Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем та програмної інженерії
(повна назва факультету)
Кафедра програмної інженерії
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(назва освітнього ступеня)

на тему: Розробка автоматизованої веб-системи з використанням CMS Opencart, мови програмування PHP, клієнтської бібліотеки JQuery та препроцесора SCSS

Виконав(ла): студент(ка) 6 курсу, групи СПм-61
спеціальності 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

(підпис) Філик В. В. (прізвище та ініціали)

Керівник (підпис) Цуприк Г. Б (прізвище та ініціали)

Нормоконтроль (підпис) Стоянов Ю. М. (прізвище та ініціали)

Завідувач кафедри (підпис) Петрик М. Р. (прізвище та ініціали)

Рецензент (підпис) Стадник Н. Б. (прізвище та ініціали)

Тернопіль
2025

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет Факультет комп'ютерно-інформаційних систем та програмної інженерії
(повна назва факультету)

Кафедра Програмної інженерії
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Петрик М. Р.
(підпис) (прізвище та ініціали)
« » 2025 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня магістр
(назва освітнього ступеня)

за спеціальністю 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

студенту Філику Віктору Васильовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка автоматизованої веб-системи із використанням CMS Opencart, мови програмування PHP, клієнтської бібліотеки JQuery та препроцесора SCSS

Керівник роботи Цуприк Галина Богданівна к.т.н. доц.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «27» листопада 2025 року № 4/7-1045

2. Термін подання студентом завершеної роботи

3. Вихідні дані до роботи

4. Зміст роботи (перелік питань, які потрібно розробити)
Сформулювати та погодити тему кваліфікаційної роботи;
розробити та затвердити завдання;
проаналізувати завдання, зробити огляд та проаналізувати бібліографічні матеріали щодо стану справ та тенденцій в обраному напрямку дослідження;
сформулювати завдання, обґрунтувати цілі дослідження;
розробити та спроєктувати систему;
описати технології, етапи розробки та саму розроблену програмну систему (у разі потреби);
розробити план тестування програмного продукту; оформити допоміжну документацію;
проаналізувати роботу щодо питань з дотримання положень про охорону праці та безпеку в надзвичайних ситуаціях; оформити пояснювальну записку;
зробити відповідні висновки за результатами виконаної роботи.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
Ілюстративна частина кваліфікаційної роботи оформляється у вигляді слайдів, висвітлює основні розділи та етапи роботи та містить: тему роботи, мету, предмет та об'єкт дослідження, сформульовані завдання, перелічено етапи виконання кваліфікаційної роботи, короткий аналіз актуальності теми, варіант архітектури програмної системи, представлено опис програмного забезпечення; опис програмного забезпечення та тестування.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Вибір та погодження теми з керівником. Створення та затвердження технічного завдання.		
2.	Аналіз технічного завдання, підбір та аналіз		
	бібліографічних матеріалів необхідних для виконання		
	кваліфікаційної роботи		
3.	Проєктування та розробка програмного забезпечення		
4.	Тестування та дослідна експлуатація програмного		
	забезпечення		
5.	Створення допоміжної документації. Написання та		
	перевірка на відповідність вимогам пояснювальної		
	записки		
6.	Апробація результатів роботи		
7.	Перевірка програмою антиплагіат		
8.	Охорона праці		
9.	Безпека в надзвичайних ситуаціях		
10.	Формування записки кваліфікаційної роботи		
11.	Нормконтроль		
12.	Попередній захист		
13.	Рецензування кваліфікаційної роботи		
14.	Захист кваліфікаційної роботи		

Студент _____ Філик В. В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Цуприк Г. Б.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Філик Віктор Васильович. Розробка автоматизованої веб-системи із використанням CMS Opencart, мови програмування PHP, клієнтської бібліотеки JQuery та препроцесора SCSS. Кваліфікаційна робота освітнього ступеня «магістр». Тернопільський національний технічний університет імені Івана Пулюя, кафедра програмної інженерії, група СПм-61, спеціальність 121 «Інженерія програмного забезпечення». Тернопіль, 2025. Сторінок 64, рисунків 11, таблиць 1, додатків 2, бібліографічних посилань 37.

Метою роботи є розробка автоматизованої веб-системи, яка забезпечує ефективне управління даними, автоматизацію внутрішніх процесів та надання зручного, адаптивного інтерфейсу для користувачів.

Об’єктом дослідження є процес створення автоматизованої веб-системи із використанням CMS Opencart та сучасних IT технологій.

Предметом дослідження є методи та технології розробки веб-системи. Серед методів дослідження: аналіз вимог системи, макетування архітектури, тестування функціональних компонентів та розгортання системи на хостингу.

В даній кваліфікаційній роботі проведено аналіз вимог та проєктування сервісу. Розглянуто основні функціональні модулі, організацію обробки даних та механізми взаємодії з користувачем, а також інтеграцію для підвищення продуктивності та автоматизації системи. Реалізовано архітектуру MVC з використанням PHP, JQuery, SCSS, Bootstrap, MySQL. Наведено опис процесу тестування, впровадження системи та аналіз результатів, які підтверджують ефективність рішень. Також розглянуто питання охорони праці та безпеки, які забезпечують комфортні умови роботи для розробників. Робота демонструє повний цикл розробки — від аналізу вимог до впровадження.

Ключові слова: веб-система, CMS, OpenCart, PHP, JQuery, SCSS, Bootstrap, SEO-оптимізація, MySQL, MVC-архітектура.

ABSTRACT

Filyk Viktor Vasylovich. Development of an automated web system using the OpenCart CMS, PHP programming language, jQuery client library, and the SCSS preprocessor. Master's degree thesis. Ternopil Ivan Pului National Technical University, Department of Software Engineering, group SPm-61, specialty 121 «Software Engineering» Ternopil, 2025. 64 pages, 11 figures, 1 tables, 2 appendices, 37 bibliographical references.

The aim of this work is the development of an automated web system that ensures efficient data management, automation of internal processes, and provides users with a convenient, adaptive interface.

The object of the study is the process of creating an automated web system using CMS OpenCart and modern IT technologies.

The subject of the study is the methods and technologies for web system development. The research methods include system requirements analysis, architecture prototyping, functional component testing, and system deployment on hosting.

In this qualification work, a requirements analysis and system design were carried out. The main functional modules, data processing organization, and user interaction mechanisms were examined, as well as the integration for improving system performance and automation. An MVC architecture was implemented using PHP, JQuery, SCSS, Bootstrap, and MySQL. The testing process, system deployment, and analysis of results confirming the effectiveness of the solutions are presented. Additionally, labor protection and safety issues were considered to ensure comfortable working conditions for developers. The work demonstrates the full development cycle - from requirements analysis to deployment.

Keywords: web system, CMS, OpenCart, PHP, JQuery, SCSS, Bootstrap, SEO-optimization, MySQL, MVC-architecture.

ЗМІСТ

ВСТУП..... 7
РОЗДІЛ 1. АНАЛІЗ ВИМОГ ДО ПРОГРАМНОЇ СИСТЕМИ..... 9
1.1 Аналіз предметної області..... 9
1.2 Вимоги до функціональності веб-системи..... 10
1.2.1 Каталог пропозицій з системою фільтрації..... 11
1.2.2 Процедура оформлення замовлення та кошик..... 12
1.2.3 Особистий кабінет та програма лояльності..... 13
1.2.4 Інтеграція з зовнішніми системами..... 15
1.2.5 SEO-оптимізація та маркетингові інструменти..... 15
1.3 Актори системи та варіанти використання..... 16
1.4 Нефункціональні вимоги до системи..... 19
РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОЇ СИСТЕМИ..... 21
2.1 Вибір процесу розробки та методології..... 21
2.2 Архітектура системи та вибір технологічного стеку..... 23
2.3 Діаграма класів системи..... 27
2.4 Діаграми діяльності системи..... 31
2.5 Проєктування бази даних..... 36
2.6 Розробка функціональних модулів системи..... 37
2.7 Реалізація інтерфейсу користувача (UI/UX)..... 41
2.8 Інтеграція з зовнішніми системами та сервісами..... 43
2.9 Реалізація SEO-оптимізації та маркетингових інструментів..... 44
РОЗДІЛ 3. ТЕСТУВАННЯ, ВПРОВАДЖЕННЯ ТА ПІДТРИМКА..... 46
3.1 План тестування системи..... 46
3.2 Види та методи тестування..... 48
3.3 Тестування функціональних модулів..... 50

3.4 Тестування інтерфейсу та адаптивності..... 51
3.5 Процес впровадження..... 52
3.6 Аналіз результатів впровадження..... 54
РОЗДІЛ 4. ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ.57
4.1 Охорона праці.....57
4.2 Забезпечення безпеки функціонування веб-системи під час НС мирного та воєнного часу..... 59
ВИСНОВКИ..... 63
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ..... 65
ДОДАТКИ.....69
Додаток А. Тези конференції..... 70
Додаток Б. Логічна та фізична модель бази даних..... 72

ВСТУП

В умовах активної цифровізації економіки важливу роль для розвитку сучасної високоефективної бізнес-діяльності відіграє використання сучасних інформаційних технологій, зокрема і веб-технологій. У межах даної роботи здійснюється розробка автоматизованої веб-системи з використанням CMS OpenCart 3, мови програмування PHP, клієнтської бібліотеки JQuery та препроцесора SCSS, а також Bootstrap та MySQL. Система орієнтована на автоматизацію ключових бізнес-процесів, підвищення зручності взаємодії з користувачами, оптимізацію внутрішньої логіки роботи та покращення загальної ефективності.

Актуальність теми кваліфікаційної роботи обумовлена необхідністю створення ефективних і безпечних автоматизованих веб-систем, які забезпечують зручну взаємодію користувачів із сервісом та надійне управління даними. У сучасних умовах швидкий доступ до інформації, адаптивний інтерфейс та автоматизація процесів стають критично важливими для підвищення продуктивності та оптимізації робочих процесів системи. Проєкт також націлений на забезпечення зручності використання та інтеграції з зовнішніми системами.

Мета дослідження полягає в розробці та впровадженні веб-системи. Щоб досягти цієї мети, сформульовано кілька задач:

- Провести аналіз предметної області.
- Проаналізувати вимоги до функціональності та нефункціональні вимоги до системи.
- Розробити архітектуру програмної системи на основі MVC-патерну з використанням OpenCart 3 та мови програмування PHP.
- Створити адаптивний інтерфейс за допомогою JQuery, SCSS для кращої зручності навігації на різних пристроях.
- Оптимізувати базу даних для надання швидкої роботи системи навіть з великим обсягом інформації.
- Інтегрувати систему з зовнішніми сервісами.

- Провести тестування системи та впровадження на продуктивному сервері, з подальшим моніторингом та підтримкою.

Об’єктом дослідження є процес створення автоматизованої веб-системи із використанням CMS Opencart та сучасних IT технологій. Предметом дослідження є методи та фронтенд і бекенд технології розробки веб-системи.

Наукова новизна роботи полягає у комплексному підході до розробки системи, яка поєднує інтеграцію із зовнішньою інформаційною системою та покращення користувацького досвіду завдяки використанню адаптивного дизайну й автоматизації бізнес-процесів. Важливим аспектом запропонованого рішення є автоматична синхронізація системи з CRM-системою, що забезпечує актуальність даних, узгодженість бізнес-логіки та підвищує загальну ефективність управління замовленнями і взаємодії з клієнтами.

Апробація результатів дослідження відбулася через впровадження системи в реальних умовах використання, а також через публікацію тез доповідей на науково-практичній конференції. Результати роботи були представлені на XIII науково-технічній конференції "Інформаційні моделі, системи та технології". На конференції було донесено різні особливості реалізації та інтеграції даної веб-системи.

Отже, ця кваліфікаційна робота розкриває важливі питання розробки автоматизованих веб-систем, пропонує ефективні технічні рішення та підтверджує їх практичну цінність через успішне впровадження та апробацію.

РОЗДІЛ 1. АНАЛІЗ ВИМОГ ДО ПРОГРАМНОЇ СИСТЕМИ

Аналіз вимог є основним етапом розробки автоматизованої веб-системи. Він визначає як функціональні так і нефункціональні характеристики майбутнього проєкту. Даний проєкт є по суті новою версією системи, яка створюватиметься з чистого листа на основі нового рушія Opencart 3.

Розділ систематизує вимоги, які формують основу для проєктування, реалізації та тестування програмного продукту. Сформульовані вимоги дозволяють мінімізувати ризики, а також оптимізувати ресурси та забезпечити взаємодію між всіма компонентами системи.

1.1 Аналіз предметної області

Сучасний ринок автоматизованих веб-систем характеризується великим масивом даних та достатніми вимогами до якості обслуговування. Такі системи забезпечують централізоване управління даними, автоматизацію типових операцій, скорочення часу на обробку запитів і зниження впливу людського фактору. Завдяки використанню веб-технологій доступ до системи можливий з будь-якого пристрою, підключеного до мережі Інтернет, що підвищує зручність та масштабованість рішень.

Предметна область даної кваліфікаційної роботи охоплює процеси створення та функціонування автоматизованих веб-систем, орієнтованих на обробку структурованих даних, управління контентом, виконання прикладних бізнес-операцій та забезпечення зручного користувацького інтерфейсу. До таких процесів належать зберігання й оновлення інформації, обробка користувацьких дій, керування доступом, інтеграція з зовнішніми сервісами та підтримка безперервної роботи системи. Важливу роль у побудові подібних веб-систем відіграють системи управління контентом (CMS), які надають інструменти адміністрування та механізми розширення функціональності. Використання CMS

дозволяє зосередитись на адаптації та автоматизації прикладних процесів, не витрачаючи ресурси на реалізацію базових компонентів з нуля. У поєднанні з серверною мовою програмування PHP, клієнтськими бібліотеками JavaScript та сучасними засобами стилізації інтерфейсу такі системи забезпечують гнучкість, масштабованість і підтримку інтерактивної взаємодії з користувачем.

Автоматизація у веб-системах передбачає мінімізацію ручних операцій шляхом використання програмної логіки для виконання повторюваних дій, обробки запитів користувачів та управління станом системи. Це дозволяє підвищити ефективність роботи, зменшити кількість помилок і забезпечити стабільність функціонування системи навіть за великого навантаження. Окрема увага при цьому приділяється зручності користування, адаптивності інтерфейсу та швидкодії системи.

Отож, предметна область дослідження охоплює автоматизовані веб-системи як комплексні програмні рішення, що поєднують серверну логіку, клієнтську взаємодію, бази даних і інструменти управління контентом. У межах цієї роботи подальший аналіз і практична реалізація зосереджуються на розробці такої системи з використанням CMS OpenCart 3 та сучасного стеку веб-технологій, а конкретна прикладна реалізація слугує прикладом практичного застосування розробленого підходу.

1.2 Вимоги до функціональності веб-системи

Розроблювана автоматизована веб-система призначена для забезпечення повного циклу взаємодії користувачів із функціональними можливостями платформи: від отримання інформації та навігації до подальшої обробки результатів взаємодії. Вимоги до функціональності формуються на основі аналізу предметної області, потреб цільової аудиторії, очікувань замовника, а також з урахуванням сучасних підходів до побудови веб-систем електронної взаємодії.

Функціональні можливості системи орієнтовані на автоматизацію ключових бізнес-процесів, мінімізацію ручних операцій та забезпечення зручного й інтуїтивно зрозумілого користувацького інтерфейсу. Реалізація функцій передбачає чіткий розподіл ролей між користувачами системи, підтримку динамічної роботи з даними, а також інтеграцію з внутрішніми й зовнішніми сервісами.

Таким чином, сукупність визначених функціональних вимог формує універсальну автоматизовану веб-систему, орієнтовану на роботу з даними, взаємодію з користувачами та автоматизацію процесів обробки запитів. Практична реалізація зазначеного функціоналу доцільна у вигляді прикладного веб-проєкту. У межах даної кваліфікаційної роботи таким прикладом виступає веб-система типу інтернет-магазин «Buddvir», що дозволяє наочно продемонструвати роботу розроблених механізмів. Дана веб-система розробляється на базі практики під замовлення клієнта.

Нижче наведено опис перелічених основних функціональних блоків даної автоматизованої веб-системи, їх призначення та роль у забезпеченні стабільної й ефективної роботи програмного рішення в цілому.

1.2.1 Каталог пропозицій з системою фільтрації

Одним із найважливіших елементів у системі буде каталог пропозицій, який має бути максимально зручним для користувача. Так як веб-система міститиме велику кількість (понад 10 000) та складні характеристики кожної пропозиції, особлива увага приділяється системі фільтрації та навігації. Цей великий обсяг інформації важливо коректно сформувати та вивести для користувача [2-4]. Для реалізації даного функціоналу обрано модуль OCFilter [5]. Він дозволяє створювати гнучку фільтрацію за різними параметрами: ціною, брендом та іншими атрибутами.

Крім фільтрів, система підтримуватиме пагінацію та сортування за ціною, датою, алфавітом. Для покращення користувацького досвіду буде реалізовано швидкий пошук, що дозволяє знаходити пропозиції вже після введення перших символів назви.

Ще важливою особливістю каталогу товарів буде інтеграція з системою управління контентом (CRM), яка буде автоматично оновлювати дані у веб-системі кожного дня (вночі). Це усуне ризик відображення застарілої інформації.

Загалом, веб-система повинна забезпечувати інтуїтивно зрозумілий та зручний каталог пропозицій, оскільки саме він є ключовим елементом взаємодії користувача з системою. Каталог виступає основною точкою доступу до інформації, формує перше враження про веб-систему та безпосередньо впливає на ефективність подальшої роботи користувача.

1.2.2 Процедура оформлення замовлення та кошик

Процес оформлення замовлення – один з найкритичніших етапів взаємодії клієнта з прикладною реалізацією веб-системи. Він повинен бути простим та інтуїтивно зрозумілим. Тут буде використано модуль Smart Checkout, який дозволяє гнучко кастомізувати поля форми замовлення [6]. Щоб оформити замовлення, клієнт повинен вказати ім’я, прізвище, телефон та е-мейл адресу без необхідності реєстрації, що значно знижує відсоток відмов на цьому етапі. Для зручності клієнтів буде реалізовано збереження цієї інформації користувача в особистому кабінеті. З висновку аналізів конкурента це має прискорити повторні покупки.

Система підтримуватиме декілька варіантів доставки: самовивіз з магазинів, доставка через Nova Poshta та адресна доставка по Києву та області. Оплатити замовлення можна буде онлайн на рахунок або готівкою при отриманні.

Кошик є теж важливим елементом автоматизованої веб-системи, який поєднує етап вибору пропозицій із подальшим оформленням замовлення. Саме на

цьому етапі користувач приймає остаточне рішення щодо продовження взаємодії із системою, тому функціональність кошика має бути інтуїтивно зрозумілою та зручною. Коректна реалізація кошика забезпечує контроль над сформованим переліком позицій і створює позитивний користувацький досвід (UX), що безпосередньо впливає на ефективність роботи.

Окрему увагу приділено візуальній складовій як одному з ключових елементів взаємодії користувача. При натисканні на іконку кошика в хедері сайту відкриватиметься попап з переліком доданих товарів де клієнт може змінити кількість, видалити позицію чи перейти до оформлення. Також, у кошику мають відображатися рекомендації, які часто купують разом з тими, що вже є вибраними: це стимулюватиме додаткові продажі.

Кошик та процес оформлення замовлення є мабуть одними з ключових елементів автоматизованої веб-системи, оскільки саме вони безпосередньо забезпечують перехід користувача від перегляду пропозицій до завершення цільової дії. Від зручності та зрозумілості цих компонентів залежить можливість ефективної монетизації веб-системи.

1.2.3 Особистий кабінет та програма лояльності

Для зареєстрованих користувачів буде передбачено особистий кабінет, де вони можуть відстежувати історію замовлень, редагувати особисту інформацію, управляти адресами доставки, підпискою на новини.

Однією з ключових функцій авторизованих користувачів є програма лояльності, яка має стимулювати клієнтів робити повторні покупки. З власного досвіду розробки інших подібних прикладних реалізацій такої веб-системи, впевнений, що програма значно підвищить задоволеність користувачів системи під час її використання [7].

Програма працюватиме на основі накопичувальної системи знижок: чим більше клієнт витрачає на покупки, тим вищий відсоток знижки він отримує на

пропозиції, позначені стікером "Програма лояльності". Знижка розраховуватиметься автоматично на основі суми всіх оброблених замовлень користувача та буде застосовуватися до відповідних товарів у реальному часі. Наприклад, якщо клієнт оформив покупок на суму від 20 000 грн, він отримує 5% знижки на товари, що беруть участь у програмі. Акційні товари та товари з позначкою "Вигідна ціна" не мають підлягати дії програми лояльності, бо їхні ціни вже знижені. Для візуального відображення знижки буде використовуватися перекреслена стара ціна (синя) та нова ціна (червона), через це пропозиція буде більш привабливою.

В таблиці 1.1 наведено відсотки знижки для пропозицій з програмою лояльності в залежності від накопиченої суми покупок користувачем.

Таблиця 1.1 – Відсотки знижки програми лояльності

Накопичена сума покупок	**Відсоток знижки**
Більше 1 грн	2%
Більше 7000 грн	3%
Більше 20000 грн	5%
Більше 50000 грн	7%
Більше 100000 грн	Менеджери встановлюють самостійно

Крім того, в особистому кабінеті клієнт зможе переглядати статус замовлень, відстежувати повернення товарів, а також підписуватися на сповіщення про наявність пропозицій, яких немає на складі.

Наявність особистого кабінету суттєво покращить користувацький досвід для постійних клієнтів магазину, а впровадження програми лояльності додатково підсилить їхню зацікавленість та сприятиме повторним покупкам.

1.2.4 Інтеграція з зовнішніми системами

Щоб повністю автоматизувати бізнес-процеси прикладна система буде інтегрована з CRM-системою замовника, що дозволить синхронізувати дані про пропозиції, категорії, бренди, замовлення, тобто, ключові сутності системи. Вся ця інформація буде автоматично надсилатися до CRM, тут менеджери зможуть обробляти її через зручний інтерфейс. Нажаль, згідно договору із замовником системи, назва та суть роботи CRM не розголошуються.

Також передбачено реалізацію зворотної синхронізації для замовлень: інформація буде оновлюватися автоматично щодня за допомогою cron-завдань. Це дозволить підтримувати актуальність даних у системі та забезпечить своєчасне відображення інформації для користувачів та адміністрації. Ще це сприятиме більш ефективному контролю процесу обробки замовлень і підвищенню надійності роботи загалом.

Окрім CRM, система має бути інтегрована з Nova Poshta API через відповідний придбаний модуль [8]. Клієнт зможе обрати найзручніше відділення чи поштомат прямо на етапі оформлення замовлення: при кліку на відповідний селект, користувачу підтягнуться всі міста, відділення та поштомати Нової Пошти.

Інтеграція із зовнішніми сервісами дасть можливість автоматизувати різні процеси, які б в іншому випадку вимагали багато рутинної роботи для майбутніх менеджерів платформи.

1.2.5 SEO-оптимізація та маркетингові інструменти

Органічний трафік з пошукових систем є головним джерелом клієнтів для подібних веб-систем, тому особлива увага приділяється SEO-оптимізації [9]. Має бути реалізовано декілька ключових рішень:

- Генерація SEO-дружніх URL за допомогою модуля OCFilter, що дозволить створювати унікальні посилання для фільтрів.

- Налаштування Google Tag Manager, який дасть змогу відстежувати поведінку користувачів на сайті (кліки, перегляди, конверсії) та оптимізувати маркетингові кампанії.
- Автоматичне формування Sitemap, розбитий на декілька частин через велику кількість товарів (понад 10 000 позицій).
- Експорт каталогу товарів Google Merchant для відображення товарів у Google Shopping, щоб залучати додатковий трафік.
- Оптимізація зображень у форматі WebP, що прискорить завантаження сторінок та покращить позиції сайту в пошукових системах.

Крім того, буде реалізовано блокування доступу для користувачів з РФ та Білорусі (згідно з вимогами замовника), а також форми зворотного зв’язку такі як "Знайшли дешевше" та "Зворотний дзвінок". Вони допоможуть зберегти клієнтів та оперативно реагувати на їхні запити.

SEO-оптимізація у поєднанні з маркетинговими інструментами забезпечує систематичне залучення нових користувачів, підвищує їхню зацікавленість у сервісі та стимулює повторні відвідування. Такі практики дозволяють не лише розширювати аудиторію, але й формувати лояльну базу клієнтів. Це є основною метою оптимізаційних та маркетингових заходів.

1.3 Актори системи та варіанти використання

Прикладна реалізація автоматизованої веб-системи «Buddvir» буде орієнтована на дві основні категорії цільової аудиторії: фізичні особи (сегмент B2C) та оптові покупці (сегмент B2B). Кожна з цих груп має свої особливості поведінки і очікування від системи. Це вимагає правильного підходу до реалізації функціоналу.

Крім того, система передбачатиме взаємодію з кількома типами акторів, кожен з яких виконуватиме чітко визначені ролі та функції в межах бізнес-процесів. Такий підхід дозволить розмежувати рівні доступу, оптимізувати

роботу з даними та забезпечити коректне виконання операцій відповідно до повноважень кожної групи користувачів. Це сприятиме підвищенню керованості системи та її безпечному й стабільному функціонуванню.

Клієнт – основний актор, який буде взаємодіяти з системою. Клієнт зможе шукати товари, переглядати каталог товарів, реєструватися та авторизуватися через email або телефон, застосовувати фільтри для швидкого пошуку, додавати товари до кошика та оформляти замовлення з вибором зручного способу доставки і оплати.

Після покупки клієнт зможе відстежувати статус замовлення в особистому кабінеті. Також він зможе залишати відгуки про товари та ставити питання щодо продукції.

Для заохочення повторних звернень користувачів у веб-системі буде впроваджено програму лояльності, яка передбачає накопичення персональних знижок залежно від загальної суми коштів, витрачених на товари з відповідним стікером. Такий підхід дозволить підвищити зацікавленість клієнтів у регулярному використанні системи, сформувати довгострокові відносини з користувачами та стимулювати їх до здійснення наступних замовлень.

Для повноти розуміння того, як клієнти взаємодіятимуть з системою, розроблена діаграма варіантів використання (Use Case Diagram),, яка ілюструє їх основні дії при роботі з системою. Діаграма допомагає візуалізувати функціональні можливості системи та взаємодію клієнтів з різними модулями, що є важливим для проєктування зручного та ефективного інтерфейсу. Клієнт може бути як не зареєстрованим відвідувачем, так і авторизованим користувачем, який має доступ до додаткових функцій, описаних вище.

Діаграма зображена на рисунку 1.2.

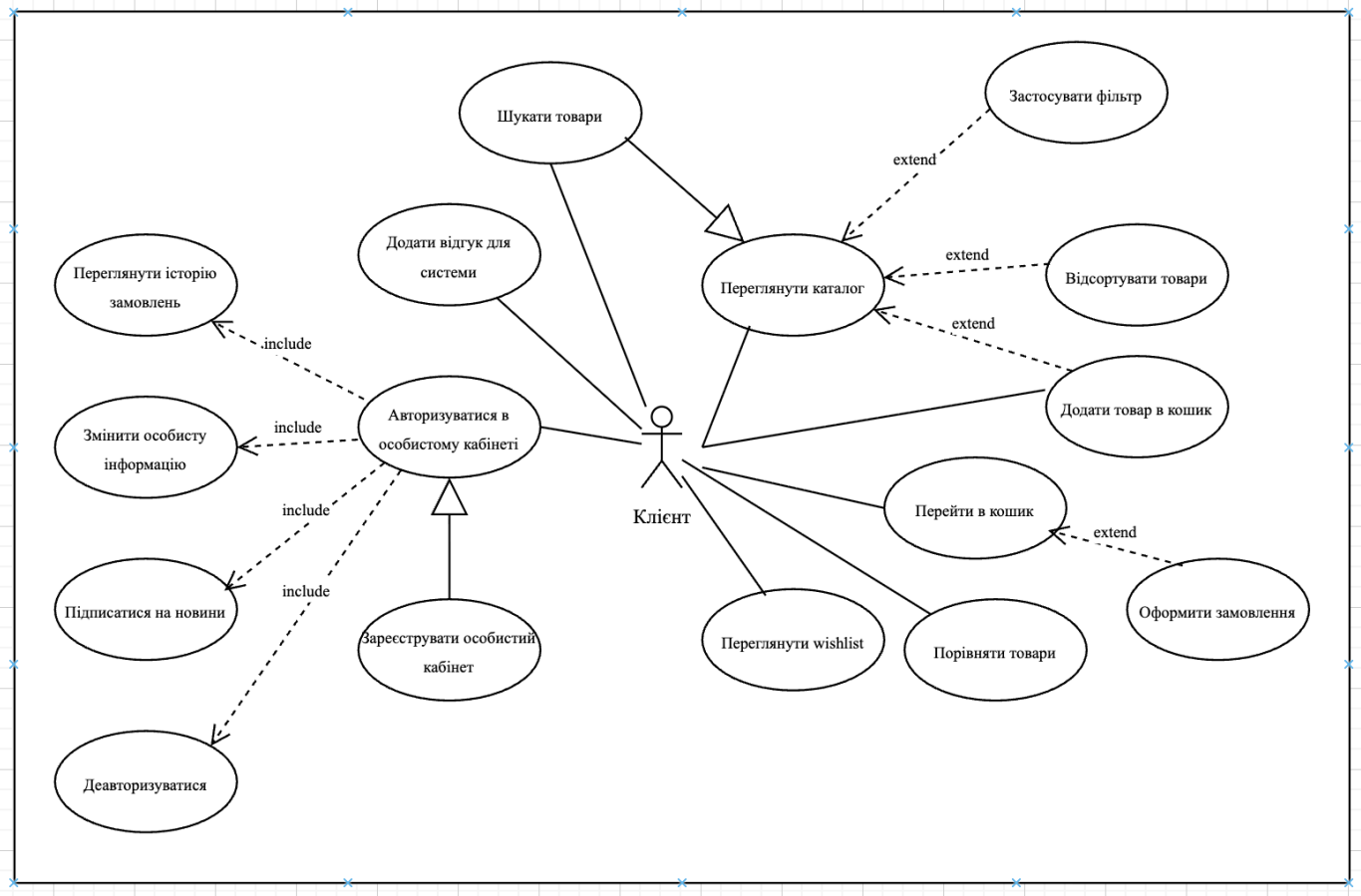

Рисунок 1.2 – Діаграма основних варіантів використання клієнта системи

Серед інших акторів системи є також адміністратор та менеджер.

Адміністратор відповідає за управління системою. До його обов’язків входить налаштування фільтрів, управління відгуками та питаннями клієнтів, а також налаштування інформаційних блоків веб-сайту (банерів, мегаменю).

Менеджер – актор, що працює з обробкою замовлень та взаємодіє з клієнтами. Менеджер буде отримувати сповіщення про нові замовлення: на електронну пошту та в панелі адміністратора. Він буде відповідати за обговорення умов доставки та оплати з покупцями. Менеджер також напряму працює з CRM-системою. Саме він вручну встановлюватиме індивідуальні знижки для оптових покупців або клієнтів з великою сумою замовлень через адмін-панель.

Для лаконічності записки, діаграми варіантів використання не було зроблено для адміністратора та менеджера, адже вони мають не таку велику кількість дій, а

також ці дії є майже однаковими для всіх подібних систем створених з використанням CMS Opencart 3.

Злагоджена робота реалізації веб-системи базуватиметься на чітко пропрацьованих сценаріях взаємодії між користувачами та платформою. Варіанти використання дають візуалізацію та опис основних процесів, які будуть відбуватися в системі, визначаючи, як актори будуть досягати своїх цілей за допомогою функціоналу платформи. Це дозволяє формалізувати вимоги до системи, виявити можливі виняткові ситуації та забезпечити узгодженість між очікуваннями користувачів і реалізованими програмними рішеннями, що, у свою чергу, сприяє підвищенню якості, надійності та зручності подальшої експлуатації веб-системи.

1.4 Нефункціональні вимоги до системи

Для забезпечення стабільної та ефективної роботи розроблювана веб-платформа повинна відповідати основним нефункціональним вимогам. Насамперед, система має бути високопродуктивною та швидко завантажуватися, забезпечуючи безперебійну роботу навіть при великому навантаженні користувачів.

Оптимізація часу відповіді та швидкості завантаження сторінок є критичною, оскільки безпосередньо впливає на задоволеність користувачів та їхню готовність повторно використовувати систему. Крім того, висока продуктивність сприяє стабільності роботи при одночасному обробленні великої кількості запитів та мінімізує ймовірність виникнення помилок або зависань системи.

Система також повинна відповідати сучасним вимогам безпеки та захисту даних. Особлива увага приділяється захисту персональної інформації користувачів, що є обов'язковою вимогою замовника. Реалізація таких заходів передбачає шифрування даних, контроль доступу до конфіденційної інформації та

впровадження механізмів безпечного зберігання та передачі даних. Це забезпечує не лише відповідність правовим нормам, але й підвищує довіру користувачів до веб-системи та безпечність її експлуатації.

Для цього реалізуються наступні заходи:

- Дотримання вимог щодо обробки персональних даних.
- Блокування доступу для користувачів з РФ та Білорусі на рівні IP-фільтрації.
- Використання HTTPS для шифрування даних, що передаються між клієнтом та сервером.
- Регулярне оновлення OpenCart та модулів для усунення вразливостей.

В реаліях сучасного світу веб-девайсів платформа має також надавати можливість для коректного відображення на різних пристроях (десктопи, планшети, смартфони). Для цього буде використано Bootstrap для створення гнучкої сітки. Даний фреймворк адаптується під розмір будь-якого екрану [10]. Також буде використовуватися препроцесор SCSS для управління CSS стилями за допомогою медіа-запитів.

Дотримання нефункціональних вимог – це ключ до успіху автоматизованої веб-системи. Якщо оптимізувати продуктивність, подбати про безпеку даних, забезпечити адаптивність для різних пристроїв і правильно пропрацювати SEO-оптимізацію, це допоможе підвищити задоволеність користувачів, збільшити конверсію та покращити позиції у пошукових системах. Всі ці заходи створюють надійний фундамент для стабільного функціонування системи в майбутньому.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОЇ СИСТЕМИ

Розробка і проєктування програмної веб-системи – це етап, який визначає структуру, функції та технічні деталі проєкту. У цьому розділі знаходиться опис методів проєктування, вибір технологічного стеку, а також те, як реалізуються основні модулі для забезпечення стабільної роботи, масштабованості та зручності у використанні системи. Проєктування допомагає оптимізувати процеси розробки, забезпечує гнучкість для майбутніх оновлень і відповідає вимогам як бізнесу, так і кінцевих користувачів. Особливу увагу приділено архітектурним рішенням, інтеграції з зовнішніми сервісами та дотриманню сучасних стандартів розробки.

2.1 Вибір процесу розробки та методології

Розробка веб-системи базується на гібридному підході, що поєднує елементи Agile та Kanban [11]. В свою чергу, вибір зумовлений гнучкими вимогами проєкту, необхідністю швидко реагувати на зміни в пріоритетах замовника, а також поступовим впровадженням функціоналу.

Хоча і немає дотримання традиційної Agile-методології з чітко визначеними спринтами, процес розробки залишається ітеративним та адаптивним. Завдання поділяються на окремі блоки і передаються на виконання поетапно, враховуючи їх пріоритетність. Це надає можливість:

- Швидко реагувати на зміни вимог (наприклад, якщо замовник вирішує, що певна функція потрібна терміново, вона переноситься в перший пріоритет).
- Поступово впроваджувати функціонал, що дає змогу тестувати та коригувати окремі компоненти системи на ранніх етапах.
- Зменшувати ризики, пов’язані з тривалим циклом розробки, оскільки кожен блок реалізовується та перевіряється окремо.

Такий підхід можна охарактеризувати як Kanban-подібний, оскільки:

- Задачі розбиваються на окремі блоки робіт.

- Виконання задачі відбувається в порядку пріоритетності, з можливістю перемикання на більш важливі завдання.
- Відсутні жорсткі спринти, але є постійний потік задач, які поступово реалізуються та впроваджуються.

Вважаю, що цей метод є ефективним для проєкту, де вимоги можуть змінюватися, а функціонал потрібно впроваджувати поетапно, без очікування завершення всього проєкту.

Розробку системи можна умовно розділити на кілька основних етапів, які відповідають класичному життєвому циклу програмного забезпечення, проте з урахуванням гнучкості та ітераційного підходу:

- Аналіз вимог та дизайну.
- Реалізація функціоналу.
- Поступове впровадження.

На етапі аналізу проводиться детальний аналіз дизайну веб-системи (макету) в дизайн-системі та функціональних вимог замовника. Особлива увага приділяється:

- Структурі каталогу товарів.
- Логіці роботи кошика та оформлення замовлення.
- Інтеграції з зовнішніми системами.
- SEO-оптимізації та адаптивності.

Даний етап дозволяє чітко зрозуміти, які саме компоненти системи потрібно реалізовувати та зрозуміти в якій саме послідовності.

Після аналізу починається реалізація окремих блоків системи. Спочатку задачі розбиваються на логічні блоки (наприклад, реалізація фільтрів OCFilter, розробка особистого кабінету). Пріоритетні задачі (наприклад, критичні помилки або функції) виконуються в першу чергу. Реалізація відбувається ітеративно: після завершення одного блоку він передається на перевірку, а розробка переходить до наступного. Постійна комунікація з замовником дозволяє оперативно вносити корективи та уточнювати деталі реалізації.

Далі відбувається перехід до поетапного впровадження функцій, які дозволяють тестувати окремі компоненти на ранніх стадіях. Це дає змогу швидко виправляти помилки без ризику зламати всю систему, а також запускати певні функції, такі як програма лояльності чи інтеграція з CRM. Це допомагає замовнику швидше отримувати результати.

Для організації робочого процесу використовується власна CRM-система агентства. Вона дозволяє відстежувати статуси кожної задачі: в роботі, на перевірці чи завершено. Також можна встановлювати пріоритети для різних блоків робіт, зберігати історію змін та коментарі від замовника, що спрощує спілкування. Крім того, система допомагає контролювати терміни виконання та оперативно реагувати на зміни вимог. Хоча точна назва та функціонал цієї CRM-системи залишаються в таємниці, вдале використання цього інструменту дозволяє ефективно керувати проєктом, зберігаючи прозорість процесів.

Вважаю, що саме такий гнучкий процес розробки та обрані методології є максимально ефективними для даного проєкту з урахуванням усіх його ключових нюансів і особливостей. Гнучкий підхід дозволяє швидко адаптуватися до змін вимог, оперативно реагувати на зворотний зв’язок від замовника та поетапно вдосконалювати функціональність системи.

2.2 Архітектура системи та вибір технологічного стеку

Виконання роботи над даною реалізацією веб-системи базується на архітектурі MVC (Model-View-Controller). Ця архітектура є по-замовчуванню для платформи OpenCart 3. Вона дозволяє розділити бізнес-логіку програми, представлення даних та обробку запитів, що спрощує подальшу підтримку, масштабування та редагування [12].

Архітектурний паттерн MVC чітко розділяє компоненти системи на три основні частини:

- Model (Модель).
- View (Представлення).
- Controller (Контролер).

Model займається обробкою даних та бізнес-логікою. У OpenCart моделі представлені класами, які співпрацюють із базою даних MySQL. Наприклад, ModelCatalogProduct працює з товарами, а ModelAccountCustomer з користувачами. Моделі обробляють запити до бази даних, виконують валідацію і повертають дані у зручному форматі для подальшої роботи.

View відповідає за показ даних користувачеві. У OpenCart для представлення використовуються шаблони Twig, які формують HTML-сторінки на основі даних, отриманих від контролерів. Шаблони надають гнучкість у дизайні та адаптують інтерфейс до різних пристроїв (завдяки Bootstrap і SCSS). Крім того, рівень представлення забезпечує відокремлення логіки від зовнішнього вигляду сторінок, що спрощує підтримку та подальшу модифікацію інтерфейсу. Використання шаблонів Twig дозволяє повторно застосовувати елементи дизайну, зменшувати дублювання коду та підвищувати читабельність структури шаблонів.

Controller відповідає за обробку запитів користувачів і забезпечує взаємодію між моделями та презентерами. Контролери в OpenCart обробляють HTTP-запити, викликають відповідні моделі для отримання даних, тоді передають ці дані в шаблони для відображення.

Наприклад, контролер ControllerCatalogProduct обробляє запити на перегляд товару, викликає модель для отримання деталей про товар і передає їх у відповідний шаблон.

Нижче на рисунку 2.1 наведено наочну діаграму того, як частини даної архітектури взаємодіють між собою.

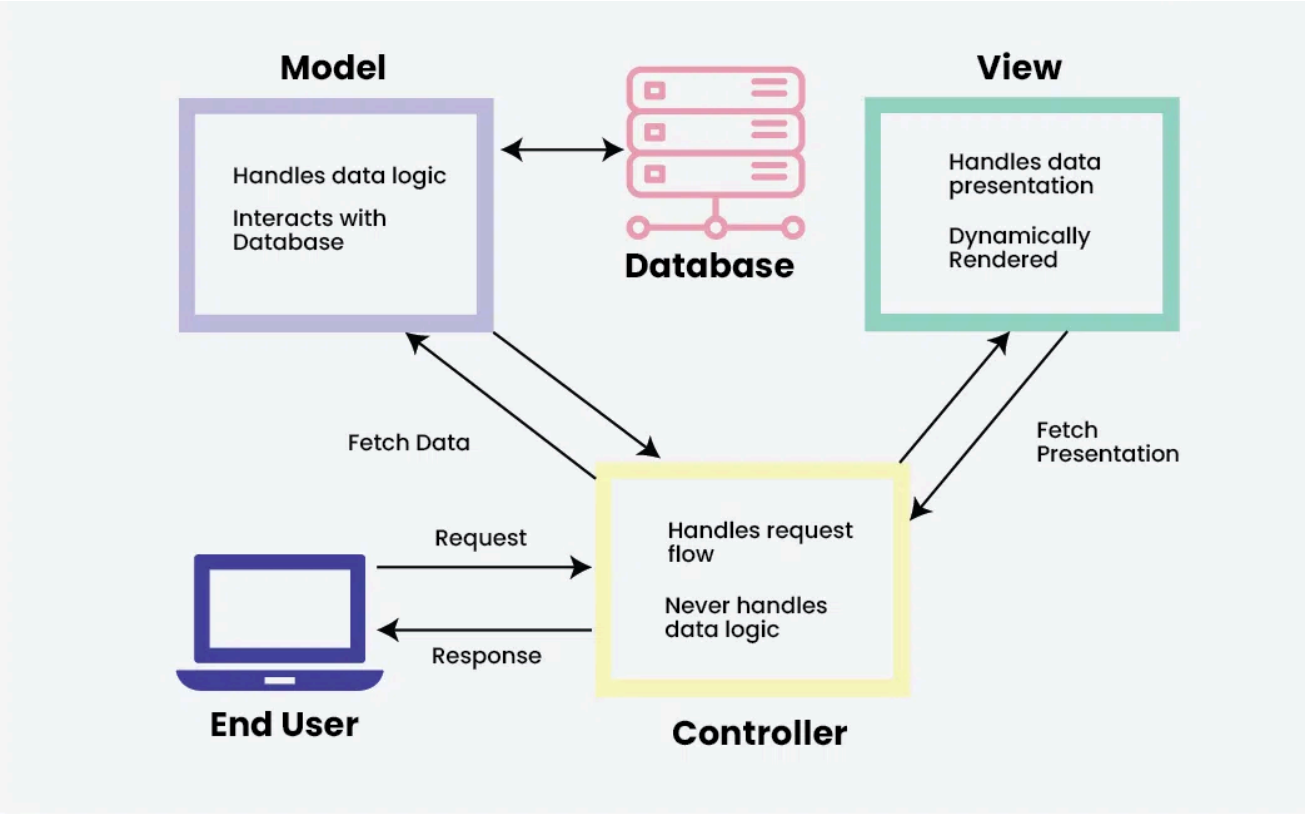

Рисунок 2.1 – Взаємодія частин MVC-архітектури

Серед переваг архітектури MVC можна виділити наступне:

- Розділення відповідальності: кожен компонент виконує свою функцію, що спрощує підтримку та модифікацію коду.
- Гнучкість: легко додавати нові функції або змінювати існуючі без ризику зламати всю систему.
- Масштабованість: дозволяє розширювати систему шляхом додавання нових модулів або зміни існуючих.

Далі обгрунтуємо вибір технологічного стеку. Вибір технологій для розробки системи зумовлений вимогами проєкту, продуктивністю та зручністю розробки.

Нижче наведено обгрунтування для кожного компонента, а саме:

- OpenCart 3
- PHP
- JQuery

- SCSS
- Bootstrap
- MySQL

OpenCart 3 – це безкоштовна та відкрита платформа для створення інтернет-магазинів, яка пропонує великий функціонал просто з коробки. Вона має гнучку архітектуру, що дозволяє без зусиль додавати нові модулі та змінювати існуючі [13-14]. Крім того, тут є велика спільнота розробників, а також багато модулів, таких як OCFilter і Smart Checkout. Платформа надає можливості адаптації під конкретні місцеві вимоги, наприклад, інтеграція з Nova Poshta.

Важливо також відмітити, що система не створюється з використанням чистого Opencart 3. Використовується надбудова над Opencart – OCStore рішення разом із OCDeals шаблоном [15-16]. OCStore – це популярна надбудова над Opencart 3, яка додає певний функціонал, відсутній в чистому Opencart. OCDeals – шаблон, написаний командою розробників з Дніпра. Цей шаблон також додає велику кількість модулів та функцій для системи: підписка на новини, стікери для товарів, розширений функціонал для блогу, тощо.

PHP – це серверна мова програмування, яка чудово підходить для створення веб-додатків і повністю сумісна з OpenCart. Вона широко використовувана та має безліч бібліотек, що спрощує розробку і підтримку. PHP дає змогу легко реалізовувати кастомну логіку, наприклад, програму лояльності [17-18].

JQuery – це клієнтська бібліотека, яка дозволяє працювати з HTML-документами, обробляти події та виконувати AJAX-запити. Використовується для динамічного оновлення сторінок, таких як попапи з кошиком і швидкий пошук товарів. Бібліотека кросбраузерна і легко інтегрується з існуючим кодом. З роками дана бібліотека показала свою ефективність та простоту в розробці [19-20].

CSS-препроцесор SCSS дає змогу використовувати змінні, міксини та вкладені правила, що значно спрощує написання стилів та їх підтримку.

Використовується для створення адаптивного дизайну та управління стилями в проєкті, зменшуючи дублювання коду та підвищуючи його читабельність [21].

Bootstrap – це фреймворк для фронтенд-розробки, який забезпечує адаптивність та кросбраузерність інтерфейсу. З його допомогою швидко розробляти кастомізовані компоненти, такі як сітка товарів і форми оформлення замовлень. Фреймворк містить безліч готових компонентів, що суттєво спрощує розробку [10].

Для бази даних використовується MySQL, яка відмінно підходить для зберігання даних системи, таких як товари, замовлення та користувачі. Ця СУБД показує високу продуктивність і надійність при роботі з великими обсягами даних. Вона також легко інтегрується з OpenCart і PHP [22].

На мою суб’єктивну думку, використання архітектури MVC разом з OpenCart 3 допоможе створити зручну, масштабовану та легку у підтримці автоматизовану веб-систему, яку можна реалізувати інтернет-магазину «Buddvir». Технологічний стек PHP, JQuery, SCSS, Bootstrap та MySQL дозволить легко розробляти систему та забезпечувати високу продуктивність.

2.3 Діаграма класів системи

Діаграма класів системи відображає архітектуру та взаємозв'язки між її компонентами. Вона ілюструє розподіл зон відповідальності між класами, які забезпечують роботу каталогу товарів, кошика, оформлення замовлення, особистого кабінету та інших функцій.

На рисунку 2.2 представлено діаграму ключових класів веб-системи. Загальна архітектура системи включає значну кількість класів, однак відображення всіх з них на одному рисунку є недоцільним через обмежений простір, тому на діаграмі наведено лише найбільш важливі та концептуально значущі елементи. Також в кожному класі відображено тільки деякі основні методи та атрибути.

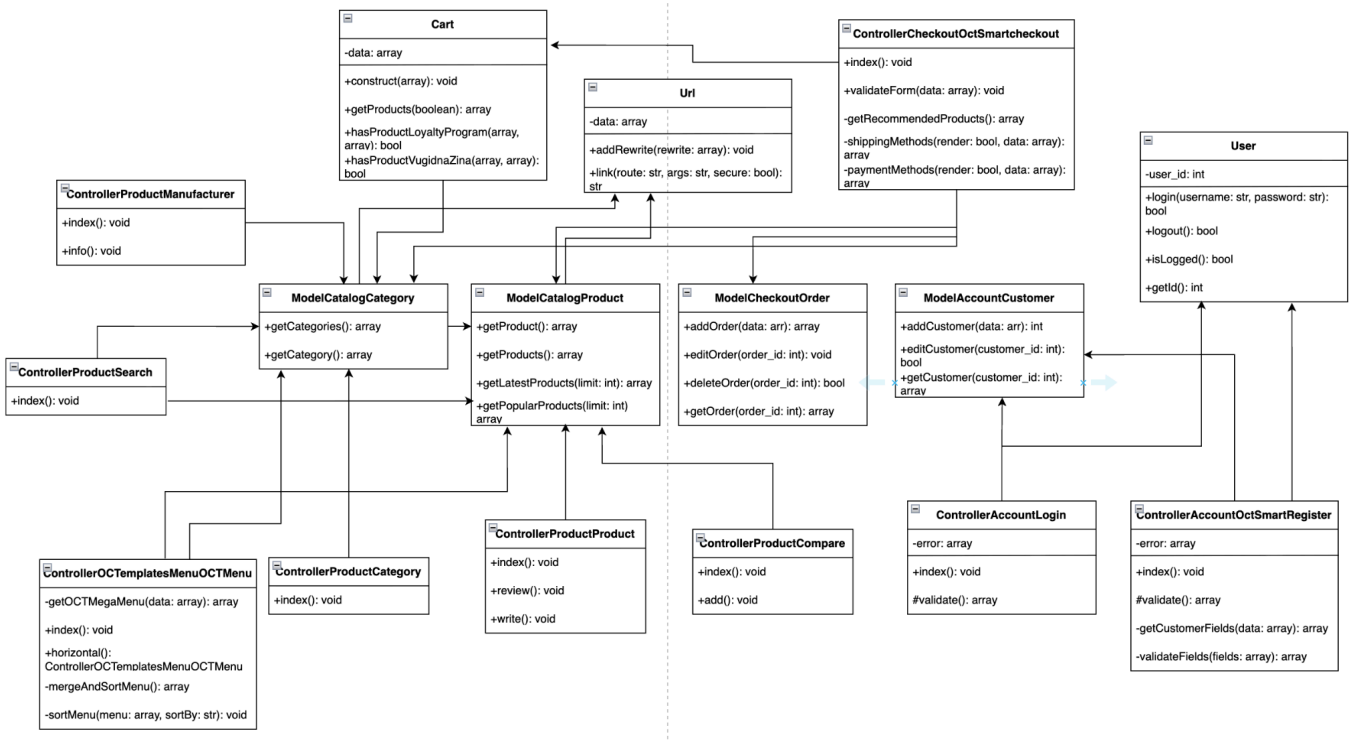

Рисунок 2.2 – Діаграма ключових класів системи

Клас Cart відповідає за управління кошиком. Даний клас зберігає інформацію про товари, які користувач додає, і обробляє ці дані. Основні методи цього класу дозволяють отримати список пропозицій у кошику, а також перевірити, чи бере пропозиція участь у програмі лояльності. Контролери оформлення замовлення активно використовують цей клас, як і при перевірці наявності товарів.

В свою чергу, клас Url, генерує SEO-дружні посилання. Він містить методи для додавання нових правил перезапису URL і формування правильних посилань на сторінки сайту. Це особливо важливо для покращення індексації сайту пошуковими системами та зручності навігації для користувачів.

Клас User відповідає за управління користувачами системи. Він забезпечує можливість автентифікації та завершення сеансу роботи, перевірку поточного стану входу користувача, а також реалізує інші функції, пов'язані з управлінням обліковими записами. Працює як центральний компонент для контролю доступу,

гарантує безпеку сесій та дозволяє інтегрувати користувацькі дії в різні частини системи.

Моделі каталогу взаємодіють з базою даних для отримання інформації про товари, категорії та виробників. Вони служать проміжним шаром між контролерами та базою даних, дозволяючи зручно отримувати доступ до даних.

Клас ModelCatalogCategory відповідає за роботу з категоріями товарів. Даний клас має методи для отримання списку категорій і деталей конкретної категорії. Цей клас використовують контролери категорій, виробників, пошуку, меню та сторінок окремих пропозицій.

Клас ModelCatalogProduct забезпечує роботу з пропозиціями. Серед його методів є способи отримати деталі конкретної пропозиції, список пропозицій, новинки та популярні товари. Цей клас активно використовують контролери сторінок товарів, категорій, пошуку, порівняння товарів і оформлення замовлень.

Клас ModelCheckoutOrder займається операціями з замовленнями. Клас відповідає за створення, редагування, видалення та отримання інформації про замовлення. Цей клас використовує контролер оформлення замовлення для обробки даних про замовлення клієнтів.

Клас ModelAccountCustomer відповідає за роботу з клієнтами. Він містить методи для додавання нового клієнта, редагування даних клієнта та отримання інформації про клієнта. Цей клас використовується контролерами авторизації, реєстрації та оформлення замовлення.

Контролери каталогу та товарів відповідають за обробку запитів користувачів і формування сторінок, які відображаються клієнтам. Вони взаємодіють з моделями для отримання потрібних даних і передають їх у шаблони для відображення.

Клас ControllerProductCategory відповідає за сторінку категорії товарів. Клас використовує моделі категорій та товарів для отримання даних та формування сторінки з товарами певної категорії.

Клас ControllerProductProduct управляє сторінкою конкретного товару. Даний клас відповідальний за відображення детальної інформації про пропозицію, включаючи опис, характеристики, відгуки та можливість додавання до кошика. Цей контролер також дозволяє залишати відгуки про товар.

Клас ControllerProductCompare відповідає за сторінку порівняння товарів: дає можливість користувачам додавати товари до списку порівняння й переглядати їх характеристики поруч, що спрощує вибір.

Клас ControllerProductSearch відповідає за сторінку пошуку. Клас обробляє запити користувачів на пошук товарів за ключовими словами та демонструє результати пошуку.

Клас ControllerProductManufacturer відповідає за сторінку виробників. Даний клас відображає інформацію про виробників і товари, які вони випускають.

Контролери шаблонів і меню забезпечують формування елементів інтерфейсу, таких як навігаційні меню, що важливо для зручності користувачів.

Клас ControllerOCTemplatesMenuOCTMenu генерує мегаменю, яке використовуються в шапці веб-системи. Він відповідає за формування горизонтального меню, об'єднання та сортування пунктів меню, а також генерацію мегаменю з категоріями та товарами. Цей контролер активно використовує моделі категорій і товарів для отримання даних.

Контролери авторизації та особистого кабінету забезпечують роботу з обліковими записами користувачів, включаючи вхід, реєстрацію й управління особистою інформацією.

Клас ControllerAccountLogin контролює процес входу користувача в систему. Він містить методи для перевірки облікових даних і авторизації користувача. Клас ControllerAccountOctSmartRegister відповідає за реєстрацію нових користувачів. Перевіряє введені дані, створює новий обліковий запис і автоматично авторизує користувача після успішної реєстрації. Цей контролер використовує модель клієнта для збереження даних про нового користувача.

Клас ControllerCheckoutOctSmartcheckout є головним контролером сторінки оформлення замовлення. Клас відповідає за обробку даних кошика, формування форми оформлення замовлення, перевірку введених даних і створення замовлення. Контролер взаємодіє з класами Cart, ModelCatalogProduct, ModelCheckoutOrder, ModelAccountCustomer та Url, щоб забезпечити коректну роботу процесу оформлення замовлення.

Діаграма класів системи відображає архітектуру, де контролери відповідальні за логіку сторінок і обробку запитів, моделі забезпечують доступ до даних, а сервісні класи виконують допоміжні функції. Така структура дозволяє розподілити відповідальність між компонентами системи. Діаграма показує основні класи, які визначають логіку роботи каталогу, кошика, оформлення замовлення та особистого кабінету, проте не охоплює всі класи системи, а лише ключові компоненти, які необхідні для розуміння архітектури.

2.4 Діаграми діяльності системи

Діаграми діяльності (Activity Diagram), представлені в цьому розділі, візуалізують ключові процеси взаємодії користувачів з автоматизованою веб-системою «Buddvir», демонструють логічну послідовність дій від початкового етапу до завершення. Ці діаграми не тільки ілюструють технічну реалізацію процесів, але й підкреслюють зручність та інтуїтивність інтерфейсу.

Перша діаграма "Процес взаємодії з каталогом" показує різні можливості взаємодії користувача із категоріями пропозицій веб-системи. Діаграма ілюструє основні сценарії поведінки відвідувачів при роботі з каталогом: перегляд, навігацію між підкатегоріями, використання фільтрів для пошуку потрібних позицій, сортування за різними параметрами, перегляд детальної інформації про обрані позиції на відповідних картках. Друга діаграма «Процес оформлення замовлення» відображає сценарій, де потрібно ввести дані, вибрати спосіб доставки й оплати, а також перевірити правильність введеної інформації. Вона

демонструє, як система адаптується до потреб користувачів, пропонуючи різні варіанти доставки та оплати. Це зменшує ризик помилок і підвищує довіру клієнтів, адже вони можуть бути впевнені в правильності обробки своїх замовлень.

Перша діаграма «Процес взаємодії з каталогом» відображає багаторівневу структуру навігації користувача веб-системою. Початковою точкою взаємодії виступає мегаменю, розміщене в шапці сайту. При виборі конкретної категорії користувач потрапляє на спеціалізовану сторінку каталогу, де реалізовано ієрархічну структуру з можливістю переходу до підкатегорій. Система фільтрації, розташована в лівій частині інтерфейсу, дозволяє деталізувати пошук за множиною параметрів. Застосування обраних фільтрів відбувається через кнопку "Показати товари", при цьому передбачена можливість скидання всіх налаштувань через функцію "Очистити". Додаткові інструменти сортування та вибору кількості позицій на сторінці забезпечують гнучкість у способі відображення результатів.

Центральним елементом каталогу є grid-структура карток товарів. Безпосередньо з картки користувач може виконати базові операції: визначити кількість одиниць, додати товар до кошика, додати в список порівняння або зберегти позицію в закладках. Перехід на детальну сторінку товару розширює функціональні можливості, додаючи опції швидкого оформлення замовлення через "Купити в 1 клік", форму "Знайшли дешевше", а також інструменти для залишення відгуків та запитань.

Діаграму зображено на рисунку 2.3.

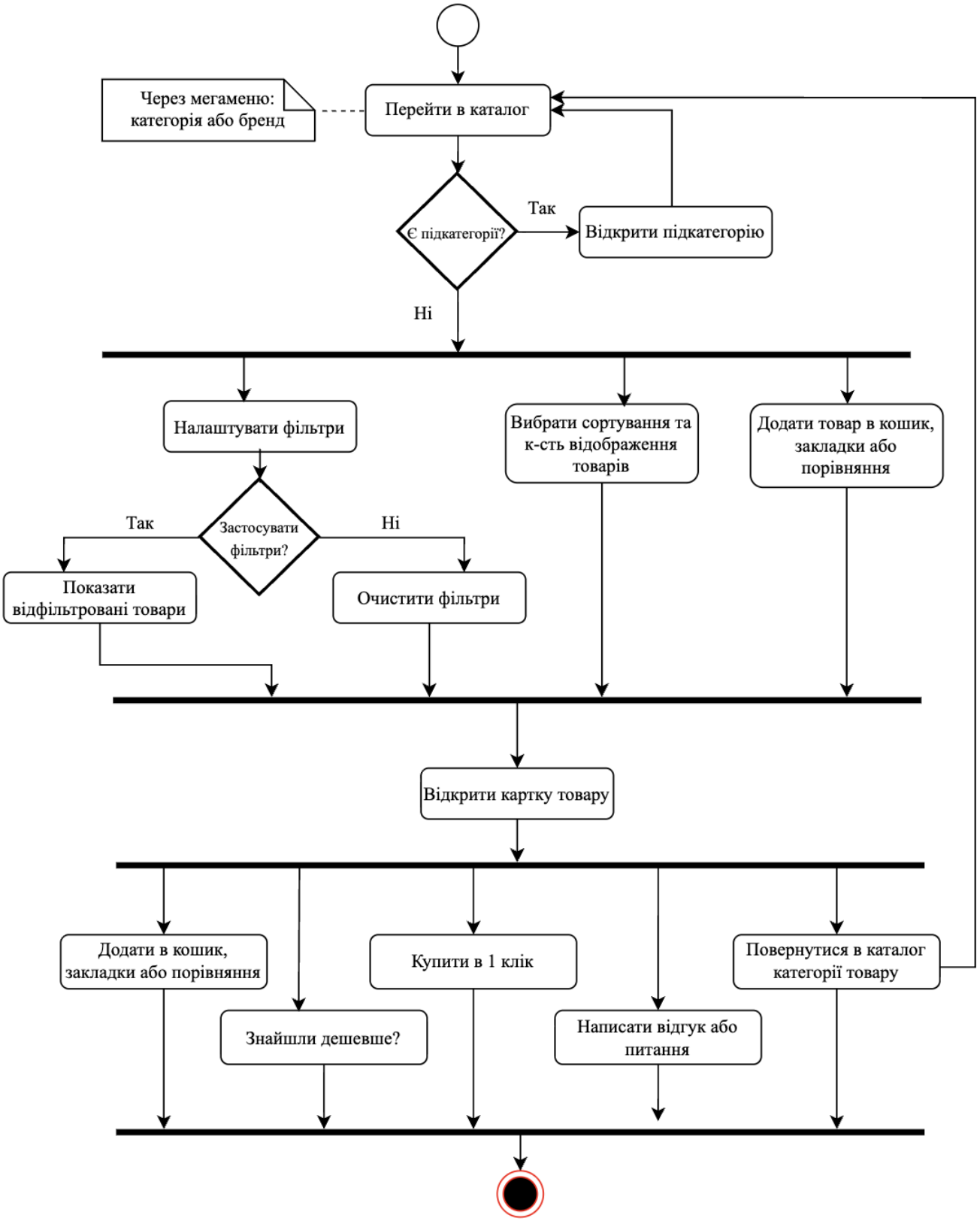

Рисунок 2.3 – Діаграма діяльності процесу взаємодії з каталогом

Далі наведено опис процесу оформлення замовлення в веб-системі. Діаграма ілюструє послідовність дій, які виконує користувач від моменту натискання кнопки «Оформити замовлення» до успішного завершення процесу або повернення на виправлення помилок валідації заповнених чи незаповнених полів.

На першому етапі система перевіряє, чи авторизований користувач. Якщо клієнт уже авторизований, він одразу переходить до вибору способу доставки. Проте, при бажанні ці дані можна відредагувати. Якщо користувач не авторизований, система пропонує йому ввести особисту інформацію (ім’я, прізвище, телефон, електронну пошту).

Етап вибору доставки є розгалуженням, оскільки клієнт може вибрати один з трьох варіантів доставки: Нова Пошта, доставка по Києву чи області, самовивіз. Кожен з варіантів забезпечує зручність для клієнта та дозволяє обрати найоптимальніший спосіб отримання товару. Після вибору доставки клієнт переходить до вибору способу оплати. Система пропонує декілька варіантів: оплата на рахунок, готівкою при отриманні товару. Цей етап дозволяє клієнту обрати найзручніший спосіб розрахунку, що сприяє покращенню користувацького досвіду.

Після заповнення всіх необхідних даних клієнт натискає кнопку «Оформити замовлення». На цьому етапі всі введені дані передаються на валідацію. Якщо всі поля заповнені коректно, замовлення успішно оформлюється, і клієнт отримує підтвердження про успішне оформлення на електронну пошту та в особистому кабінеті. Якщо виявлені помилки, система повертає користувача на етап редагування даних, щоб він міг виправити помилки та спробувати знову.

Дану діаграму зображено на рисунку 2.4.

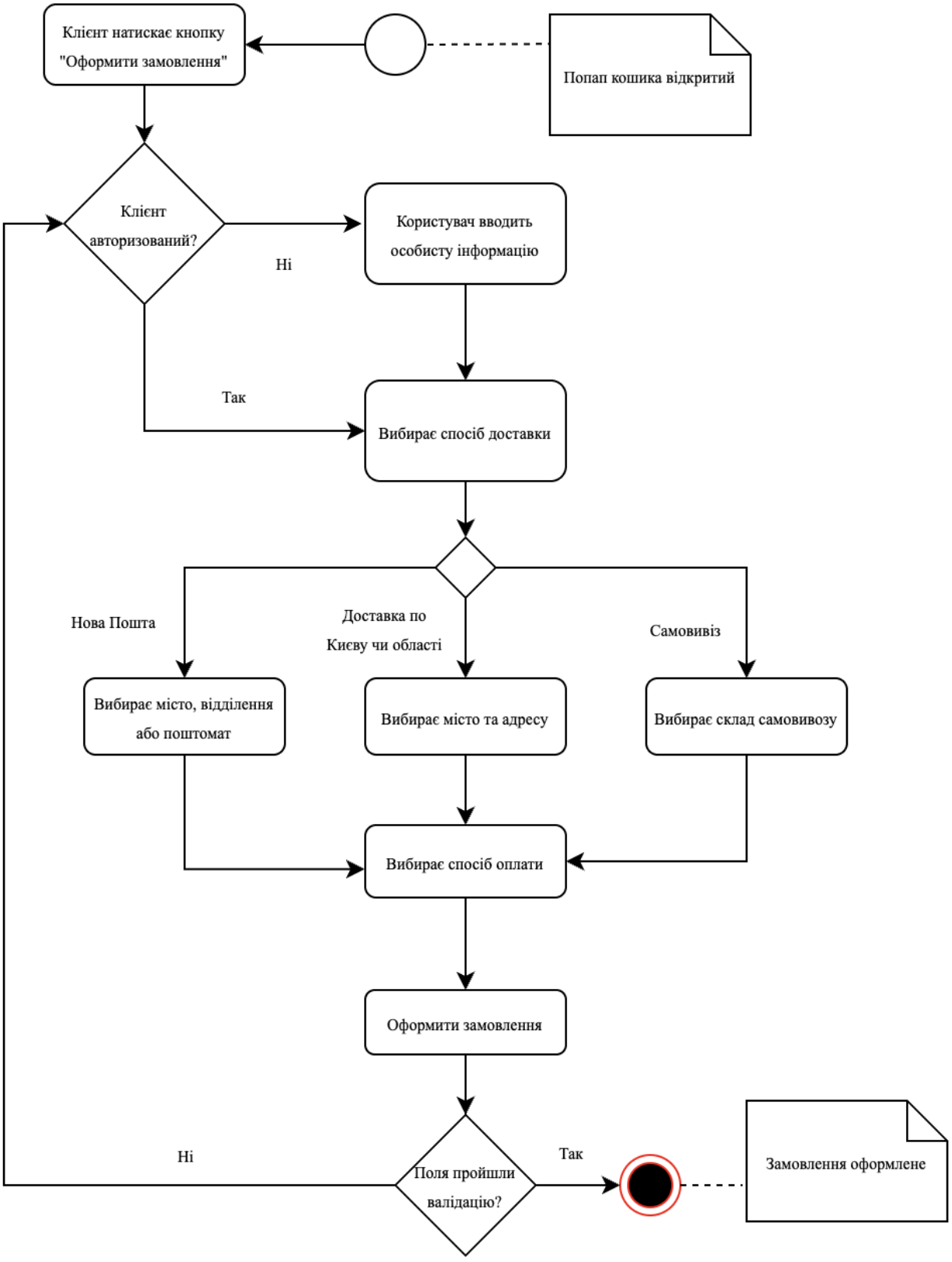

Рисунок 2.4 – Діаграма діяльності процесу оформлення замовлення

У підсумку, обидві діаграми показують логічну структуру процесів, що допомагають користувачам ефективно взаємодіяти із системою. Впровадження цих процесів у веб-системі забезпечує легкість, швидкість і надійність, що є важливими аспектами для успішного проєкту.

2.5 Проєктування бази даних

Проєктування бази даних для веб-системи відбувається з використанням стандартної структури OpenCart 3, яка вже містить основні таблиці для товарів, замовлень, користувачів та фільтрів. Ця структура має бути розширена та оптимізована для забезпечення роботи системи з урахуванням специфіки проєкту, зокрема великої кількості пропозицій (понад 10 000) та інтеграції з CRM-системою.

OpenCart 3 використовує реляційну базу даних MySQL, яка складається з декількох десятків таблиць, об’єднаних у логічні групи. Основні таблиці, які використовуються в проєкті, можна поділити на декілька категорій:

- Товари та категорії.
- Замовлення.
- Користувачі.
- Фільтри та атрибути.
- Інші важливі таблиці.

Діаграму основних сутностей бази даних зображено в додатку Б, рисунок Б.1.

Для забезпечення швидкої роботи системи з великою кількістю даних потрібно реалізувати декілька оптимізацій. Спочатку провести індексування таблиць: додати індекси до часто запитуваних полів (наприклад, product_id у таблицях oc_product, oc_product_to_category). Також потрібно провести індексування полів, які часто використовуються в WHERE-умовах та JOIN-запитах [22].

Використання реляційної бази даних MySQL із додатковим індексуванням ключових полів сприяє підвищенню загальної продуктивності системи, забезпечує швидший і більш ефективний доступ до даних, а також дозволяє зменшити навантаження на сервер під час обробки запитів.

2.6 Розробка функціональних модулів системи

Розробка функціональних модулів має на меті створити зручний та автоматизований інтерфейс для користувачів. Ключові модулі системи включають каталог товарів з фільтрами, кошик та оформлення замовлення, особистий кабінет з програмою лояльності, а також інтеграцію з CRM та Новою Поштою. Все це розробляється з урахуванням особливостей бізнес-процесів та вимог до продуктивності. Далі розглянемо логіку роботи та специфіку реалізації кожного з цих модулів.

Каталог товарів з фільтрами є основою системи, адже тут клієнти обирають те, що їм потрібно. Для зручності реалізується потужна система фільтрації на базі OCFilter. Вона дозволяє гнучко налаштовувати параметри пошуку за різними критеріями: ціна, виробник, розміри, колір, матеріал та інші атрибути. Інтерфейс фільтра потребує майже повного перероблення, тож залишиться лише частково логіка роботи.

Модуль OCFilter використовує декілька таблиць бази даних для зберігання інформації про фільтри та їх зв’язки з товарами. Клієнтські фільтри відображаються у вигляді блоків з можливістю вибору кількох значень для одного фільтра (наприклад, різні кольори або розміри). Після вибору потрібно натиснути «Показати товари», і система перезавантажить сторінку з відфільтрованими товарами. При виборі фільтрів система формує SQL-запит, що враховує всі обрані параметри, і повертає відфільтрований список товарів. Для покращення продуктивності використовуються індексовані поля в базі даних та кешування результатів через Redis.

Вигляд каталогу товарів зображено на рисунку 2.5.

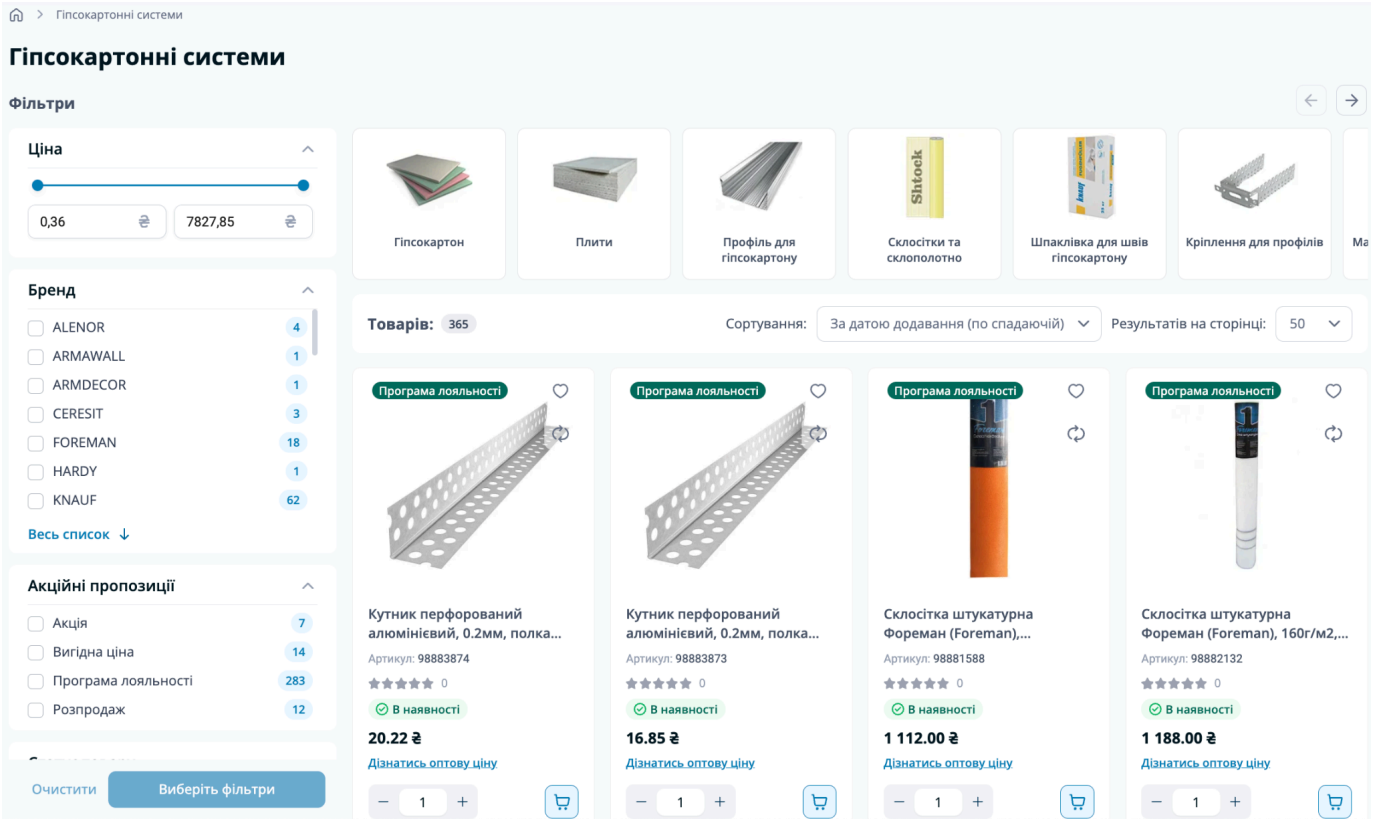

Рисунок 2.5 – Вигляд каталогу товарів веб-системи

Одна з ключових функцій OCFilter – генерація SEO-дружніх URL для сторінок з фільтрами, що покращує індексацію сторінок пошуковими системами та приносить додатковий органічний трафік.

Кошик працює динамічно, оновлюється без перезавантаження сторінки за допомогою JQuery і AJAX. Користувач може додавати, видаляти товари або змінювати їхню кількість безпосередньо у попапі, а система автоматично перерахує суму замовлення. Тут також відображаються рекомендації «Товари, які купують разом».

За основу для розробки функціоналу кошику було взято його реалізацію з шаблону OCDeals. Розроблений кошик зображено на рисунку 2.6.

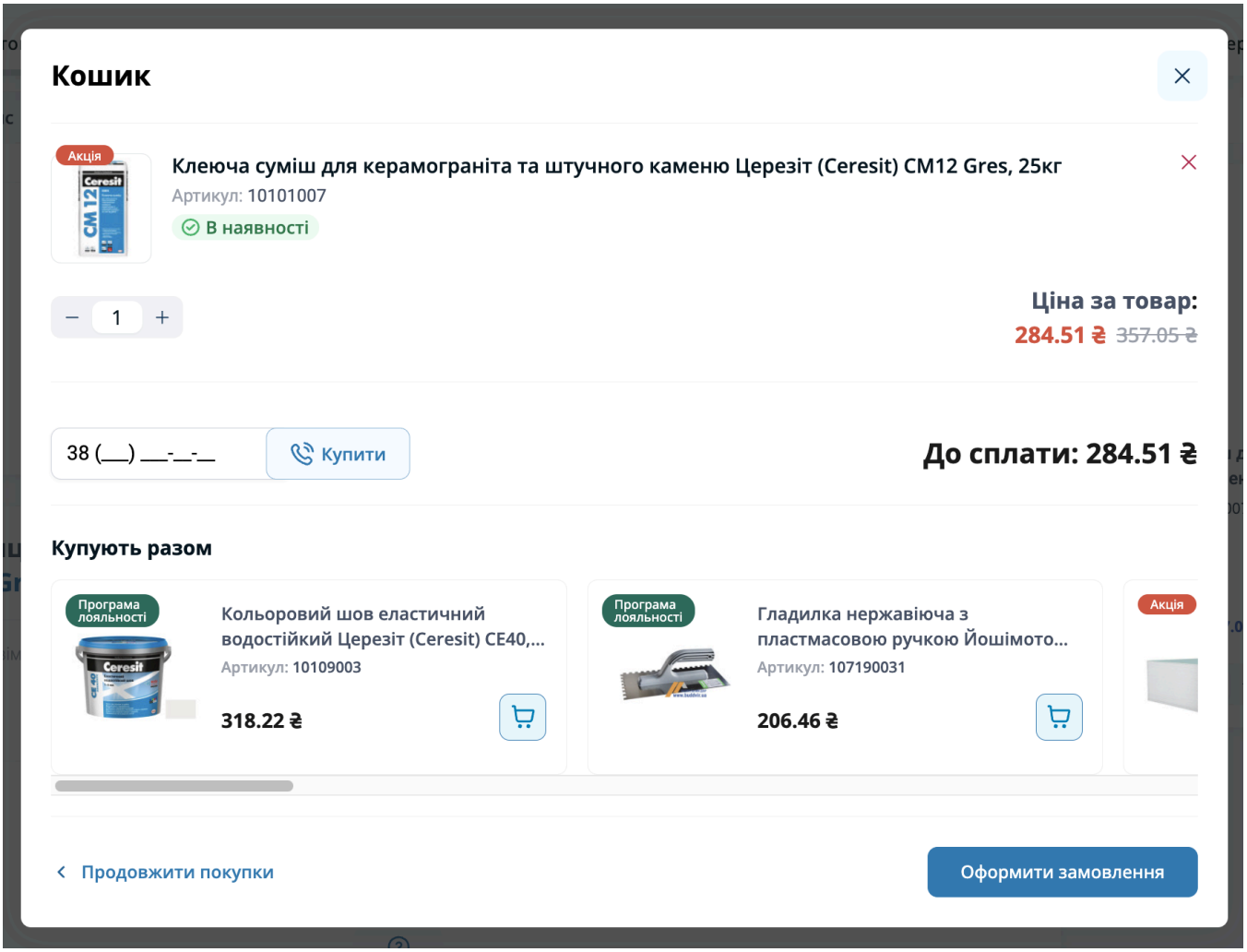

Рисунок 2.6 – Функціонал кошика веб-системи

Модуль оформлення замовлення має критичне значення для конверсії, адже саме тут клієнт вирішує, купувати товар чи ні. Smart Checkout спрощує цей процес, надаючи базовий функціонал для налаштування поля форми замовлення, а також реалізації функції «Купити в 1 клік».

Щоб забезпечити зручність оплати та доставки, система підтримує оплату за перерахунком і готівкою при отриманні. Клієнти можуть обрати найзручніше відділення Нової Пошти під час оформлення замовлення, що значно спрощує процес і покращує досвід користувачів. Сторінку оформлення замовлення зображено на рисунку 2.7.

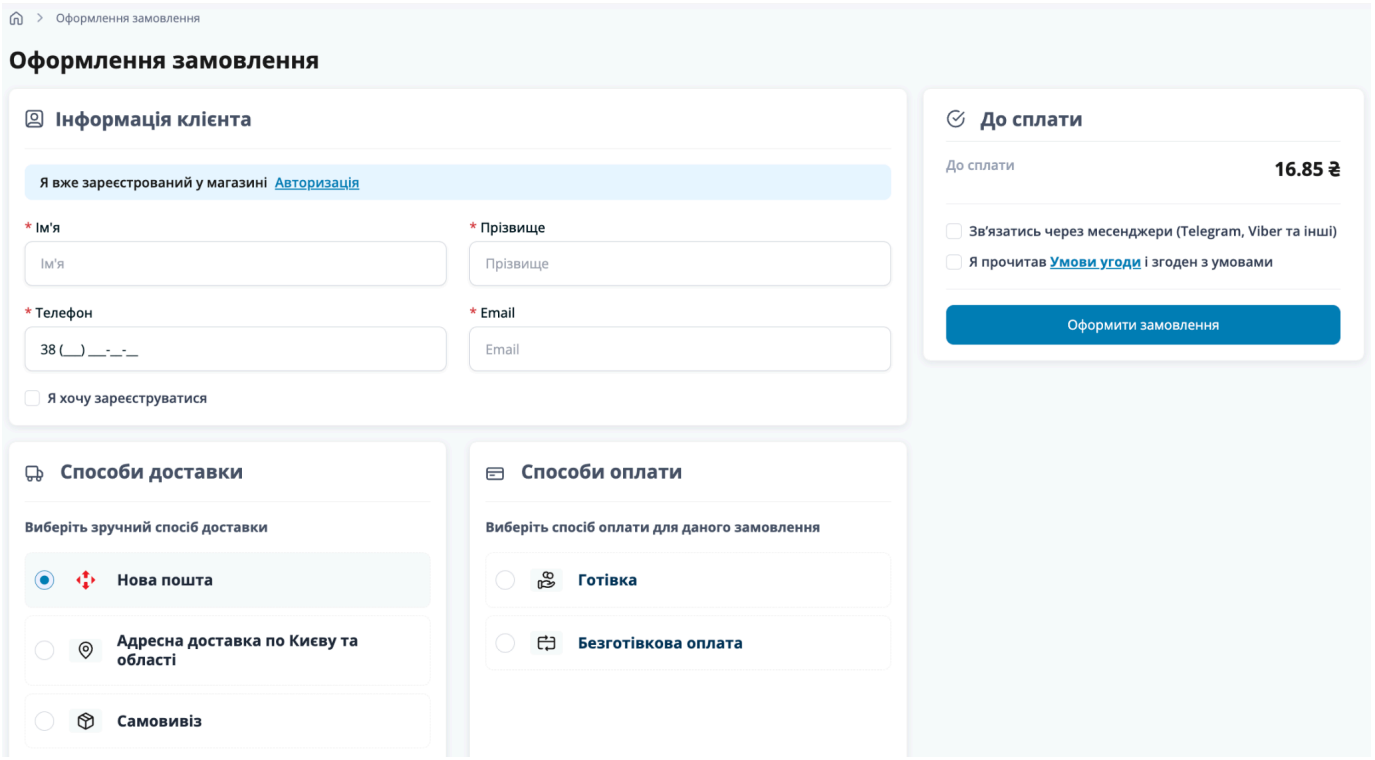

Рисунок 2.7 – Сторінка оформлення замовлення

Особистий кабінет – важлива складова системи, де зареєстровані користувачі можуть керувати своїми замовленнями, особистою інформацією та брати участь у програмі лояльності. Тут клієнти можуть відстежувати історію замовлень, редагувати дані, отримувати сповіщення про наявність товарів, на які підписались, а також слідкувати за статусом повернення [23].

Програма лояльності – це інструмент для заохочення повторних покупок. Система автоматично розраховує суму всіх замовлень клієнта та встановлює відповідний рівень знижки. Знижки автоматично застосовуються до товарів, що беруть участь у програмі лояльності (за винятком акційних пропозицій та пропозицій з позначкою «Вигідна ціна»). Менеджери можуть вручну встановлювати індивідуальні знижки для клієнтів, які роблять замовлення на суму понад 100 000 гривень (VIP користувачі). Для авторизованих користувачів в програмі лояльності товари відображаються з двома цінами: старою перекресленою та новою з урахуванням знижки. Це підкреслює вигоду для

клієнта та стимулює до покупки. Персоналізація програми є її особливістю – кожен клієнт бачить свої індивідуальні знижки.

Інтеграція з CRM-системою – ще один важливий елемент для автоматизації бізнес-процесів. Цей модуль дозволяє синхронізувати дані про товари, замовлення, виробників і категорії. Нові замовлення передаються до CRM, де менеджери можуть їх оперативно обробляти [24].

Щодо інтеграції з Новою Поштою, вона дає можливість обрати найзручніше відділення або поштомат при оформленні замовлення. Цей модуль створюється на основі іншого платного модуля – Nova Poshta API, який потрібно придбати та налаштувати під потреби веб-системи.

У цьому розділі розглянуто лише розробку основних модулів системи, однак фактична структура включає значно ширший набір компонентів, які забезпечують повноцінне функціонування платформи. Кожен із додаткових модулів прямо або опосередковано взаємодіє з базовими модулями, описаними вище, доповнюючи їхню функціональність та забезпечуючи цілісну роботу всієї системи.

2.7 Реалізація інтерфейсу користувача (UI/UX)

Інтерфейс розроблюваної реалізації веб-системи виконується з урахуванням сучасних стандартів UX/UI. Тут звертається увага на зручність навігації, адаптивність, візуальну привабливість і швидкість роботи. Головна мета – створити інтуїтивно зрозумілий та функціональний дизайн, який допоможе досягти максимальної конверсії і позитивного досвіду для користувачів на десктопах і мобільних пристроях [25-26].

Для розробки інтерфейсу використовується Bootstrap, SCSS, Swiper.js і Twig. Ці технології дозволяють створити гнучкий, адаптивний та легкий у підтримці інтерфейс. Однією з ключових вимог є коректне відображення на всіх пристроях від комп'ютерів до смартфонів. Для цього використовується Bootstrap 4, який

пропонує готову сітку, адаптивні компоненти і утиліти для швидкої розробки відповідальних інтерфейсів.

SCSS допомагає організувати й оптимізувати стилі, дозволяючи використовувати змінні для кольорів, відступів та шрифтів. Це спрощує підтримку та оновлення дизайну, а також дає можливість створювати міксини і функції для повторного використання коду. Стилі структуровані в окремі файли для різних компонентів, що значно покращує читабельність та підтримку коду [21].

Для реалізації інтерактивних слайдерів на головній сторінці сайту та на сторінках категорій була обрана бібліотека Swiper.js. Використання саме цієї бібліотеки пояснюється її високою гнучкістю та широкими можливостями налаштування, що дозволяє створювати різноманітні анімаційні ефекти та переходи між слайдами. Крім того, Swiper.js легко інтегрується з існуючою архітектурою веб-системи, підтримує різні типи взаємодії з користувачем.

На рисунку 2.8 зображено слайдер веб-системи, зроблений з допомогою Swiper.js.

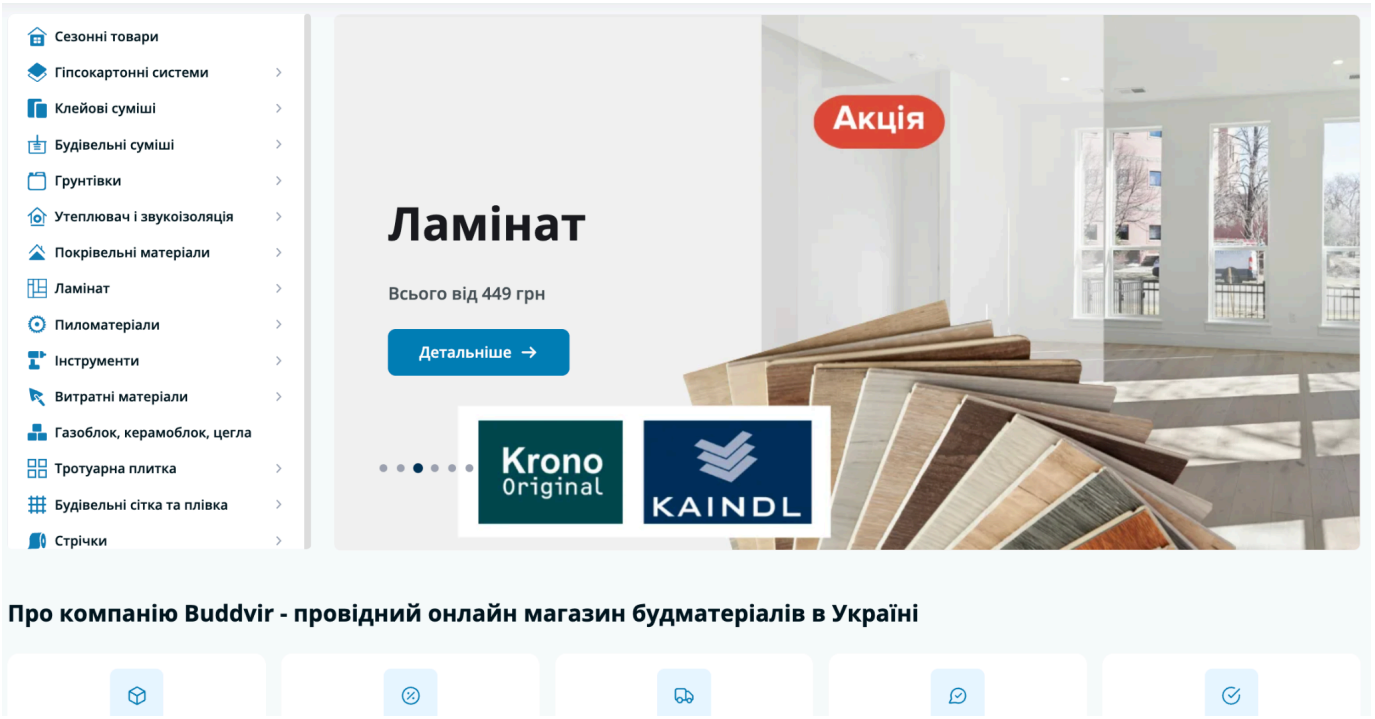

Рисунок 2.8 – Слайдер (банер) головної сторінки веб-системи

Для динамічного генерування HTML-сторінок застосовується шаблонізатор Twig, який за замовчуванням інтегрований у OpenCart версії 3. Twig дозволяє розділити логіку і представлення, що спрощує підтримку та модифікацію інтерфейсу.

В результаті, реалізація UI/UX для автоматизованої веб-системи поєднує Bootstrap 4 для адаптивності, SCSS для стильового оформлення та Swiper.js для слайдерів. Використання Twig гарантує розділення логіки та дизайну, підвищуючи безпеку і підтримку коду. Такий підхід забезпечує швидку роботу, інтуїтивну навігацію та високу конверсію.

2.8 Інтеграція з зовнішніми системами та сервісами

Інтеграція з зовнішніми системами – це одна з основ автоматизованої роботи платформи. У рамках проєкту реалізується інтеграція з CRM-системою, а також підключається API Nova Poshta для вибору відділень та поштоматів через відповідний модуль.

Завдяки інтеграції з CRM, обмін даними між веб-системою та внутрішніми процесами компанії замовника автоматизується. Це означає, що інформація про товари, категорії, бренди, атрибути товарів, замовлення синхронізується без ручного втручання. Якщо пропозиція закінчується, система може приховати його з каталогу або показати статус «Під замовлення». Після того як клієнт оформляє замовлення в системи, вся інформація відправляється до CRM, де менеджери стежать за статусом та вносять зміни. Клієнти завжди бачать актуальні дані про свої замовлення у своєму кабінеті [24, 27].

Для доставки реалізується інтеграція з API Nova Poshta, що дозволяє клієнтам вибрати відділення для самовивозу або вказати адресу для кур'єрської доставки. На етапі оформлення замовлення покупець може вибрати доставку до відділення або поштомату Нової Пошти, і система через API запитує список

відділень за введеною адресою. Також клієнт може вибрати самовивіз з одного зі складів компанії замовника.

Запити до API авторизуються токенами, які запобігають несанкціонованому доступу. Всі дані, що передаються між сайтом та зовнішніми системами, шифруються протоколом HTTPS. Чутливі дані, наприклад, особиста інформація клієнтів, не зберігається в відкритому вигляді і захищена від несанкціонованого доступу.

2.9 Реалізація SEO-оптимізації та маркетингових інструментів

Ефективна оптимізація для пошукових систем і застосування маркетингових ресурсів мають критичне значення для залучення природного трафіку та збільшення рівня конверсії. У процесі реалізації проєкту здійснюється ряд дій, щоб підвищити помітність прикладної реалізації веб-системи в пошуковиках.

Одним із ключових елементів SEO є створення SEO-дружніх URL (ЧПУ) за допомогою налаштувань модуля на базі OCFilter. Це підвищує CTR (click-through rate), адже користувачі легше розуміють, куди їх веде посилання. Модуль автоматично генерує URL на основі вибраних фільтрів, таких як категорії або виробники.

Для зручного управління тегами без необхідності змінювати код системи є інтеграція з Google Tag Manager (GTM). За допомогою GTM налаштовується відстеження подій, таких як кліки по кнопках «Додати до кошика», «Купити в 1 клік», перегляди сторінок товарів, оформлення замовлень, тощо.

Щоб покращити індексацію сайту пошуковими системами, реалізується автоматичне генерування та оновлення Sitemap. Оскільки в каталозі більше 10,000 пропозицій, Sitemap розбивається на кілька частин для більшої зручності для пошукових роботів. Генерація Sitemap здійснюється через кастомно написаний клас. Для важливих сторінок, таких як головна, категорії та популярні товари,

вказуються пріоритети, що прискорює індексацію нових сторінок та покращує охоплення системи у пошукових системах [28].

На рисунку 2.9 показано вигляд вхідної точки Sitemap системи.

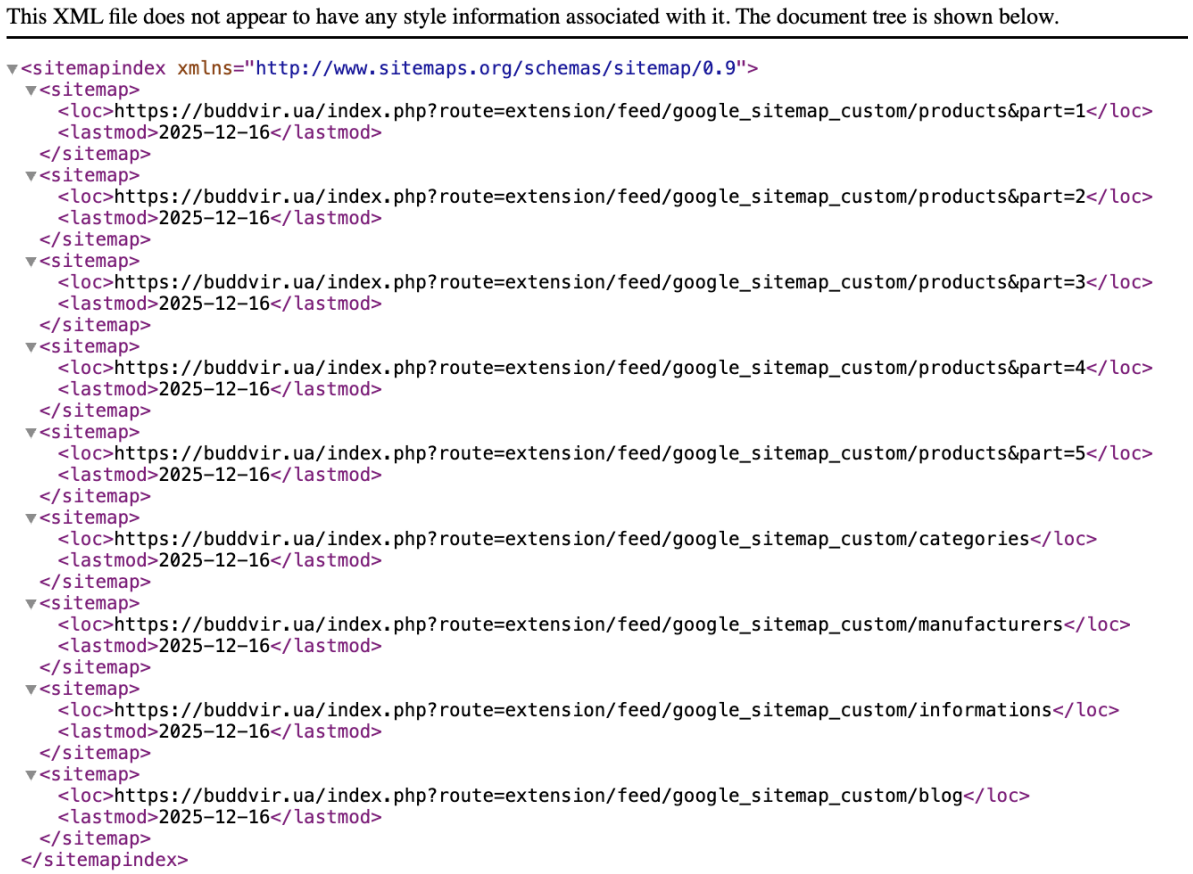

Рисунок 2.9 – Вхідна точка Sitemap системи

Для просування товарів через Google Shopping налаштовується експорт каталогу через XML формат для Google Merchant Feed. Цей інструмент дає можливість відображати товари в спеціальних блоках пошукової видачі. Особлива увага приділяється правильному заповненню обов'язкових атрибутів.

Впровадження SEO-заходів і сучасних маркетингових інструментів сприяє підвищенню видимості сайту в пошукових системах, збільшенню обсягу органічного трафіку та зростанню рівня конверсії. Застосування SEO-дружніх URL-адрес, GTM, Sitemap і Google Merchant Feed забезпечує більш ефективне просування ресурсу та покращує аналітику взаємодії користувачів із системою.

РОЗДІЛ 3. ТЕСТУВАННЯ, ВПРОВАДЖЕННЯ ТА ПІДТРИМКА

Тестування, впровадження та підтримка програмної системи є тими етапами життєвого циклу розробки, що забезпечують стабільну роботу. Даний розділ присвячено методологіям тестування, процесу впровадження та механізмам підтримки, які гарантують надійність, безпеку і безперебійну роботу веб-системи в умовах реального навантаження.

Прикладним проєктом для тестування та впровадження виступає інтернет-магазин «Buddvir», в якому реалізовані функції та модулі розроблюваної автоматизованої веб-системи, описані в попередніх розділах.

3.1 План тестування системи

План тестування системи враховує практичні вимоги та обмежені ресурси. Тому варто сконцентруватися на перевірці готових функцій і виявленні критичних помилок під час роботи. Основна мета тестування полягає в тому, щоб система функціонувала стабільно, відповідала дизайн-макетам і працювала належним чином під навантаженням. Стратегія тестування ґрунтується на покроковій перевірці функціоналу, тестах швидкодії на живій системі і виправленні виявлених помилок шляхом створення нових завдань [29].

У процесі розробки застосовується ітеративний підхід до тестування, за якого перевірка здійснюється поетапно у міру готовності окремих модулів і функціональних можливостей. Ключовим принципом цієї стратегії є тестування завершеного функціоналу: кожен модуль проходить перевірку одразу після реалізації, а не на окремому фінальному етапі. Усі виявлені помилки фіксуються у вигляді окремих завдань на доопрацювання та усунення недоліків. UX-тестування передбачає детальне візуальне порівняння інтерфейсу з дизайн-макетами з метою забезпечення повної відповідності запланованому вигляду та користувацькому досвіду. Усі виявлені розбіжності ретельно документуються та усуваються шляхом

створення відповідних завдань. Тестування швидкодії проводиться вже після запуску системи на продуктивному сервері, безпосередньо на живій системі. Виявлені проблеми з продуктивністю аналізуються та вирішуються шляхом оптимізації системи або тимчасового відключення проблемних компонентів.

У рамках проєкту реалізується кілька видів тестування, кожен з яких має своє призначення і специфічну методику виконання.

Функціональне тестування проводиться для перевірки роботи основних модулів системи, таких як каталог пропозицій з фільтрами, кошик, оформлення замовлення, особистий кабінет, а також інтеграція з CRM і Nova Poshta. Методика передбачає ручне тестування кожного модуля та фіксацію виявлених помилок у вигляді окремих завдань для подальшого виправлення.

Тести швидкодії також запускаються після введення системи в експлуатацію на продуктивному сервері, щоб виявити проблеми з продуктивністю під реальним навантаженням. Це включає моніторинг швидкості завантаження сторінок у пікові години, пошук «вузьких місць» і оптимізацію проблемних модулів.

UX-тестування спрямоване на перевірку того, наскільки інтерфейс відповідає дизайн-макетам і є зручним у використанні. Воно включає візуальне порівняння інтерфейсу, перевірку адаптивності на різних пристроях і виявлення проблем з навігацією.

Безпекове тестування проводиться для запобігання вразливостям та захисту даних користувачів. Воно включає санітайзинг вхідних даних, перевірку захисту особистих даних згідно з вимогами GDPR та тестування на вразливості через введення спеціальних запитів.

Для кожного виду тестування встановлені критерії приймання, які дозволяють оцінити готовність системи до впровадження.

Для функціонального тестування критеріями є відсутність критичних помилок у роботі основних модулів та правильна робота всіх функцій. Тести швидкодії вимагають, щоб швидкість завантаження сторінок не перевищувала 2-3 секунди і щоб не виникало критичних уповільнень чи збоїв у пікові моменти.

UX-тестування вимагає повної відповідності інтерфейсу дизайн-макетам без візуальних дефектів на різних пристроях. Безпекове тестування передбачає виявлення вразливостей до SQL-ін'єкцій та XSS, а також належний захист особистих даних користувачів відповідно до GDPR.

Для реалізації плану тестування використовуються різноманітні інструменти та методики. Ручне тестування передбачає перевірку функціональності та інтерфейсу вручну з використанням розробницьких інструментів браузера (Chrome DevTools) для аналізу продуктивності.

Моніторинг хостингу здійснюється за допомогою cPanel та інструментів хостингу (HostPro), які дозволяють відстежувати навантаження на сервер і аналізувати логи помилок. Санітайзинг даних виконується через вбудовані функції PHP для очищення вхідних даних. Зворотний зв'язок від користувачів збирається через модулі зворотного зв’язку, що дає можливість оперативно виявляти та виправляти помилки.

3.2 Види та методи тестування

Тестування включає різні види перевірок, кожна з яких має свої цілі та способи виконання. Вважаю, що головна мета цих перевірок впевнитися, що система працює правильно, відповідає вимогам та залишається стабільною під навантаженням. Тестування має здійснюватися систематично, охоплюючи усі важливі аспекти, такі як функціональність, продуктивність, зручність користування та безпека.

Функціональне тестування зосереджене на перевірці правильності роботи основних модулів системи. Його проводять після реалізації кожного функціонального блоку, і це включає ручну перевірку всіх ключових функцій. Серед об’єктів тестування – каталог пропозицій з фільтрами (з використанням OCFilter), де потрібно перевірити, чи правильно відображаються пропозиції за різними параметрами (категорії, ціна, виробник, розмір, колір), швидкість

завантаження сторінок з фільтрами, а також генерацію SEO-дружніх URL (ЧПУ). Перевіряються також кошик та оформлення замовлення, включаючи додавання товарів, правильний розрахунок суми, функція «Купити в 1 клік» та оформлення замовлення з різними способами доставки (Nova Poshta, самовивіз). У особистому кабінеті перевіряються можливості історії замовлень, програми лояльності та редагування особистих даних. Особливу увагу приділяють інтеграції з CRM та Nova Poshta, зокрема синхронізації даних про товари, замовлення та вибір відділень для доставки. Тестування проводиться вручну, фіксуються виявлені помилки у вигляді окремих завдань та проводиться повторне тестування після їх виправлення.

Тести швидкодії проводяться після запуску системи на продуктивному сервері, щоб виявити потенційні проблеми з продуктивністю під реальним навантаженням. Основна увага приділяється швидкості завантаження сторінок та стабільності роботи системи під час пікового трафіку [30].

UX-тестування проводять, щоб перевірити, наскільки інтерфейс відповідає дизайн-макетам та чи зручно ним користуватися. Головна мета забезпечити інтуїтивно зрозумілий інтерфейс, що відповідає очікуванням користувачів [31].

Тестування безпеки спрямоване на запобігання вразливостям і захист даних користувачів. Основна увага зосереджена на санітайзингу вхідних даних, захисті від SQL-ін єкцій та XSS, а також дотриманні вимог GDPR.

Висновки свідчать про те, що різні види тестування дозволяють забезпечити стабільну роботу системи, виявити й виправити помилки, поліпшити користувацький досвід. Функціональне тестування гарантує правильність роботи основних модулів, тести швидкодії – стабільність під реальним навантаженням, UX-тестування – відповідність дизайн-макетам, а тестування безпеки – захист даних користувачів.

3.3 Тестування функціональних модулів

Тестування функціональних модулів веб-системи проходить в міру реалізації кожного елемента системи. Важливо перевірити, як усе працює, чи відповідає це вимогам, та виявити помилки, які можуть заважати користувачам або бізнес-процесам. Тестування каталогу товарів з фільтрами зосереджене на перевірці правильного відображення товарів, швидкості завантаження сторінок і SEO-оптимізації.

В процесі тестування знайдено проблему з відображенням деяких товарів при використанні кількох фільтрів. Крім того, в категоріях з великою кількістю товарів спостерігається повільне завантаження сторінок, і ця проблема була усунена завдяки впровадженню пагінації та оптимізації запитів до бази даних.

Кошик та оформлення замовлення є особливо важливими, адже ці частини системи безпосередньо впливають на конверсію. Під час тестування перевіряється чи правильно товари додаються до кошика, чи оновлюється їх кількість, чи все добре з видаленням, а також правильність підрахунку суми замовлення. Також увага приділяється процесу оформлення: заповненню форми, вибору способів доставки та оплати, а також передачі даних у CRM та зворотньо. Перевіряється функція «Купити в 1 клік», яка дозволяє швидко оформити замовлення.

Тестування особистого кабінету спрямоване на перевірку історії замовлень, можливості редагування особистої інформації та роботи програми лояльності. Тут варто перевірити, чи всі замовлення правильно відображаються в кабінеті, як працює розрахунок знижок у програмі лояльності. Вигляд особистого кабінету, де також можна побачити власний відсоток знижки програми лояльності, подано на рисунку 3.1.

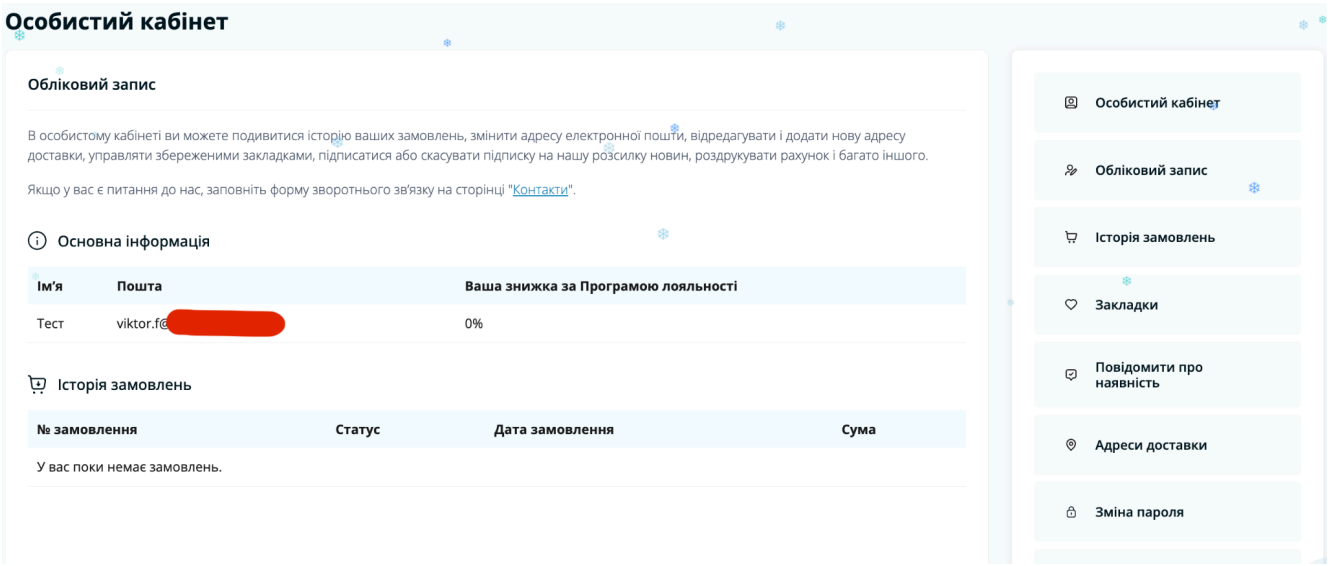

Рисунок 3.1 – Зовнішній вигляд особистого кабінету

Тестування інтеграції з CRM зосереджене на перевірці синхронізації пропозицій, замовлень та інших сутностей. В даному випадку перевіряється чи правильно оновлюються дані про товари в CRM, чи нові замовлення коректно передаються до CRM, чи приходять всі сутності з CRM в систему.

Отож, тестування функціональних модулів дозволяє вчасно виявляти й виправляти критичні помилки, забезпечуючи стабільну роботу автоматизованої веб-системи та покращуючи досвід користувачів. Кожен модуль ретельно перевіряється на коректність роботи, продуктивність і відповідність вимогам, і це в свою чергу сприяє ефективному функціонуванню системи.

3.4 Тестування інтерфейсу та адаптивності

Тестуючи інтерфейс та адаптивність немає чітко сформульованого плану, але є ясне уявлення про те, що сайт повинен коректно відображатися на різних пристроях і бути зручним для користувачів. Основна увага зосереджена на візуальній відповідності дизайн-макетам, адаптації для мобільних пристроїв та легкості навігації.

Після завершення роботи над кожним функціональним модулем системи обов'язково проводиться візуальна перевірка: реалізований інтерфейс порівнюється з дизайн-макетами в Figma, наданими замовником. Такий підхід дозволяє своєчасно виявляти та виправляти розбіжності у використанні кольорової гами, типографіки, розташуванні елементів керування, відступах та інтервалах між компонентами. Завдяки цьому досягається висока точність відповідності візуальної частини системи початковому задуму дизайнера, що покращує користувацький досвід і забезпечує цілісність стилю на всіх сторінках веб-системи.

Сучасні користувачі заходять на сайт з різних пристроїв десктопів, планшетів, смартфонів, тому обов’язково потрібно подивитися, як сайт виглядає на різних роздільних здатностях. Для цього використовуються інструменти розробника в браузері (Chrome DevTools) або реальні пристрої (iPhone, iPad, Android-смартфони). Також є ресурс під назвою Browserstack, який дозволяє переглядати роботу веб-сайту на телефонах та планшетах прямо з браузера ПК.

Тестуючи систему з допомогою вище перерахованих способів, критичних помилок в реалізованому дизайні не знайдено. Були знайдені невеликі проблеми: трохи інший відтінок певних кольорів, не ті заокруглення для кнопок, трохи більші або менші відступи, тощо. Ці проблеми було виправлено, щоб зовнішній вигляд повністю відповідав макету. Тестування інтерфейсу та адаптивності веб-системи дало можливість виявити та виправити основні проблеми, які можуть негативно вплинути на досвід користувачів. Візуальна частина системи є дуже важливою в умовах сучасного ринку, адже зараз багато приділяється тому, щоб продукт мав привабливий вигляд.

3.5 Процес впровадження

Оскільки важливо забезпечити стабільну роботу та зменшити ймовірність помилок під час запуску, впровадження проходить поетапно.

Спочатку реалізована система тестується на тестовому сервері, а потім відбувається перехід до проміжної версії (RC), де вже використовуються актуальні дані з бази. Тільки після успішних тестів на цих етапах система переноситься на продуктивне середовище. На початку розробка відбувається на закритому тестовому домені (Dev-версія), де реалізується та тестується функціонал. Ця версія містить копію бази даних, яка не відображає актуальні дані з продуктивного середовища. Такий підхід дозволяє швидко вносити зміни, тестувати нові функції та виправляти помилки. На цьому етапі тестується інтеграція з CRM та Nova Poshta, виправляються виявлені помилки та оптимізується код. Але оскільки база даних на Dev-версії не завжди актуальна, певні помилки з реальними даними можуть залишитися непоміченими. Отже, перед остаточним запуском потрібно перевірити реалізовану систему з актуальними даними.

Перед фінальним деплоєм код переноситься на проміжну версію (RC), яка максимально наближена до продуктового сервера. Тут код з Dev-версії копіюється на RC-сервер, база даних оновлюється до актуальної версії, щоб імітувати реальні умови, і перевіряється правильність роботи всіх модулів. Саме на цьому етапі можуть виявитися помилки, які не помічали на Dev-версії через відмінності в даних. Наприклад, можуть виникати проблеми з відображенням пропозицій, яких немає в тестовій базі, помилки синхронізації з CRM, пов'язані з реальними замовленнями та клієнтами, а також сповільнена робота деяких модулів, як-от «Нові надходження», які на Dev-версії працюють добре, але на RC уповільнюють сайт через велику кількість даних. Виявлені помилки фіксуються, і система знову тестується, поки не запрацює стабільно.

Після успішного тестування на RC-версії код переноситься на продуктивне середовище (Prod). Цей процес передбачає копіювання коду з RC-версії на Prod-сервер, оновлення бази даних до актуальної версії та перевірку роботи всіх модулів на бойовому сервері. Під час деплою використовуються інструменти хостингу (HostPro), які дозволяють автоматично створювати резервні копії сайту та бази даних перед оновленням, швидко відкотитися до попередньої версії у разі

критичних помилок, а також моніторити роботу сайту після деплою для виявлення можливих проблем. Резервні копії налаштовуються так, щоб автоматично створювати бекапи та бази даних щоночі за допомогою cron-задач. Також вони дають можливість зберігати копії на окремому сервері для швидкого відновлення при збоях і використовувати GitHub для контролю версій коду, що дозволяє відстежувати зміни та відновлювати попередні версії, якщо буде потрібно.

Після деплою на Prod-сервер відбувається моніторинг для виявлення можливих проблем. Зокрема, слідкування за продуктивністю під реальним навантаженням, аналіз відгуків користувачів. На цьому етапі виявилося, що модуль «Нові надходження» серйозно сповільнює роботу сайту через неоптимізований SQL-запит. Оскільки проблему не вдалося виправити, його тимчасово відключено з головної сторінки, щоб не впливати на продуктивність.

Для зручного деплою та відкату використовуються інструменти, як GitHub для контролю версій коду та можливості швидкого повернення до попередньої версії, cPanel та інструменти хостингу для автоматичного резервного копіювання та моніторингу, а також cron-задачі для регулярного створення бекапів бази даних та файлів. Ці інструменти мінімізують ризики під час деплою і допомагають швидко реагувати на помилки, забезпечуючи стабільну роботу системи та надійність даних.

3.6 Аналіз результатів впровадження

Оновлення інтернет-магазину «Buddvir» з старої версії на розроблену нову версію автоматизованої веб-системи з використанням Opencart 3 було спрямоване на покращення досвіду користувачів, автоматизацію бізнес-процесів і підвищення ефективності роботи в порівнянні з попередньою версією сайту. Хоча точні та повні дані поки ще формуються, вже можна зробити попередні висновки про те, як вдалося реалізувати ці цілі.

Одним із головних результатів впровадження стало покращення користувацького досвіду та конверсії. Новий дизайн і функціональність були створені для того, щоб спростити та пришвидшити процес оформлення замовлень, що позитивно вплинуло на задоволеність клієнтів. Завдяки зручному інтерфейсу, адаптивному дизайну, спрощеній формі замовлення (включаючи функцію «Купити в 1 клік») та вдосконаленій навігації, кількість відмов під час оформлення зменшилася. Клієнти легше знаходять потрібні пропозиції завдяки ефективним фільтрам і зручному каталогу. Крім того, програма лояльності з системою знижок для постійних клієнтів стимулює повторні покупки, що підвищує їхню залученість. Інтеграція з Nova Poshta дозволила вибирати зручне відділення під час оформлення замовлень. Також зазначено, що конверсії зросли, а клієнти частіше завершують покупки та менше покидають кошики під час оплати.

Автоматизація бізнес-процесів стала ще однією важливою перевагою впровадження. Інтеграція з CRM дозволила зменшити обсяг ручної праці. Ще це допомогло пришвидшити обробку замовлень і знизити ймовірність помилок, які виникають з ручного введення даних. Тепер менеджерам потрібно менше часу, щоб оновлювати інформацію про товари й статуси замовлень, оскільки ці процеси автоматизовані. Також зменшилась кількість запитів до служби підтримки щодо статусу замовлень, адже клієнти можуть відстежувати їх у своїх особистих кабінетах.

Покращення SEO та залучення трафіку теж стали важливими аспектами впровадження. Завдяки SEO-оптимізації вдалося поліпшити позиції сайту в пошукових системах. Валідні ЧПУ для фільтрів дозволили краще індексувати сторінки та залучати цільовий трафік. Google Merchant Feed забезпечив відображення товарів у Google Shopping, що збільшило видимість магазину для потенційних покупців. Налаштування GTM дало можливість відстежувати поведінку користувачів і оптимізувати маркетингові кампанії. За попередніми даними, органічний трафік зріс, але точні цифри ще вивчаються.

У порівнянні з попередньою версією, нова версія має кілька ключових переваг. По-перше, сучасний адаптивний дизайн, який краще сприймається на різних пристроях. По-друге, швидша робота завдяки оптимізації коду та бази даних. По-третє, більше функцій, таких як зручне оформлення замовлення. По-четверте, краща SEO-оптимізація, що призводить до зростання органічного трафіку. Стара версія мала застарілий інтерфейс, незручну навігацію та обмежений функціонал. Нова версія вирішила ці проблеми, зробивши систему більш конкурентоспроможною. Також тут варто зазначити, що тепер сайт працює на рушії нового покоління – Opencart 3, який розвивається та є актуальним на даний момент.

Незважаючи на позитивні результати, впровадження супроводжувалося деякими викликами. По-перше, повільна робота модуля «Нові надходження» гальмувала головну сторінку на сервері з високою продуктивністю.. По-друге, потрібно було додатково налаштувати маркетинг, адже, хоча технічна частина працює стабільно, для досягнення максимального ефекту варто продовжувати розвивати маркетингові стратегії (контекстна реклама, соціальні мережі).

Остаточні висновки підтверджують, що впровадження сайту на базі розробленої автоматизованої веб-системи покращило користувацький досвід та автоматизувало ключові бізнес-процеси. Система забезпечує зручніше оформлення замовлень, зменшує навантаження на менеджерів і створює передумови для зростання трафіку та ефективності SEO. Подальший розвиток передбачає оптимізацію продуктивності окремих модулів, удосконалення маркетингових інструментів і аналіз показників конверсії для оцінки результативності впроваджених рішень.

Отже, впровадження розробленої веб-системи повністю виправдало очікування: воно забезпечило стабільну та швидку роботу, значно підвищило зручність і комфорт користувачів та спростило внутрішні бізнес-процеси для подальшого розвитку і масштабування діяльності підприємства, а також інтеграції додаткових сервісів і функціональних модулів у майбутньому.

РОЗДІЛ 4. ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

Даний розділ присвячений аналізу умов безпечної роботи під час розробки та експлуатації автоматизованої веб-системи, а також заходам щодо зниження ризиків для персоналу в разі виникнення надзвичайних ситуацій. У межах розділу розглядаються основні вимоги охорони праці, інформаційної та цивільної безпеки, дотримання яких забезпечує безпечні та комфортні умови роботи в мирний і воєнний час.

4.1 Охорона праці

Під час розробки та практичного використання автоматизованої веб-системи, створеної з використанням CMS OpenCart, мови програмування PHP, клієнтської бібліотеки JQuery та препроцесора SCSS, особлива увага приділялася питанням безпечних умов роботи та збереження працездатності. Основна взаємодія з розробленою системою здійснюється за допомогою персональних комп'ютерів, що зумовлює необхідність дотримання встановлених норм безпеки під час тривалої роботи з електронними засобами оброблення інформації.

У процесі проєктування та впровадження веб-системи враховувалися чинні нормативно-правові акти України у сфері охорони праці, зокрема Закон України «Про охорону праці», НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями» [32-33]. Дотримання цих вимог є важливим чинником зниження ризику професійних захворювань, перевтоми та погіршення стану здоров'я осіб, які працюють із системою на постійній основі.

Інтерфейс користувача автоматизованої веб-системи було розроблено з урахуванням принципів і вимог до зручності сприйняття інформації згідно стандарту ДСТУ ISO 9241 [34]. Для цього застосовано стриману кольорову

палітру, читабельні шрифти оптимального розміру, достатні міжрядкові інтервали та логічну структуру елементів керування. Це спрямовано на зменшення зорового навантаження, підвищення зручності навігації та зниження напруги під час роботи з веб-інтерфейсом.

Згідно з санітарними нормами, під час експлуатації програмного забезпечення передбачається дотримання регламентованої тривалості безперервної роботи за комп'ютером. Рекомендується періодично робити перерви для відпочинку зорового апарату та зменшення статичного навантаження. Робочі місця користувачів повинні бути обладнані ергономічними меблями, а монітори - розташовані на оптимальній відстані та висоті відносно рівня очей при нормативному рівні освітлення.

Важливим аспектом безпечної експлуатації веб-системи є дотримання вимог електробезпеки та пожежної безпеки. Комп'ютерне обладнання має використовуватися лише за умови справності електричних мереж, наявності заземлення та відсутності пошкоджених кабелів або з'єднань, так як це описано в НПАОП 0.00-7.15-18. Не допускається перевантаження електромереж, використання несправних розеток чи подовжувачів, а також розміщення техніки поблизу джерел тепла або легкозаймистих матеріалів.

Додатково слід враховувати вимоги пожежної безпеки під час організації робочих місць та експлуатації комп'ютерної техніки. Приміщення, у яких здійснюється робота з веб-системою, повинні бути обладнані первинними засобами пожежогасіння, а персонал - проінструктований щодо дій у разі виникнення пожежі або аварійної ситуації. Забезпечення вільного доступу до евакуаційних виходів, дотримання правил зберігання електрообладнання та регулярний контроль технічного стану пристроїв сприяють зниженню ризику виникнення пожежонебезпечних ситуацій і підвищують загальний рівень безпеки праці.

У приміщеннях, у яких здійснюється розробка веб-системи, мають забезпечуватися належні та комфортні параметри мікроклімату, що відповідають

чинним санітарно-гігієнічним вимогам. Ці вимоги описані в ДСН 3.3.6.042-99 [35]. Повинен підтримуватися оптимальний температурний режим у робочих зонах, що створює сприятливі умови для тривалої інтелектуальної діяльності та знижує ризик перевтоми персоналу. Відносна вологість повітря має знаходитись у допустимих межах, оскільки надмірно сухе або вологе повітря негативно впливає на самопочуття користувачів, стан зорового апарату та загальну працездатність.

Важливу роль відіграє наявність ефективної системи вентиляції або регулярне провітрювання приміщень, що забезпечує належний повітрообмін і запобігає накопиченню вуглекислого газу. Недостатня вентиляція може призводити до зниження концентрації уваги, підвищеної втомлюваності та зниження продуктивності праці. Рівень шуму в робочих приміщеннях також повинен відповідати нормативним значенням і не перевищувати допустимих показників, оскільки постійний фоновий шум негативно впливає на нервову систему та ускладнює виконання завдань, пов'язаних із розробкою програмного забезпечення [35].

Таким чином, у процесі розробки та впровадження автоматизованої веб-системи було враховано комплекс вимог охорони праці та безпеки, що забезпечує безпечні й комфортні умови роботи користувачів. Реалізація зазначених заходів відповідає чинному законодавству України та сприяє підвищенню ефективності використання програмного продукту в практичній діяльності.

4.2 Забезпечення безпеки функціонування веб-системи під час НС мирного та воєнного часу

Сучасні автоматизовані веб-системи відіграють важливу роль у функціонуванні бізнесу, зокрема у сфері електронної комерції, де стабільність роботи інформаційних ресурсів безпосередньо впливає на економічну діяльність підприємств. В умовах надзвичайних ситуацій мирного часу, а також під час воєнного стану, питання безпеки та стійкості функціонування веб-систем набуває

особливої актуальності. Це обумовлено зростанням ризиків порушення інфраструктури, кіберзагроз, перебоїв електропостачання та обмеження доступу до інформаційних ресурсів. Відповідно до основ цивільної безпеки, забезпечення безперервної роботи інформаційних систем є складовою загальної системи захисту об'єктів господарської діяльності [36].

Розроблена в межах роботи автоматизована веб-система з використанням CMS OpenCart, мови програмування PHP, клієнтської бібліотеки JQuery та препроцесора SCSS розглядається як інформаційна система підтримки бізнес-процесів для інтернет-магазину «Buddvir». З огляду на це, веб-система може бути віднесена до об'єктів, стабільність функціонування яких є важливою для збереження економічної активності підприємства в умовах надзвичайних ситуацій. Хоча подібні системи не належать до об'єктів підвищеної небезпеки, їх відмова або тривала недоступність може призвести до фінансових втрат, втрати клієнтів та порушення ланцюгів постачання.

Одним із ключових факторів впливу надзвичайних ситуацій на роботу веб-системи є порушення фізичної інфраструктури, зокрема серверного обладнання та мережевих каналів. У мирний час це можуть бути аварії на електромережах, пожежі, техногенні інциденти або стихійні лиха. У воєнний час до цього додаються ризики цілеспрямованого ураження об'єктів критичної інфраструктури, кібератак, обмеження доступу до центрів обробки даних та загроза втрати інформації. Відповідно до положень цивільної безпеки, викладених у навчальний посібнику «ТЕХНОЕКОЛОГІЯ ТА ЦИВІЛЬНА БЕЗПЕКА. ЧАСТИНА «ЦИВІЛЬНА БЕЗПЕКА»», важливим завданням є завчасне планування заходів, спрямованих на підвищення стійкості функціонування інформаційних систем у таких умовах.

Для забезпечення безпеки функціонування розробленої веб-системи доцільним є застосування комплексу організаційних та технічних заходів. Насамперед, до організаційних заходів належить використання хмарної або віддаленої серверної інфраструктури, розміщеної поза межами зон потенційного

ураження. Такий підхід дозволяє знизити залежність системи від локальних факторів ризику та забезпечити доступ до веб-ресурсу навіть у разі порушення роботи окремих регіональних мереж.

Важливим аспектом стійкості веб-системи є регулярне резервне копіювання даних. Бази даних MySQL, повинні зберігатися у вигляді резервних копій на фізично відокремлених носіях або у хмарних сховищах. У контексті цивільної безпеки резервне копіювання розглядається як один із базових заходів зменшення наслідків надзвичайних подій.

Окрему увагу слід приділити питанням кібербезпеки, оскільки в умовах воєнного часу суттєво зростає ймовірність кібератак на інформаційні ресурси. Захист веб-системи реалізується шляхом використання актуальної версії CMS OpenCart, своєчасного оновлення програмного забезпечення, обмеження доступу до адміністративної частини системи, а також застосування засобів фільтрації трафіку. Такі заходи зменшують ризик несанкціонованого доступу та порушення цілісності даних, що відповідає загальним принципам захисту інформації в надзвичайних ситуаціях [37].

Забезпечення безпеки функціонування веб-системи нерозривно пов'язане із захистом персоналу, який здійснює її підтримку та адміністрування. У разі виникнення надзвичайних ситуацій пріоритетом є збереження життя і здоров'я працівників, тому організація віддаленого доступу до системи є важливим елементом безпеки. Можливість адміністрування веб-системи з використанням захищених каналів зв'язку дозволяє зменшити необхідність фізичної присутності персоналу на робочих місцях, що особливо актуально під час воєнного стану або техногенних загроз.

У межах цивільного захисту також важливим є інформування персоналу про порядок дій у разі виникнення надзвичайних ситуацій. Хоча розроблена веб-система не передбачає безпосередньої роботи у зонах ураження, працівники, відповідальні за її експлуатацію, повинні бути обізнані з базовими правилами поведінки, евакуації та реагування на сигнали тривоги. Це відповідає загальним

вимогам організації цивільного захисту населення та персоналу об'єктів господарської діяльності.

Окрім цього, доцільним є періодичне проведення інструктажів та навчальних заходів з питань цивільного захисту, що дозволяє підтримувати належний рівень готовності персоналу до дій у разі виникнення надзвичайних ситуацій. Наявність чітко визначених маршрутів евакуації, доступ до засобів оповіщення та розуміння алгоритмів взаємодії з відповідними службами сприяють зниженню ризиків для життя і здоров'я працівників [36].

Таким чином, забезпечення безпеки функціонування автоматизованої веб-системи у надзвичайних ситуаціях мирного та воєнного часу досягається шляхом поєднання організаційних, технічних і інформаційних заходів. Реалізація резервного копіювання, використання віддаленої серверної інфраструктури, підвищення рівня кіберзахисту та організація безпечних умов роботи персоналу дозволяють підвищити стійкість системи та зменшити негативні наслідки можливих надзвичайних ситуацій. Запропоновані підходи узгоджуються з основними положеннями цивільної безпеки та відповідають сучасним вимогам до захисту інформаційних систем в умовах підвищених ризиків.

ВИСНОВКИ

У процесі виконання даної кваліфікаційної роботи було успішно вирішено науково-технічне завдання по розробці та впровадженню автоматизованої веб-системи із використанням платформи OpenCart 3, мови програмування PHP, бібліотеки JQuery, препроцесора SCSS, та інших IT-технологій. Розроблена система об'єднує сучасне архітектурне рішення, інтеграцію зовнішнього сервісу і оптимізацію бізнес-процесів. Основна мета дослідження полягала в тому, щоб створити систему, яка забезпечує високий рівень автоматизації, зручність для користувачів та стабільну роботу навіть під навантаженням.

Теоретичне значення роботи виявляється у систематизації підходів до проєктування та реалізації автоматизованої веб-системи для e-commerce сфери з використанням OpenCart 3. Адаптуючи архітектуру MVC, було розділено логіку, представлення та управління даними. Також увагу зосереджено на оптимізації бази даних MySQL, зокрема на індексуванні таблиць. Було проведено аналіз і реалізацію механізмів інтеграції з зовнішніми системами, такими як CRM та API Nova Poshta.

Практичне значення цієї роботи полягає в успішному впровадженні розробленої прикладної веб-системи для інтернет-магазину «Buddvir», який пропонує покращений користувацький досвід завдяки адаптивному дизайну і зручній навігації, автоматизацію бізнес-процесів через інтеграцію з CRM-системою, високу продуктивність, покращення SEO-оптимізації.

Показники отриманих результатів підтверджують ефективність реалізованих рішень. Серед них:

- Зменшення часу обробки замовлень завдяки інтеграції з CRM-системою.
- Швидкість завантаження сторінок до 2-3 секунд.
- Збільшення конверсії під час оформлення замовлення.
- Покращення індексації сайту пошуковими системами через SEO-оптимізацію.

- Покращена надійність та безпека системи через оновлення до актуальної версії Opencart 3.
- Збільшена задоволеність користувачів в користуванні системою через покращений зовнішній інтерфейс.

Достовірність результатів підтверджується успішним впровадженням системи в реальних умовах експлуатації, а також позитивним зворотнім зв'язком від користувачів і менеджерів. Проведені тести: UX-тестування, функціональне тестування, тестування безпеки свідчать про стабільність, надійність та захищеність системи.

Результати, отримані в ході розробки автоматизованої веб-системи, можуть бути рекомендовані до практичного використання під час створення та модернізації веб-платформ електронної комерції різного профілю. Запропоноване архітектурне рішення, модульна структура та механізми автоматизації можуть бути використані для впровадження аналогічних систем з метою оптимізації бізнес-процесів, покращення користувацького досвіду, підвищення продуктивності. Також, результати роботи можуть бути використані в навчальному процесі як приклад практичної реалізації веб-системи на базі CMS OpenCart із застосуванням сучасних веб-технологій.

Отже, результати цієї кваліфікаційної роботи мають теоретичне та практичне значення, бо не лише вирішують актуальне науково-технічне завдання по розробці автоматизованої веб-системи, але й слугують основою для подальшого розвитку та впровадження подібних рішень для інших платформ електронної комерції.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Методичні вказівки до виконання кваліфікаційної роботи магістра для здобувачів спеціальності 121 – Інженерія програмного забезпечення, всіх форм навчання / укладачі: Михалик Д.М., Цуприк Г.Б., Бревус В.М., Мудрик І.Я. – Тернопіль: Тернопільський національний технічний університет імені Івана Пулюя, 2024. – 44 с. URL: http://elartu.tntu.edu.ua/handle/lib/50316

2. Олянін, Д., Цуприк, Г. (2025) Transformer Neural Networks in Industry 4.0 / Д. Олянін, Г. Цуприк, Т. Говорущенко, О. Багрій-Заяць, І. Андрущак // Computer Information Technologies in Industry 4.0: proceedings of the 3rd International Workshop (CITI-2025), Ternopil, Ukraine, 11–12 June 2025. – Ternopil: Ternopil Ivan Puluj National Technical University, 2025 (Scopus)

3. Tsupryk, H., Olianin, D. (2025). Vydobuvannia danyh z tekstu vykorystovuiuchy transformerni neironni merezhi [Data extraction from text using Transformer Neural Networks]. Information Technology: Computer Science, Software Engineering and Cyber Security, 125–130, DOI: https://doi.org/10.32782/IT/2025-2-13

4. Tsupryk, H., Olianin, D. (2025). Overview of transformers role indata mining from unstructured data. International Scientific-technical journal «Measuring and computing devices in technological processes» 2025, Issue 2, 125–130, DOI: 10.31891/2219-9365-2025-82-52

5. Фільтр для Opencart 3 – OCFilter. Документація, опис модулю, особливості роботи. URL: https://ocfilter.com/documentation/4.8/

6. Smart Checkout – модуль для оформлення замовлення для Opencart 3. URL: https://dsdocs.octemplates.net/oformlennya-zamovlennya

7. Андрющенко Т.Ю., Скрипань Р.О. Аналіз особливостей розробки веб-системи для одягу з адаптивним дизайном, 2022. URL: https://openarchive.nure.ua/entities/publication/0a22e487-5b14-4bf7-8111-730fb83e51c3

8. Nova Poshta API – модуль доставки для OpenCart. OcMax. URL: https://oc-max.com/docs/novaposhta/instruction.html

9. Elegant Themes – Technical SEO Strategies for Web Developers (2025 Guide). URL: https://www.elegantthemes.com/blog/marketing/technical-seo/

10. Bootstrap 4 – Верстка та компоненти. Офіційна документація. URL: https://getbootstrap.com/docs/4.0/

11. Сазерленд Д. Scrum: Навчись робити вдвічі більше за менший час. Київ, 2018. 280 с.

12. MVC Architecture Explained: Model, View, Controller. Codeacademy. URL: https://www.codecademy.com/article/mvc-architecture-model-view-controller/

13. Opencart 3 – Developer Guide. Opencart, 2016. URL: https://docs.opencart.com/en-gb/developer/module/

14. OpenCart документація – Гайд для розробника. URL: https://docs.opencart.com/

15. OCStore рішення для Opencart 3. URL: https://ocstore.com/

16. Шаблон OCDeals для Opencart 3. URL: https://dsdocs.octemplates.net/

17. PHP Tutorial: PHP The Right Way. URL: https://phptherightway.com/

18. PHP Manual – Офіційна документація мови PHP. URL: https://www.php.net/manual/en/

19. Бер Б., Єгуда К. JQuery. Детальний посібник з просунутого JavaScript. Київ, 2018. 384 с.

20. JQuery Learning Center. URL: https://learn.jquery.com/

21. SASS – Офіційна документація препроцесора. URL: https://sass-lang.com/documentation/

22. MySQL 8.0 Manual. URL: https://dev.mysql.com/doc/refman/8.0/en/

23. Mr Web – How to Make an Ecommerce Website with OpenCart | Opencart Tutorial for Beginners 2025. Youtube. URL: https://www.youtube.com/watch?v=Vqac4gfciYs

24.Силка М.В., Цуприк Г.Б. Сучасні технології при розробці програмного забезпечення для автоматизованої інформаційної системи обробки замовлень та товарообігу, 2018. URL: https://elartu.tntu.edu.ua/handle/lib/27022

25.Гринчак О.В., Моцик Р.В. Веб-технології та дизайн. URL: https://journals.khnu.km.ua/vestnik/wp-content/uploads/2021/08/6-1.pdf

26.Кобзев І.В., Колоусова А.К. Розробка корпоративного веб-сайту для IT-компанії. URL: https://openarchive.nure.ua/entities/publication/cc218e3d-0cc1-49ac-9da5-244aec491b21

27.Костів С.В., Цуприк Г.Б. Розробка системи автоматизації бізнес-процесів через централізацію інформаційних ресурсів, 2018. URL: https://elartu.tntu.edu.ua/handle/lib/27110

28.Wedex: Що таке карта сайту та три способи, як створити файл sitemap.xml. URL: https://wedex.com.ua/blog/sposobi-stvoriti-kartu/

29.Gayathri M. Full Stack Testing: A Practical Guide for Delivering High Quality Software. O'Reilly Media, Inc., 2022. 405 с.

30.LuxeQuality – How to do Load Testing? [A FULL GUIDE]. URL: https://luxequality.com/blog/how-to-do-load-testing/

31.Evergreen – Все що ви хотіли дізнатись про UX тестування: етапи, види, процес. URL: https://evergreens.com.ua/ua/articles/ux-testing.html

32.Закон України «Про охорону праці» від 14.10.1992 № 2694-XII (зі змінами). URL: https://zakon.rada.gov.ua/laws/show/2694-12 (дата звернення: 16.12.2025).

33.НПАОП 0.00-7.15-18. Вимоги щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями: затв. Наказом Мінсоцполітики України від 14.02.2018 № 207. URL: https://zakon.rada.gov.ua/laws/show/z0508-18 (дата звернення: 16.12.2025).

34.ДСТУ ISO 9241. Ергономіка взаємодії людина-система (різні частини). URL: https://online.budstandart.com/ua (пошук за серією стандартів) (дата звернення: 16.12.2025)

35.Санітарні норми мікроклімату виробничих приміщень ДСН 3.3.6.042-99. URL: https://zakon.rada.gov.ua/rada/show/va042282-99#Text (дата звернення: 16.12.2025)

36.Стручок В.С. Навчальний посібник «ТЕХНОЕКОЛОГІЯ ТА ЦИВІЛЬНА БЕЗПЕКА. ЧАСТИНА «ЦИВІЛЬНА БЕЗПЕКА»» / Тернопіль: ФОП Паляниця В. А., – 156 с. Отримано з http://elartu.tntu.edu.ua/handle/lib/39424

37.Стручок В.С. Методичний посібник для здобувачів освітнього ступеня «магістр» всіх спеціальностей денної та заочної (дистанційної) форм навчання «БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ» / Тернопіль: ФОП Паляниця В. А., –156 с. Отримано з https://elartu.tntu.edu.ua/handle/lib/39196.

ДОДАТКИ

Додаток А. Тези конференції

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ІВАНА ПУЛЮЯ

М А Т Е Р І А Л И

XIII НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

17-18 грудня 2025 року

ТЕРНОПІЛЬ
2025

004.41
В. Філик; Г. Цуприк, к. т. н., доц.
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

АВТОМАТИЗОВАНА ВЕБ-СИСТЕМА НА ОСНОВІ OPENCART 3: ОСОБЛИВОСТІ РЕАЛІЗАЦІЇ ТА ІНТЕГРАЦІЇ

UDC 004.41
V. Filyk; H. Tsupryk, PhD., Assoc. Prof.

AUTOMATED WEB SYSTEM BASED ON OPENCART 3: IMPLEMENTATION AND INTEGRATION FEATURES

Сучасні тенденції розвитку сприяють створенню таких систем, що здатні забезпечити високий рівень автоматизації бізнес-процесів та інтеграцію з зовнішніми сервісами. Ефективність e-commerce систем безпосередньо залежить від архітектурних рішень, оптимізації бази даних та зручності користувацького інтерфейсу. Актуальність теми полягає в необхідності розробки автоматизованої веб-системи, яка б об'єднувала функціональність, масштабованість та зручність для кінцевого користувача.

Об’єктом дослідження є процес електронної комерції, що забезпечує функціонування онлайн-торгівлі. Предметом дослідження є інтернет-магазин «Buddvir», створений на базі платформи OpenCart 3 із використанням мови програмування PHP. Для його аналізу застосовуються методи структурного проєктування, тестування продуктивності, оптимізації та дослідження користувацького досвіду. Інтеграція зовнішніх систем здійснюється через REST API.

Розроблена багатошарова архітектура на основі MVC-патерну забезпечує розділення логіки, представлення та управління даними. Використання Bootstrap 4 та SCSS дозволило створити адаптивний інтерфейс. Оптимізація MySQL бази даних підвищила швидкість обробки запитів для великого каталогу.

Інтеграція з CRM-системою автоматизувала синхронізацію даних, що зменшило кількість ручної роботи та помилок. Використання Nova Poshta API спростило процес вибору відділень доставки. Тестування системи показало стабільну роботу під навантаженням та зручний інтерфейс.

Розроблена автоматизована веб-система демонструє високий рівень продуктивності, зручність використання, інтеграцію з зовнішніми сервісами. Отримані результати можуть бути використані для створення подібних e-commerce проєктів, що вимагають масштабованості, надійності та автоматизації.

Література

1. Opencart 3 – Developer Guide. Opencart, 2016. URL: https://docs.opencart.com/en-gb/developer/module/
2. MVC Architecture Explained: Model, View, Controller. Codeacademy. URL: https://www.codecademy.com/article/mvc-architecture-model-view-controller/
3. Олянін, Д., Цуприк, Г. (2025) Transformer Neural Networks in Industry 4.0 / Д. Олянін, Г. Цуприк, Т. Говорущенко, О. Багрій-Заяць, І. Андрущак // Computer Information Technologies in Industry 4.0: proceedings of the 3rd International Workshop (CITI-2025), Ternopil, Ukraine, 11–12 June 2025. – Ternopil : Ternopil Ivan Puluj National Technical University, 2025 (Scopus)
4. Tsupryk, H., Olianin, D. (2025). Vydobuvannia danyh z tekstu vykorystovuiuchy transformerni neironni merezhi [Data extraction from text using Transformer Neural Networks]. Information Technology: Computer Science, Software Engineering and Cyber Security, 125–130, DOI: https://doi.org/10.32782/IT/2025-2-13
5. Tsupryk, H., Olianin, D. (2025). Overview of transformers role indata mining from unstructured data. International Scientific-technical journal «Measuring and computing devices in technological processes» 2025, Issue 2, 125–130, DOI: 10.31891/2219-9365-2025-82-52
6. Tsupryk H. LLM-based Extraction from Resumes / D. Olianin, H. Tsupryk // Advanced Technologies in Scientific Research: collection of scientific papers with proceedings of the 1st International Scientific and Practical Conference, Rotterdam, Netherlands, 20–22 August 2025. – International Scientific Unity, 2025. – 72–76

Додаток Б. Логічна та фізична модель бази даних

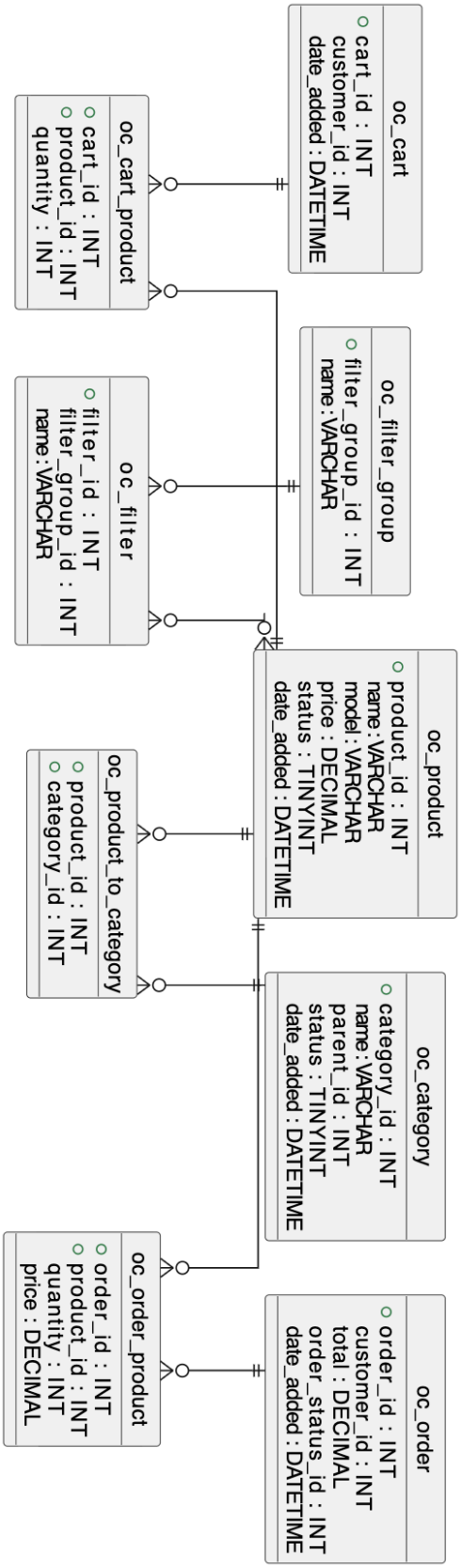

Рисунок Б.1 – Діаграма основних сутностей бази даних веб-системи