boot є оптимальним вибором для розробки сучасних, гнучких і масштабованих веб-додатків, що відповідають вимогам сучасного бізнесу та електронної комерції.

2.2.2 Переваги архітектурного підходу MVC

Архітектурний підхід Model-View-Controller (MVC) є одним із найпоширеніших стандартів у розробці веб-застосунків, зокрема на платформі Java. Його ключова ідея полягає у чіткому розподілі додатку на три основні компоненти: модель (Model), подання (View) та контролер (Controller). Така структура забезпечує ряд суттєвих переваг для побудови сучасних і масштабованих інформаційних систем.

По-перше, MVC сприяє розділенню логіки додатку. Модель відповідає за доступ до даних та бізнес-логіку, подання – за формування інтерфейсу користувача, а контролер – за обробку запитів та взаємодію між моделлю й поданням. Завдяки цьому кожний компонент можна розробляти, тестувати й модифікувати незалежно від інших, що підвищує гнучкість і знижує ризик виникнення помилок при доопрацюванні системи[3][4].

По-друге, MVC-архітектура істотно спрощує підтримку та масштабування програмного забезпечення. Нові функції чи зміни у відображенні інтерфейсу можуть впроваджуватися без значного втручання у бізнес-логіку або доступ до даних. Це особливо важливо для проєктів, які активно розвиваються й вимагають регулярних оновлень.

По-третє, застосування MVC підвищує зручність командної розробки. Розробники можуть паралельно працювати над різними шарами застосунку: backend-спеціалісти – над моделлю й контролером, frontend-розробники – над поданням. Такий підхід зменшує конфлікти в коді та підвищує ефективність спільної роботи.

Крім того, використання MVC-архітектури у фреймворках на кшталт Spring MVC дозволяє легко інтегрувати зовнішні сервіси, впроваджувати RESTful API, тестувати окремі компоненти та забезпечувати розширюваність системи. MVC підходить як для монолітних, так і для мікросервісних архітектур, що додає універсальності рішенню.

Таким чином, застосування архітектурного підходу MVC у розробці

веб-додатків забезпечує чистоту коду, полегшує підтримку, розширення та командну роботу, що є особливо важливим для реалізації масштабованих та конкурентоспроможних інтернет-магазинів побутової техніки.

2.2.3 ORM-технології та Spring Data JPA

Однією з ключових вимог до сучасних веб-додатків є ефективна, безпечна та масштабована робота з базами даних. Для спрощення цього процесу у програмній інженерії широко використовуються ORM-технології (Object-Relational Mapping, об’єктно-реляційне відображення), які дозволяють взаємодіяти з реляційними базами даних за допомогою об’єктно-орієнтованих підходів.

ORM-технології автоматично виконують трансляцію між об’єктами в коді програми та записами у таблицях бази даних. Це суттєво знижує необхідність у ручному написанні SQL-запитів, зменшує ймовірність помилок та покращує читабельність і підтримку коду. Серед найпоширеніших ORM-фреймворків у Java-екосистемі можна виокремити Hibernate, EclipseLink, TopLink та інші.

Одним із найзручніших інструментів для інтеграції ORM-технологій у Spring-проєктах є Spring Data JPA. Цей модуль базується на стандарті Java Persistence API (JPA) і забезпечує високий рівень абстракції для роботи з базою даних. Spring Data JPA дозволяє автоматизувати рутинні CRUD-операції (створення, читання, оновлення, видалення), організовувати складні запити через методи репозиторіїв, а також легко підключати сторонні рішення для оптимізації роботи із великими обсягами даних.

Переваги використання Spring Data JPA:

- Автоматизація типових операцій. Розробник може зосередитися на бізнес-логіці, оскільки основні операції з даними виконуються фреймворком.
- Зручність і розширюваність. Інтерфейси репозиторіїв дозволяють швидко створювати методи для пошуку, сортування чи фільтрації даних без

написання SQL.

- Інтеграція з іншими модулями Spring. Spring Data JPA працює у зв’язці з Spring Boot, Spring Security, транзакційним менеджментом тощо.

- Підтримка різних СУБД. Легко змінювати тип бази даних, використовуючи різні драйвери та адаптери.

У контексті інтернет-магазину побутової техніки впровадження Spring Data JPA забезпечує високу продуктивність роботи із каталогом товарів, кошиком, замовленнями, дозволяє ефективно управляти даними користувачів і підтримувати складні зв’язки між сутностями. Завдяки ORM-технологіям спрощується масштабування та модернізація додатку, а також підвищується безпека і надійність системи.

2.3 Обґрунтування вибору СУБД MySQL

Ефективна організація зберігання, обробки та пошуку даних є однією з основних вимог до сучасних веб-додатків, зокрема інтернет-магазинів, де обсяги інформації про товари, користувачів, замовлення й транзакції постійно зростають. Саме тому вибір системи управління базами даних (СУБД) має ґрунтуватися на критеріях надійності, масштабованості, продуктивності та інтеграції з іншими компонентами програмної системи.

MySQL є однією з найпопулярніших реляційних СУБД у світі. Вона характеризується відкритим вихідним кодом, широкою підтримкою з боку спільноти, стабільністю та високою швидкістю роботи. Для веб-проєктів на Java, зокрема із використанням Spring Boot, MySQL легко інтегрується завдяки офіційним драйверам та розширенням, що дозволяють швидко налагодити взаємодію між додатком і базою даних[5].

Основними перевагами MySQL для застосування у проєктах електронної комерції є:

Висока продуктивність та оптимізація під web-навантаження. MySQL забезпечує швидке виконання транзакцій та роботу із великими обсягами

даних, що особливо важливо для інтернет-магазинів з великою кількістю товарів та замовлень.

Надійність і підтримка транзакційності. MySQL реалізує механізми ACID, що гарантує цілісність та консистентність даних навіть при одночасній роботі багатьох користувачів.

Гнучкість налаштувань та розширюваність. MySQL дозволяє легко масштабувати базу даних, додавати індекси, оптимізувати запити, підключати сторонні модулі для розширення функціональності.

Підтримка повнотекстового пошуку. Для інтернет-магазинів можливість швидко виконувати пошук за ключовими словами по каталогу товарів є істотною конкурентною перевагою.

Широка інтеграція із сучасними фреймворками. MySQL підтримується більшістю ORM-рішень, зокрема Hibernate та Spring Data JPA, що дає змогу використовувати переваги об’єктно-реляційного відображення без додаткових складнощів.

Варто також зазначити, що MySQL має потужні засоби резервного копіювання, відновлення даних та моніторингу, що дозволяє підтримувати стабільну роботу веб-додатку навіть за умов зростання навантаження чи виникнення технічних збоїв.

Таким чином, використання MySQL як основної СУБД для інтернет-магазину побутової техніки є оптимальним рішенням, що забезпечує надійність, швидкість, масштабованість і легку інтеграцію з іншими компонентами інформаційної системи.

2.4 Інструменти розробки, супроводу та контролю версій (Maven, Lombok, Git)

Ефективна організація зберігання, обробки та пошуку даних є однією з основних вимог до сучасних веб-додатків, зокрема інтернет-магазинів, де обсяги інформації про товари, користувачів, замовлення й транзакції

постійно зростають. Саме тому вибір системи управління базами даних (СУБД) має ґрунтуватися на критеріях надійності, масштабованості, продуктивності та інтеграції з іншими компонентами програмної системи.

MySQL є однією з найпопулярніших реляційних СУБД у світі. Вона характеризується відкритим вихідним кодом, широкою підтримкою з боку спільноти, стабільністю та високою швидкістю роботи. Для веб-проєктів на Java, зокрема із використанням Spring Boot, MySQL легко інтегрується завдяки офіційним драйверам та розширенням, що дозволяють швидко налагодити взаємодію між додатком і базою даних.

Основними перевагами MySQL для застосування у проєктах електронної комерції є висока продуктивність та оптимізація під web-навантаження. MySQL забезпечує швидке виконання транзакцій та роботу із великими обсягами даних, що особливо важливо для інтернет-магазинів з великою кількістю товарів та замовлень[6].

Надійність і підтримка транзакційності. MySQL реалізує механізми ACID, що гарантує цілісність та консистентність даних навіть при одночасній роботі багатьох користувачів.

Гнучкість налаштувань та розширюваність. MySQL дозволяє легко масштабувати базу даних, додавати індекси, оптимізувати запити, підключати сторонні модулі для розширення функціональності.

Підтримка повнотекстового пошуку. Для інтернет-магазинів можливість швидко виконувати пошук за ключовими словами по каталогу товарів є істотною конкурентною перевагою.

Широка інтеграція із сучасними фреймворками. MySQL підтримується більшістю ORM-рішень, зокрема Hibernate та Spring Data JPA, що дає змогу використовувати переваги об’єктно-реляційного відображення без додаткових складнощів.

Варто також зазначити, що MySQL має потужні засоби резервного копіювання, відновлення даних та моніторингу, що дозволяє підтримувати стабільну роботу веб-додатку навіть за умов зростання навантаження чи

виникнення технічних збоїв.

Таким чином, використання MySQL як основної СУБД для інтернет-магазину побутової техніки є оптимальним рішенням, що забезпечує надійність, швидкість, масштабованість і легку інтеграцію з іншими компонентами інформаційної системи.

3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ВЕБ-ДОДАТКУ

3.1 Архітектура веб-застосунку

Архітектура веб-застосунку інтернет-магазину побутової техніки базується на багаторівневій структурі, що забезпечує розділення відповідальностей, легкість підтримки, розширюваність та гнучкість системи. Головними принципами побудови архітектури є використання шаблону Model-View-Controller (MVC), чітке розмежування клієнтської та серверної частини, а також забезпечення безпечної взаємодії з базою даних.

Основні компоненти архітектури:

Клієнтська частина (Front-end): Реалізована за допомогою JSP (JavaServer Pages), JSTL (JavaServer Pages Standard Tag Library) та CSS-фреймворку Bootstrap. Клієнтська частина відповідає за формування інтерфейсу користувача, відображення динамічного контенту, а також забезпечує адаптивність та зручність взаємодії на різних пристроях.

Серверна частина (Back-end): Побудована на основі Spring Boot та модуля Spring MVC. Серверна частина обробляє HTTP-запити, здійснює маршрутизацію, виконує бізнес-логіку, координує взаємодію з базою даних і зовнішніми сервісами. Контролери приймають запити користувача, взаємодіють із сервісним шаром і передають дані для відображення у вигляді моделей.

База даних (Data Layer): Для зберігання даних використовується реляційна СУБД MySQL. Доступ до даних організовано через Spring Data JPA, що дозволяє працювати з інформацією у вигляді об’єктів та забезпечує ефективну роботу із запитами, транзакціями й захистом даних.

Сервісний шар (Service Layer):

Забезпечує реалізацію основної бізнес-логіки, виконує операції з товарами, користувачами, кошиком, замовленнями, а також координує взаємодію між контролерами та репозиторіями.

Безпека та автентифікація:
Для захисту даних, контролю доступу до функціоналу й безпечного зберігання інформації про користувачів впроваджено Spring Security. Аутентифікація, авторизація, а також захист від основних веб-загроз є обов'язковими складовими архітектури.

Інтеграція зі сторонніми сервісами:
Для забезпечення функцій онлайн-оплати інтегрується платіжна система LiqPay через RESTful API. Також архітектура передбачає можливість підключення інших зовнішніх сервісів (наприклад, сервісів доставки чи SMS-інформування).

Архітектурна схема:
Всі компоненти працюють у тісній взаємодії, забезпечуючи цілісність і узгодженість даних. Клієнтська частина ініціює запити, які потрапляють до серверної частини, далі обробляються контролерами та сервісами, після чого результат повертається користувачу у вигляді сторінки або JSON-даних (у випадку роботи з REST API).

Переваги обраної архітектури:

- Спрощене розширення функціоналу;
- Легкість підтримки та тестування окремих модулів;
- Висока безпека та захист даних користувачів;
- Можливість масштабування й інтеграції з іншими сервісами.
- Таким чином, архітектура веб-застосунку дозволяє забезпечити надійну роботу інтернет-магазину побутової техніки, відповідає сучасним стандартам і вимогам ринку електронної комерції.

Архітектура сучасного інтернет-магазину побудована за модульним принципом, що дозволяє ефективно організувати всі основні функції платформи: каталогізацію товарів, управління обліковими записами,

формування замовлень, списки побажань та інші сервіси для користувачів.

На рисунку 3.1 наведено типову структурну схему платформи Amazon, яка може слугувати зразком для проєктування архітектури власного інтернет-магазину побутової техніки.

Головна сторінка (Home Amazon) є центральною точкою входу до системи, звідки користувачі можуть перейти до різних функціональних розділів.

Категорії товарів поділяються на тематичні підгрупи (наприклад, "Books and Audible", "Digital Music"). Кожна категорія містить відповідні підкатегорії чи списки товарів ("Product Listing Children’s Books", "Product Listing Digital Music Store").

Списки продуктів ведуть до конкретних товарних позицій, які згруповані у вигляді списків або карток (наприклад, "Products Books", "Products Albums").

Оформлення замовлення (Checkout) є кінцевим етапом, де користувач підтверджує покупку вибраних товарів.

Окрім каталогу, платформа має особисті кабінети користувачів (Your Account), де зібрана інформація про замовлення ("Your Orders"), списки бажань ("Your Wishlist"), а також реєстри спеціальних подій (наприклад, "Wedding Registry", "Baby Registry").

Система бажаних товарів (WishList) дає змогу зберігати цікаві позиції та використовувати їх для майбутніх покупок.

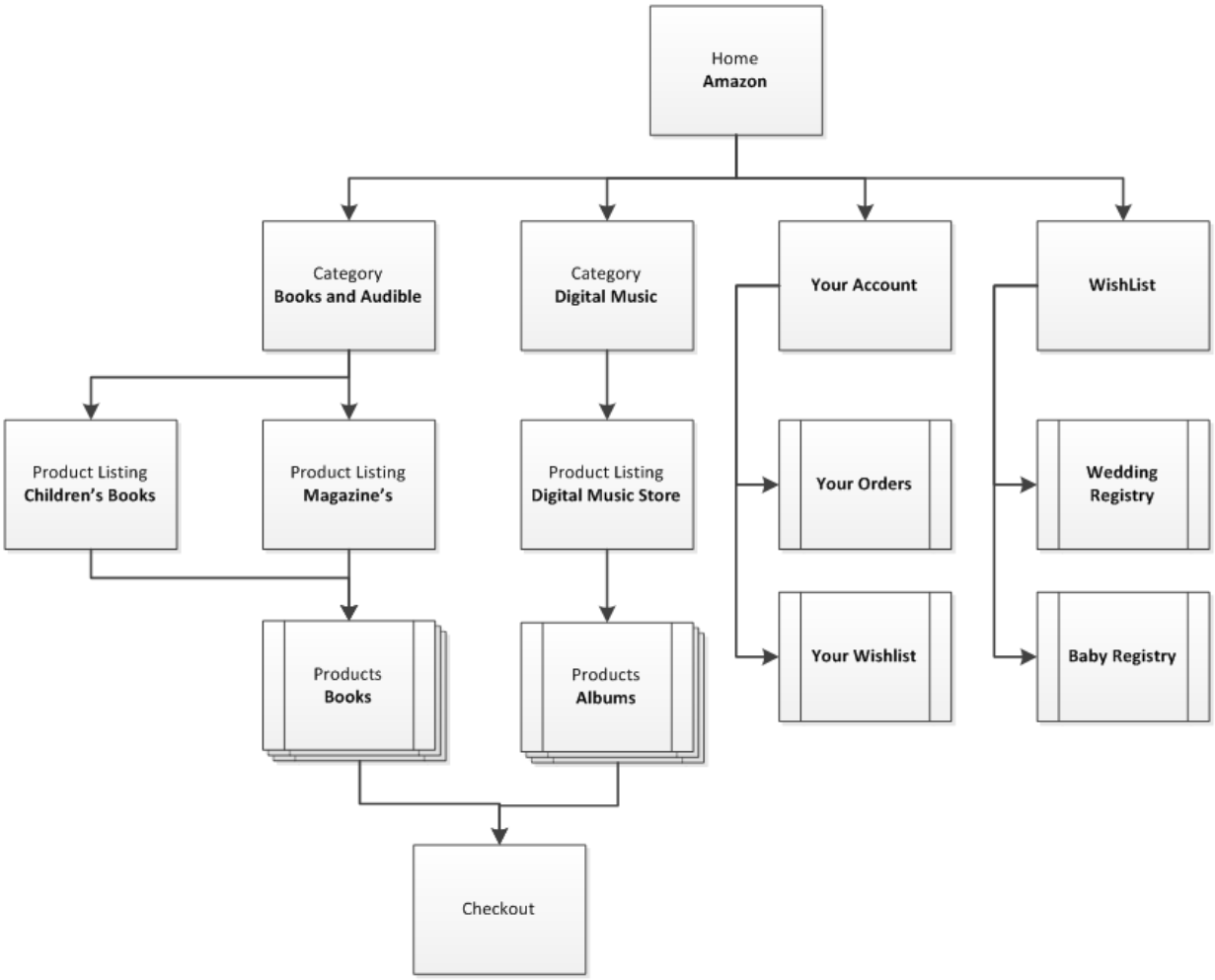

Рисунок 3.1 – Структурна архітектурна схема інтернет-магазину Amazon

Цей підхід може бути застосований і для створення власної платформи, наприклад, для маркетплейсу побутової техніки або маркетплейсу вживаних товарів. Гнучкість і модульність архітектури дозволяють реалізувати весь спектр сучасних вимог до онлайн-комерції: від індивідуальних кабінетів до складних механізмів обліку та управління замовленнями.

3.1.1 Загальна схема компонентів системи

Архітектурна модель інтернет-магазину побутової техніки будується на основі взаємодії кількох ключових компонентів, кожен із яких виконує власні функції в межах багаторівневої структури системи. Такий підхід забезпечує чіткий розподіл відповідальності, гнучкість, масштабованість і полегшує

подальший розвиток та супровід веб-застосунку.

Основні компоненти системи:

- Інтерфейс користувача (UI, Front-end): Відповідає за взаємодію з кінцевим користувачем, відображення товарів, форм, кошика, сторінок оформлення замовлення та профілю. Здійснюється за допомогою JSP/JSTL, CSS, Bootstrap.
- Контролери (Controllers): Приймають HTTP-запити від користувачів, визначають подальший маршрут обробки, викликають необхідні сервіси, формують моделі для подання даних у відповідь[7].
- Сервісний шар (Services): Реалізує основну бізнес-логіку проєкту – операції з товарами, користувачами, обробку кошика та замовлень, валідацію даних, взаємодію зі сторонніми сервісами (зокрема, з платіжною системою LiqPay).
- Репозиторії (Repositories): Відповідають за доступ до бази даних, виконання CRUD-операцій, пошук, фільтрацію та роботу із транзакціями. Побудовані на основі Spring Data JPA, що дозволяє інтегрувати ORM-рівень із бізнес-логікою.
- База даних (MySQL): Зберігає всю інформацію про користувачів, товари, категорії, замовлення, історію оплат тощо. Гарантує цілісність, захищеність і швидкий доступ до даних.
- Модулі безпеки: Реалізовані на основі Spring Security та відповідають за аутентифікацію, авторизацію, контроль доступу до різних функцій, захист від поширених веб-загроз.
- Зовнішні інтеграції: Передбачають підключення до платіжної системи LiqPay, а також можливість взаємодії з іншими зовнішніми сервісами (наприклад, службами доставки чи сповіщення).

- Інформаційний потік у системі передбачає, що користувач надсилає запит через UI, який потрапляє до контролера. Контролер, у разі потреби, звертається до сервісного шару, який може взаємодіяти з репозиторіями для роботи з даними. Після обробки інформація повертається до користувача у вигляді сторінки або відповідного повідомлення.

- Таким чином, загальна схема компонентів системи забезпечує ефективну та стабільну роботу інтернет-магазину, дозволяє реалізувати всі необхідні бізнес-процеси та гарантує безпечну роботу з даними.

3.1.2 Моделювання бази даних (ER-діаграми)

Якісне моделювання бази даних є ключовим етапом проєктування інформаційної системи інтернет-магазину побутової техніки. Від структури даних залежать швидкодія, надійність, масштабованість та коректність реалізації основних бізнес-процесів. Для графічного представлення структури даних та їх взаємозв’язків застосовується ER-моделювання (Entity-Relationship diagram, ER-діаграма).

Основні сутності (Entity) системи:

- Користувач (User): зберігає персональні дані, контактну інформацію, облікові параметри та ролі користувачів (адміністратор, покупець).

- Товар (Product): містить назву, опис, ціну, характеристики, категорію, наявність, зображення.

- Категорія (Category): групує товари за типом, призначенням чи іншими ознаками.

- Замовлення (Order): фіксує дані про покупку, статус замовлення, дату створення, спосіб оплати й доставки.

- Кошик (Cart): зберігає список вибраних товарів для певного користувача до моменту оформлення замовлення.

- Деталі замовлення (OrderItem): зв’язує замовлення з конкретними товарами та їх кількістю.

- Відгук/Рейтинг (Review): містить відгуки користувачів про товари, рейтинг, дату залишення.

- Основні зв’язки між сутностями:

- Один Користувач може мати кілька Замовлень та лише один активний Кошик.

- Замовлення складається з багатьох Деталей замовлення (OrderItem), кожна з яких пов’язана з конкретним Товаром.

- Товар належить до певної Категорії.

- Користувач може залишати багато Відгуків для різних Товарів.

3.2 Розробка клієнтської частини веб-застосунку

Клієнтська частина веб-застосунку є надзвичайно важливим елементом системи, адже саме через неї відбувається взаємодія користувача з інтернет-магазином побутової техніки. Від якості розробки інтерфейсу залежить не лише перше враження про ресурс, а й ефективність пошуку товарів, зручність оформлення замовлень, доступність сервісів підтримки та загальний рівень задоволення користувача[8][9].

У межах даного проєкту клієнтська частина реалізована на основі технологій JSP (JavaServer Pages) та JSTL (JavaServer Pages Standard Tag Library), що дозволяє динамічно формувати HTML-сторінки на сервері й підключати до них актуальні дані про товари, кошик, замовлення та користувачів. Використання JSTL спрощує ітерацію по списках, умовне відображення блоків, форматування дат і чисел, що суттєво підвищує гнучкість і читабельність коду інтерфейсу.

Для забезпечення сучасного зовнішнього вигляду та адаптивності сторінок у проєкті використовується фреймворк Bootstrap. Він надає велику кількість готових компонентів і шаблонів (навігаційні панелі, кнопки, форми, картки товарів тощо), які дозволяють швидко створити привабливий, зручний та кросбраузерний інтерфейс. Адаптивний дизайн гарантує коректне

відображення й повноцінну роботу інтернет-магазину на різних пристроях, включно зі смартфонами й планшетами.

Основні елементи клієнтської частини включають:

- Головну сторінку з інформаційними банерами, категоріями товарів і спеціальними пропозиціями;
- Каталог продукції з можливістю фільтрації, сортування та пошуку за різними критеріями;
- Сторінки товарів з детальним описом, відгуками, рейтингом і кнопкою додавання до кошика;
- Особистий кабінет користувача для перегляду профілю, історії замовлень, редагування особистої інформації;
- Форми реєстрації, авторизації та відновлення доступу;
- Сторінку кошика для управління замовленнями, зміни кількості товарів, оформлення покупки;
- Сторінки зворотного зв’язку, часто задаваних питань і підтримки.
- Для підвищення інтерактивності окремих елементів застосовується JavaScript, що дозволяє реалізовувати перевірку форм, динамічне оновлення кошика, спливаючі повідомлення тощо.

Завдяки використанню перевірених технологій JSP/JSTL та Bootstrap клієнтська частина веб-застосунку поєднує сучасний вигляд, швидкодію, гнучкість налаштувань і повну відповідність потребам кінцевих користувачів. Це створює надійну основу для успішного функціонування інтернет-магазину побутової техніки та підвищує його конкурентоспроможність на ринку.

3.2.1 Проєктування та розробка інтерфейсу користувача

Інтерфейс користувача (UI) є визначальною складовою будь-якого сучасного веб-застосунку, оскільки саме через нього відбувається основна комунікація користувача із системою. Вдалий дизайн інтерфейсу здатний не

лише підвищити ефективність використання ресурсу, а й сформувати позитивне враження, стимулювати повторні покупки та збільшити лояльність клієнтів до бренду.

Етапи проєктування інтерфейсу:

- Визначення основних сценаріїв користувача. Аналізуються типові дії: перегляд каталогу, пошук і сортування товарів, додавання до кошика, оформлення замовлення, перегляд профілю, написання відгуків тощо.
- Розробка прототипів і макетів. Використовуються інструменти для створення схематичних ескізів сторінок, що дозволяє протестувати логіку розташування елементів, шлях користувача, зручність навігації.
- Вибір стилістики та кольорової гами. Перевага надається сучасним, лаконічним і легким для сприйняття рішенням, що асоціюються із чистотою, технологічністю та довірою (наприклад, світлі тони, акцентні кнопки, читабельні шрифти).
- Створення адаптивної верстки. За допомогою Bootstrap та медіа-запитів забезпечується коректне відображення інтерфейсу на різних пристроях: ПК, планшетах, смартфонах.

3.2.2 Реалізація функціоналу реєстрації та авторизації користувачів

Однією з базових функцій сучасного інтернет-магазину є організація безпечної системи реєстрації та авторизації користувачів. Це необхідно для забезпечення індивідуалізації сервісу, ведення історії замовлень, налаштування профілю, збереження кошика й реалізації додаткових сервісів для зареєстрованих клієнтів.

Процес реєстрації користувача Реєстрація здійснюється за допомогою спеціальної веб-форми, де користувач заповнює основні персональні дані: ім’я, адресу електронної пошти, номер

телефону, пароль та, за потреби, адресу доставки. Під час обробки форми реалізовано перевірку унікальності email-адреси, відповідність пароля вимогам безпеки та валідацію інших полів (наприклад, коректність електронної пошти або номера телефону). Дані користувача зберігаються у захищеному вигляді у базі даних, а паролі – у вигляді хешу з використанням сучасних криптографічних алгоритмів (наприклад, BCrypt).

Механізм авторизації
Авторизація (вхід до особистого кабінету) здійснюється через введення email та пароля. Після підтвердження правильності даних користувач отримує сесію з відповідними правами доступу. У разі невірного введення даних система повідомляє про помилку, не розкриваючи зайвої інформації для захисту від атак типу “brute force”.

Забезпечення безпеки та ролей
Для реалізації контролю доступу впроваджено ролі користувачів (звичайний користувач, адміністратор). Для кожної ролі визначено відповідний набір прав: перегляд і редагування профілю, оформлення замовлень, доступ до адміністративної панелі тощо. Система реєстрації й авторизації базується на можливостях Spring Security, що забезпечує:

- захист від несанкціонованого доступу;
- можливість розмежування доступу до окремих сторінок та функцій;
- захист від поширених загроз, зокрема CSRF-атак.

Наприклад, сервіс UserService реалізує логіку реєстрації користувача (лістинг 3.1)

Лістинг 3.1 – Лістинг сервісу UserService

```
@Service
public class UserService {

    @Autowired
    private UserRepository userRepository;

    @Autowired
    private PasswordEncoder passwordEncoder;
```

```
    public void register(User user) {
        // Хешування пароля перед збереженням
        String encodedPassword =
passwordEncoder.encode(user.getPassword());
        user.setPassword(encodedPassword);
        user.setRole("USER");
        user.setActive(true);
        userRepository.save(user);
    }
}
```

3.2.3 Каталог продукції: перегляд, пошук та фільтрація товарів

Каталог продукції в інтернет-магазині побутової техніки є центральним елементом для забезпечення зручного та ефективного пошуку й вибору товарів користувачами. Його реалізація передбачає динамічне відображення асортименту, що містить найважливішу інформацію про кожен товар: назву, категорію, бренд, ціну, наявність, основні характеристики, фотографії, рейтинг та відгуки інших покупців. Вся ця інформація отримується з бази даних і оновлюється у режимі реального часу, що дозволяє відображати тільки актуальні пропозиції[10].

Важливою складовою каталогу є функціонал пошуку, який дає змогу користувачам швидко знаходити потрібні товари за ключовими словами, артикулом, назвою або описом. Для цього пошуковий механізм підтримує різноманітні запити, враховуючи навіть часткові збіги слів, а також пропонує підказки під час введення тексту, що значно пришвидшує процес. Система автоматично виводить результати пошуку у вигляді зручного списку з можливістю переходу до детальної сторінки обраного товару.

Ще однією важливою опцією є фільтрація продукції за різними параметрами: категоріями, брендами, ціновим діапазоном, наявністю на складі, рейтингом, популярністю тощо. Для цього на сторінці каталогу передбачено інтуїтивно зрозумілі фільтри та панель сортування, які дозволяють користувачам миттєво звузити коло пошуку й відібрати лише ті товари, що відповідають їхнім потребам. Додатково реалізовано сортування

за різними критеріями – наприклад, за зростанням чи спаданням ціни, за рейтингом, за датою надходження або за популярністю серед покупців.

Використання сучасних технологій JSP/JSTL разом із фреймворком Bootstrap дає змогу зробити інтерфейс каталогу адаптивним, легким у сприйнятті й однаково зручним як на персональних комп'ютерах, так і на мобільних пристроях. Структура сторінки є логічною та зрозумілою: основна частина займається виведенням списку товарів, а бокова або верхня панель – розміщенням фільтрів та пошукового рядка. Для покращення користувацького досвіду додано можливість зберігати вибрані товари у "Обране", порівнювати кілька позицій одночасно та переглядати акційні пропозиції окремо.

Якісний функціонал каталогу, пошуку й фільтрації значно підвищує рівень задоволення користувачів, сприяє швидкому прийняттю рішень про покупку і збільшує загальну ефективність роботи інтернет-магазину побутової техніки.

3.2.4 Система кошика та оформлення замовлень

Система кошика в інтернет-магазині побутової техніки реалізована таким чином, щоб забезпечити максимально простий, швидкий і зрозумілий процес покупки для користувача. При виборі товару на сторінці каталогу чи детального перегляду, покупець має можливість одним натисканням додати товар до кошика, після чого він відображається у спеціальному розділі сайту. У кошику зберігається повний список обраних позицій із зазначенням кількості, ціни за одиницю, загальної вартості, а також доступними опціями для видалення окремого товару або зміни кількості.

Зміни, які користувач вносить до вмісту кошика, відразу відображаються на сторінці – оновлюється підсумкова сума, кількість товарів, можливі знижки чи акційні пропозиції. Для авторизованих користувачів вміст кошика зберігається на сервері, що дозволяє продовжити

оформлення покупки навіть після повторного входу з іншого пристрою. Для неавторизованих відвідувачів дані можуть тимчасово зберігатися у сесії браузера.

Процес оформлення замовлення також максимально автоматизований. Після переходу до оформлення замовлення користувачу пропонується перевірити вміст кошика, вказати або підтвердити контактну інформацію та адресу доставки, обрати спосіб оплати (наприклад, онлайн через платіжну систему чи при отриманні), а також залишити коментар або обрати додаткові сервіси. Для зареєстрованих користувачів дані можуть підтягуватися автоматично з профілю, що скорочує час на введення інформації.

Особливу увагу приділено безпеці оформлення замовлення: усі дані передаються захищеними каналами, а оплата відбувається через інтегровану платіжну систему, що відповідає сучасним стандартам безпеки. Після успішного оформлення замовлення користувач отримує підтвердження на екрані та, додатково, на вказану електронну пошту. Замовлення зберігається в особистому кабінеті користувача, де можна переглянути його статус, історію покупок, а також за необхідності оформити повторну покупку.

Завдяки такій системі реалізується не лише зручний і інтуїтивно зрозумілий процес покупки, а й підвищується довіра користувачів до інтернет-магазину, що є важливим фактором успіху електронної комерції.

3.3 Розробка серверної частини веб-застосунку

Розробка серверної частини веб-застосунку інтернет-магазину побутової техніки базується на використанні сучасних технологій Java та фреймворку Spring Boot, що дозволяє забезпечити високу продуктивність, гнучкість і безпеку системи. Серверна частина відповідає за обробку всіх бізнес-процесів, взаємодію з базою даних, реалізацію авторизації, реєстрації, управління каталогом, обробку кошика та замовлень, інтеграцію із зовнішніми сервісами та платіжними системами.

Головною архітектурною особливістю є застосування багаторівневої моделі із розділенням на контролери, сервіси та репозиторії. Контролери приймають HTTP-запити з клієнтської частини, відповідають за маршрутизацію, валідацію введених даних і формування відповідей. Сервісний шар реалізує бізнес-логіку: виконання операцій із користувачами, товарами, кошиком, обробкою замовлень, а також взаємодію зі сторонніми API. Репозиторії здійснюють прямий доступ до бази даних через Spring Data JPA, забезпечують виконання CRUD-операцій, пошук, фільтрацію та роботу з транзакціями.

Для організації взаємодії клієнтської та серверної частин реалізовано RESTful API, що дозволяє ефективно обмінюватися даними у форматі JSON, забезпечувати інтеграцію з мобільними застосунками чи сторонніми сервісами. Для підвищення безпеки серверна частина використовує Spring Security, що дозволяє захищати маршрути, керувати ролями користувачів та реалізовувати багатофакторну аутентифікацію.

Додатково впроваджуються механізми обробки помилок, логування, тестування та моніторингу системи. Це дозволяє не лише забезпечити стабільну та безперебійну роботу веб-застосунку, а й оперативно реагувати на можливі проблеми чи підвищене навантаження. Таким чином, серверна частина є ядром інформаційної системи, що гарантує цілісність, безпеку і високу якість обслуговування користувачів інтернет-магазину побутової техніки.

3.3.1 Реалізація RESTful API за допомогою Spring MVC

Реалізація RESTful API за допомогою Spring MVC є одним із ключових аспектів серверної частини сучасного веб-застосунку інтернет-магазину побутової техніки. Використання REST-архітектури дозволяє створити простий, масштабований і зручний для інтеграції інтерфейс взаємодії між клієнтом і сервером, що особливо актуально для обміну даними з

мобільними додатками, сторонніми сервісами або SPA-фронтендами.

Spring MVC надає зручний інструментарій для створення REST-контролерів, які відповідають за обробку HTTP-запитів різних типів (GET, POST, PUT, DELETE) та повертають дані у форматі JSON або XML. Для кожного ресурсу (наприклад, товар, користувач, замовлення) створюється окремий контролер, у якому визначаються відповідні ендпоїнти для виконання операцій читання, створення, оновлення та видалення.

У процесі розробки RESTful API особлива увага приділяється коректній організації маршрутів, структурі відповідей, обробці помилок та валідації вхідних даних. Для цього використовуються стандартні HTTP-статуси, механізми обробки винятків (Exception Handling), а також засоби документації API, такі як Swagger/OpenAPI. Це спрощує тестування, підтримку та розширення інтерфейсу. Зараз, давайте подивимось на приклад контроллеру ProductRestController (лістинг 3.2).

Лістинг 3.2 – Лістинг сервісу ProductRestController, що реалізує стандартний RESTful інтерфейс для взаємодії з ресурсом Product:

```
@RestController
@RequestMapping("/api/products")
public class ProductRestController {

    @Autowired
    private ProductService productService;

    // Отримати список усіх товарів
    @GetMapping
    public List<Product> getAllProducts() {
        return productService.findAll();
    }

    // Отримати товар за ID
    @GetMapping("/{id}")
    public ResponseEntity<Product> getProductById(@PathVariable
Long id) {
        Optional<Product> product = productService.findById(id);
        return product.map(ResponseEntity::ok)

.orElse(ResponseEntity.notFound().build());
```

```
    }

    // Створити новий товар
    @PostMapping
    public ResponseEntity<Product> createProduct(@RequestBody
Product product) {
        Product saved = productService.save(product);
        return new ResponseEntity<>(saved, HttpStatus.CREATED);
    }

    // Оновити існуючий товар
    @PutMapping("/{id}")
    public ResponseEntity<Product> updateProduct(@PathVariable
Long id, @RequestBody Product updatedProduct) {
        Optional<Product> existing =
productService.findById(id);
        if (existing.isPresent()) {
            updatedProduct.setId(id);
            Product saved = productService.save(updatedProduct);
            return ResponseEntity.ok(saved);
        } else {
            return ResponseEntity.notFound().build();
        }
    }

    // Видалити товар
    @DeleteMapping("/{id}")
    public ResponseEntity<Void> deleteProduct(@PathVariable Long
id) {
        if (productService.existsById(id)) {
            productService.deleteById(id);
            return ResponseEntity.noContent().build();
        } else {
            return ResponseEntity.notFound().build();
        }
    }
}
```

Завдяки впровадженню RESTful API веб-застосунок отримує низку важливих переваг:

- Відокремлення клієнтської та серверної логіки, що полегшує супровід і розвиток системи;

- Можливість легкого підключення зовнішніх сервісів, мобільних додатків, автоматизації процесів;

- Стандартизований спосіб обміну даними, який спрощує інтеграцію і знижує ймовірність помилок;

- Підвищення масштабованості і гнучкості архітектури.

Таким чином, реалізація RESTful API за допомогою Spring MVC забезпечує сучасний рівень сервісу, відкриває нові можливості для розвитку інтернет-магазину та дозволяє легко інтегруватися з іншими цифровими продуктами.

3.3.2 Організація доступу до даних через Spring Data JPA

Організація доступу до даних у веб-застосунку інтернет-магазину побутової техніки здійснюється за допомогою Spring Data JPA, що є потужним інструментом для взаємодії з реляційними базами даних у Java-проєктах. Використання Spring Data JPA дозволяє значно спростити процес розробки, оскільки більшість стандартних операцій над даними (створення, читання, оновлення, видалення) автоматизуються і виконуються за допомогою готових методів репозиторіїв[11].

Кожна сутність у базі даних, наприклад, користувач, товар, замовлення чи категорія, описується у вигляді окремого класу-моделі (Entity), а для роботи з цими сутностями створюються інтерфейси-репозиторії, які наслідують стандартні інтерфейси Spring Data JPA, такі як JpaRepository або CrudRepository. Це дозволяє здійснювати всі основні операції з даними без необхідності написання SQL-запитів: додавання нових записів, отримання списку всіх об'єктів певного типу, пошук за параметрами, редагування та видалення.

Для складніших сценаріїв, наприклад, пошуку за декількома полями, фільтрації або сортування, Spring Data JPA дозволяє легко створювати власні методи з використанням зрозумілих імен методів, які автоматично транслюються у відповідні SQL-запити. За потреби можна реалізувати і нативні запити, а також використовувати JPQL (Java Persistence Query Language).

Доступ до даних у Spring Data JPA є безпечним і підтримує

транзакційність, що гарантує цілісність інформації навіть за одночасної роботи кількох користувачів. Додатковою перевагою є інтеграція з іншими модулями Spring, наприклад, Spring Security, що дозволяє обмежувати доступ до певних даних відповідно до ролей та прав користувача.

Завдяки впровадженню Spring Data JPA структура роботи з даними стає зрозумілою, масштабованою та легкою у підтримці. Це забезпечує стабільну взаємодію між бізнес-логікою застосунку та базою даних, прискорює розробку, знижує ймовірність помилок і полегшує майбутню модернізацію інформаційної системи.

3.3.3 Адміністративна панель управління товарами та замовленнями

Адміністративна панель управління товарами та замовленнями є важливою складовою веб-застосунку інтернет-магазину побутової техніки, оскільки дозволяє адміністраторам та менеджерам ефективно здійснювати контроль над асортиментом, оперативно реагувати на зміни та забезпечувати якість обслуговування клієнтів. Доступ до адміністративного функціоналу реалізовано через окремий розділ сайту, вхід до якого дозволено лише користувачам із відповідними правами (адміністраторам або менеджерам).

Основні можливості панелі управління включають перегляд, додавання, редагування та видалення товарів у каталозі. Адміністратор може швидко змінювати ціни, наявність, описи, фотографії, параметри техніки, призначати товари до відповідних категорій або позначати їх як акційні чи новинки. Для зручності реалізовано функцію пошуку й фільтрації товарів у базі, а також масове оновлення або видалення позицій за певними критеріями.

Ще одним важливим напрямком є управління замовленнями. Адміністративна панель дозволяє переглядати список усіх поточних, завершених та скасованих замовлень із деталями по кожному: склад замовлення, контактні дані клієнта, вибраний спосіб доставки й оплати,

поточний статус, історія змін і коментарі. Адміністратор має можливість змінювати статус замовлення (наприклад, «очікує обробки», «виконується», «відправлено», «завершено», «скасовано»), надсилати сповіщення клієнту, формувати звіти за період та аналізувати статистику продажів.

З технічного боку, реалізація адміністративної панелі базується на тих самих принципах безпеки, що і весь веб-застосунок. Для захисту від несанкціонованого доступу використовується Spring Security, реалізовано багаторівневу автентифікацію, валідацію даних і обмеження доступу до критичних функцій. Інтерфейс панелі створено з урахуванням вимог до зручності й ефективності роботи: використано адаптивний дизайн, зручну навігацію, інформативні таблиці та форми для швидкого редагування.

Наявність повнофункціональної адміністративної панелі дозволяє оперативно управляти бізнес-процесами, підвищувати якість обслуговування, зменшувати кількість помилок і прискорювати прийняття управлінських рішень, що є одним із ключових чинників успішної роботи сучасного інтернет-магазину побутової техніки.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було спроєктовано, реалізовано та протестовано повноцінний веб-застосунок маркетплейсу, призначений для купівлі та продажу вживаних товарів між користувачами. Розробка охопила всі етапи створення програмного забезпечення – від аналізу предметної області до розгортання та тестування продукту в серверному середовищі.

На основі вивчених технологій та проведеного аналізу було обґрунтовано доцільність використання стеку Java/Spring для реалізації серверної логіки. Було впроваджено такі компоненти:

- Spring Boot як основа серверного застосунку з вбудованим Tomcat;
- Spring Security для організації системи ролей і захисту доступу;
- Spring Data JPA та Hibernate для ефективної роботи з реляційною базою MySQL;
- Thymeleaf і Bootstrap для побудови динамічного та адаптивного інтерфейсу;
- Lombok, Maven, DevTools для підтримки й автоматизації процесу розробки.

Було реалізовано такі функціональні можливості:

- реєстрація, вхід і авторизація користувачів;
- публікація, редагування, перегляд і видалення оголошень;
- категоризація товарів і пошук із фільтрами;
- додавання товарів до кошика, оформлення замовлень;
- інтеграція з платіжною системою LiqPay для онлайн-оплати;
- система відгуків і коментарів;
- історія замовлень, перегляд куплених товарів;
- багатомовна підтримка (українська, англійська);
- інтерфейс адміністратора з можливістю модерування.

Окрему увагу приділено безпеці застосунку – реалізовано валідацію даних на сервері, захист від CSRF і XSS-атак, шифрування паролів та ізольоване збереження платіжної інформації.

Після завершення розробки система була успішно розгорнута на віддаленому сервері, проведено тестування всіх основних модулів, налагоджено логування, забезпечено захищене з'єднання через HTTPS. Функціонування застосунку підтверджено в умовах, наближених до реального середовища експлуатації.

Результатом роботи став готовий до використання веб-застосунок, що відповідає сучасним вимогам до якості, безпеки, масштабованості та зручності для кінцевого користувача. Реалізована архітектура дозволяє розвивати проєкт у майбутньому, зокрема впроваджувати нові бізнес-функції, розширювати підтримку мов, підключати додаткові платіжні шлюзи або мобільний клієнт.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Spring Boot Documentation: Офіційна документація Spring Boot, що містить керівництва та API-документацію [Електронний ресурс]. – 2020. – Режим доступу до ресурсу: https://spring.io/projects/spring-boot.

2. Spring in Action, 4th edition / Craig Walls. MANNING Shelter Island, 2019 – 500 с. – ISBN: 978-1617291203.

3. Java Persistence API (JPA) Documentation [Електронний ресурс]/ – 2019. – Режим доступу до ресурсу: https://docs.oracle.com/javaee/7/tutorial/persistence-intro.html.

4. Spring Security Documentation: Офіційна документація Spring Security, що охоплює аутентифікацію, авторизацію та захист веб-застосунків [Електронний ресурс]. – 2023. – Режим доступу до ресурсу: https://spring.io/projects/spring-security.

5. MySQL Documentation: Офіційна документація MySQL, яка розкриває можливості СУБД та її налаштування [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://dev.mysql.com/doc/.

6. Hibernate ORM Documentation: Документація Hibernate ORM, яка пояснює основні функції та налаштування для роботи з базами даних. [Електронний ресурс]. – 2020. – Режим доступу до ресурсу: https://hibernate.org/orm/documentation/.

7. Thymeleaf Documentation: Офіційна документація Thymeleaf, сучасного серверного шаблонізатора Java [Електронний ресурс]. – 2019. – Режим доступу до ресурсу: https://www.thymeleaf.org/documentation.html.

8. Bootstrap Documentation: Офіційна документація Bootstrap, CSS-фреймворку для розробки адаптивних веб-сторінок [Електронний ресурс]. – 2020. – Режим доступу до ресурсу: https://getbootstrap.com/docs/5.0/getting-started/introduction/.

9. Maven Documentation: Офіційна документація Maven, що описує

управління проектами та залежностями [Електронний ресурс]. – 2023. – Режим доступу до ресурсу: https://maven.apache.org/guides/index.html.

10. Martin F. Patterns of Enterprise Application Architecture / Fowler Martin. – Київ: Addison-Wesley Professional, 2020. – 544 с. – (Signature Series; 4).

11. Роберт М. Чистий код / Мартин Роберт. – Київ: Фабула, 2019. – 448 с.