Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інфокомунікаціїі
(повна назва)

Кафедра Інформаційно-мережної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Створення вебсайту за допомогою Wordpress

(тема)

Виконав:
студент 2 курсу, групи *ІМІм-22-3*
Кутя Б.С.

Спеціальності 172 Телекомунікації та радіотехніка
(код і повна назва спеціальності)
Тип програми Освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма Інформаційно-мережна інженерія
(повна назва освітньої програми)
Керівник доц. Скорик Ю.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Безрук В.М.
(підпис) (прізвище, ініціали)

2024 р.

Не містить відомостей, заборонених до відкритого публікування

Студент _____ Кутя Б.С.
(підпис) (прізвище та ініціали)

Керівник _____ Скорик Ю.В.
(підпис) (прізвище та ініціали)

Харківський національний університет радіоелектроніки

Факультет Інфокомунікацій
(повна назва)

Кафедра Інформаційно-мережної інженерії
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 172 Телекомунікації та радіотехніка
(код і повна назва)

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

Освітня програма Інформаційно-мережна інженерія
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри ІМІ_____
(підпис)
“____” _____2024 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Студентові *Куті Богдану Сергійовичу*
(прізвище, ім’я, по батькові)

1. Тема роботи Створення вебсайту за допомогою Wordpress

затверджені наказом університету від 18.03.2024 року № 232 *Ст*

2. Термін подання студентом роботи до екзаменаційної комісії *20 червня 2024 р.*

3. Вихідні дані до роботи

Розробити функціональний інтернет магазин на базі WordPress. Описати процес створення та налаштування сайту.

4. Перелік питань, що потрібно опрацювати в роботі

Вступ

1.Опис платформи Wordpress

2.Огляд можливостей створення вебсайту на Wordpress

3.Створення вебсайту

Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) *Слайди у форматі Power Point (назва, мета роботи, вступ, план поверху будівлі офісу, вибір мережного обладнання, загальна схема підприємства, розподіл мережі на VLAN у головному офісі, висновки)*

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	*Ознайомлення із завданням. Уточнення ТЗ*	*30.05.24*	*виконано*
2	*Підбір літератури за темою роботи*	*31.05-01.06.24*	*виконано*
3	*Виконання розділу 1*	*02.06-06.06.24*	*виконано*
4	*Виконання розділу 2*	*07.06-09.06.24*	*виконано*
5	*Виконання розділу 3*	*10.06-11.06.24*	*виконано*
6	*Оформлення пояснювальної записки*	*12.06-13.06.24*	*виконано*
7	*Оформлення презентаційного матеріалу,*	*14.05-20.06.24*	*виконано*
	підготовка до захисту у ЕК		

Дата видачі завдання *18.03.2024 р.*

Студент _____ Кутя Б.С.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Скорик Ю.В.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка: 81 с., 46 рис., 15 джерел.

Об'єкт дослідження – створення вебсайту на базі WordPress.

Мета роботи – створення інтернет магазину на базі WordPress та плагіну WooCommerce для замовника “ShinaHub”.

Розроблено вебсайт на основі технологій HTML5, CSS3, JavaScript, PHP.

Створено всі шаблони сторінок, наповнили вебсайт контентом. Розроблено фільтр на головній сторінки, для пошуку товарів по заданим параметрами, а також кастомний кошик, який відображає товари, які додані до нього, кількість їх, та загальну суму товарів. Розроблено плагін, який додає та оновлює вже існуючі товари в базі даних. Отримано повністю функціонуючий вебсайт, який готовий до переїзду до хостингу.

ПРОГРАМУВАННЯ, WORDPRESS, ВЕБСАЙТ, СЕРВЕР, АДМІНКА, ПЛАГІН, КОД.

THE ABSTRACT

Explanatory note: 81 pp., 46 pictures, 15 sources.

The object of the research is the creation of a website based on WordPress.

The purpose of the work is to create an online store based on WordPress and the WooCommerce plugin for the client "ShinaHub".

Developed a website based on HTML5, CSS3, JavaScript, PHP.
Created all page templates, filled the website with content. Developed a filter on the main page to search for products by specified parameters, as well as a custom shopping cart, which displays the goods that are added to it, the number of them, and the total amount of goods. Developed a plugin that adds and updates existing products in the database. Got a fully functioning website that is ready to be moved to hosting.

PROGRAMMING, WORDPRESS, WEBSITE, SERVER, ADMINISTRATION, PLUG-IN, CODE.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

PHP - Hypertext Preprocessor

HTML - HyperText Markup Language

CSS - Cascading Style Sheets

JS - JavaScript

AJAX - Asynchronous JavaScript And XML

ВСТУП

В нашому сучасному світі є Інтернет, який став не лише інструментом для отримання нових знань, але й важливим інструментом для розвитку у різних сферах діяльності. Створення вебсайту стало однією з найефективніших стратегій для залучення аудиторії, просування бренду, спілкування з клієнтами та подальшого розвитку, продаж своїх товарів, створення корпоративних мереж, та інше. У цьому контексті WordPress, як найпопулярніша платформа для створення вебсайтів, виявляється надзвичайно корисним інструментом для будь-якого, хто має бажання мати власний онлайн-присутність.

Спрощений інтерфейс, велика кількість готових шаблонів, безліч додаткових плагінів та розширень роблять WordPress доступним для широкого кола користувачів, незалежно від їхнього рівня технічних знань. Від невеликих особистих портфоліо до великих корпоративних вебсайтів, онлайн магазинів, площадок для навчання. WordPress може задовольнити різноманітні потреби та вимоги. Але для розробки більш складних рішень необхідні навички в програмуванні, та розуміння ядра WP.

Створення вебсайту на базі WordPress у створенні вебсайтів може мати безліч переваг:

- WordPress надає інтуїтивно зрозумілий інтерфейс, що дозволяє швидко оволодіти базовими функціями та почати створення вебсайту без особливих навичок програмування чи дизайну, легкість наповнення контентом сайт. Легко масштабувати існуючий сайт.

- WordPress може бути використаний для створення вебсайтів будь-якої складності – від простого блогу до складних корпоративних порталів чи електронних магазинів, також є можливість зв’язати сайт на Wordpress зі своєю CRM системою, щоб отримати повноцінну інтегровану систему, яку можна використовувати для магазинів, корпоративної сітки, інтеграція WordPress сайту з CRM системою може дозволити ефективно управляти складом та інше.

- Велика кількість безкоштовних та платних готових тем відображення візуальної складовою сайту. Велика кількість безкоштовних та платних плагінів дозволяє налаштувати вебсайт з індивідуальних потреб, вимог, та функціональної необхідності.

- WordPress має велику, активну та дружню спільноту користувачів і розробників, в яка завжди може відповісти на ваші запитання та допомогти з будь-якими питаннями чи проблемами.

- WordPress має вбудовані інструменти для оптимізації вебсайту для пошукових систем, що допомагає підвищити його видимість в пошукових системах, але також її можна покращити за допомогою різних способів.

Отже, створення вебсайту за допомогою WordPress є важливим кроком для будь-якої організації, бізнесу, творчого проекту чи приватного користувача, який прагне мати власну онлайн-присутність. Його використання відкриває безмежні можливості для комунікації, бізнесу, та розвитку та досягнення успіху в інтернет-середовищі.

1 ОГЛЯД WORDPRESS

1.1 Основні дані та принципи Wordpress

WordPress (також відомий як WP або WordPress.org) – це система керування веб-контентом з відкритим кодом. Спочатку він був створений як інструмент для публікації блогів, але розвинувся для підтримки публікації іншого веб-вмісту, включаючи більш традиційні веб-сайти, списки розсилки та Інтернет-форуми, медіа-галереї, сайти членства, системи керування навчанням та онлайн-магазини. WordPress доступний як безкоштовне програмне забезпечення з відкритим вихідним кодом, і є однією з найпопулярніших систем керування вмістом – його використовували 59.9% із 10 мільйонів найкращих веб-сайтів станом на грудень 2023 року [1, 2].

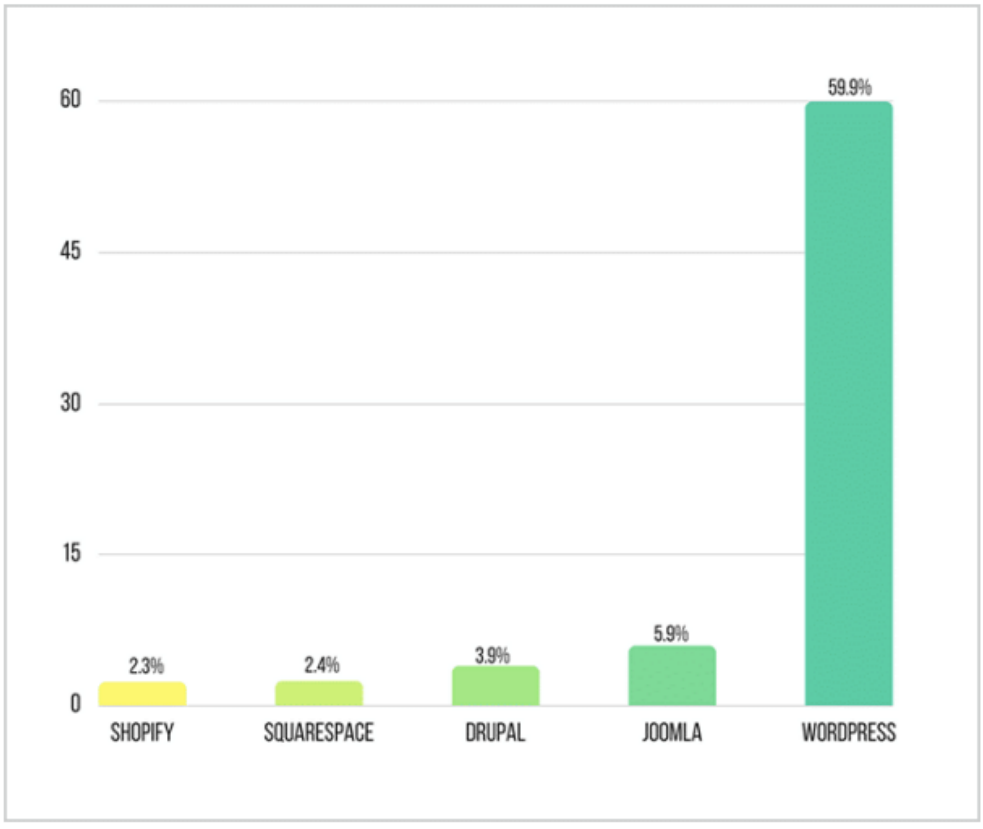


Рисунок 1.1 – Статистика використання різних фреймворків для розробки web-сайтів

WordPress написаний мовою гіпертекстового препроцесора PHP і працює в парі з базою даних MySQL або MariaDB. Функції включають архітектуру плагінів і систему шаблонів, які в WordPress називаються «Темами» [2].

Щоб працювати, WordPress має бути встановлено на веб-сервері як частину служби Інтернет-хостингу або на комп'ютері, на якому запущено програмний пакет WordPress.

WordPress був випущений 27 травня 2003 року його засновниками, американським розробником Меттом Малленвегом і англійським розробником Майком Літтлом.[1]. WordPress Foundation володіє WordPress, проектами WordPress та іншими пов'язаними товарними знаками. WordPress сьогодні підтримується кількома основними розробниками та багатьма ключовими учасниками. Майк Літтл керує спеціалізованим магазином WordPress zed1.com і час від часу додає патчі для коду. Компанія Метта Малленвега, Automattic, продовжує керувати хостингом wordpress.com, а також фінансування розробки відповідного вмісту та інструментів керування сайтом, у тому числі Akismet, Gravatar і останні плагіни, такі як JetPack. Akismet це надійна служба виявлення та захисту спаму, розміщена на сервері Automattic, зі статистичними (і неймовірно) низький рівень не виявлення [3].

Користувачі WordPress мають змогу установлювати багато різних тем і перемикатися між ними. Теми дозволяють користувачам змінювати вигляд і функціональність веб-сайту WordPress, не змінюючи основний код або вміст сайту. Спеціальний код можна додати на веб-сайт за допомогою дочірньої теми або через редактор коду. Кожен веб-сайт WordPress вимагає наявності принаймні однієї теми. Теми можна встановлювати безпосередньо за допомогою інструмента адміністрування "Вигляд" WordPress на інформаційній панелі, або папки тем можна скопіювати безпосередньо в каталог тем. Теми WordPress зазвичай поділяються на дві категорії: безкоштовні та преміум теми. Багато безкоштовних тем перераховано в каталозі тем WordPress (також відомому як репозиторій), але з захистом сайту від зловмисників можуть бути деякі проблеми, а преміум-теми можна придбати на торгових майданчиках і в окремих

розробників WordPress. Також є конструктори такі як Elementor, Divi, Beaver. Користувачі WordPress також можуть створювати та розробляти власні власні теми та завантажувати їх у каталог або репозиторій WordPress [3].

Архітектура плагінів WordPress дозволяє користувачам розширювати або знецінювати функції та функції веб-сайту чи блогу. Станом на грудень 2024 року WordPress.org має 70 756 доступних плагінів та стрімко збільшуєтсья, кожен із яких пропонує спеціальні функції та можливості, що дозволяє користувачам адаптувати свої сайти до своїх конкретних потреб. Однак це не включає доступні преміум-плагіни (приблизно 1500+), які можуть не бути в списку в репозиторії WordPress.org. Ці налаштування варіюються від оптимізації пошукових систем (SEO) до клієнтських порталів, які використовуються для відображення приватної інформації користувачам, які ввійшли в систему, до систем керування вмістом і функцій відображення вмісту, таких як додавання віджетів і панелей навігації. Не всі доступні плагіни завжди відповідають оновленням, і в результаті вони можуть не працювати належним чином або не працювати взагалі. Якщо розробник плагіна не тестував плагін з двома останніми основними версіями WordPress, у каталозі плагінів буде відображено попереджувальне повідомлення, яке інформує користувачів про те, що плагін може не працювати належним чином з останньою версією WordPress. Більшість плагінів доступні через сам WordPress, завантажуючи їх і встановлюючи файли вручну через FTP або через інформаційну панель WordPress. Однак багато третіх сторін пропонують плагіни на своїх веб-сайтах, багато з яких є платними пакетами, які часто тестуються та які мають підтримку для своїх клієнтів [3].

Веб-розробникам, які бажають розробляти плагіни, потрібно вивчити систему хуків WordPress, яка складається з понад 2000 хуків (станом на версію 5.7 у 2021 році), розділених на дві категорії: хуки - дій і хуки - фільтрів.

Плагіни також представляють собою стратегію розробки, яка може перетворити WordPress на різноманітні програмні системи та програми, обмежені лише уявою та креативністю програмістів. Вони реалізуються за допомогою користувальницьких плагінів для створення систем, не пов’язаних із

веб-сайтами, наприклад безголових програм WordPress і продуктів програмного забезпечення як послуги (SaaS) [4].

WordPress ядро складається з 3 папок wp-includes, wp-admin, wp-content і з кількох файлів поруч із цими папками.

Всі файли та папки, крім wp-content - це і є WordPress, двигун. Тобто. каталоги: wp-includes і wp-admin - це ядро WordPress, а wp-content – це папка з даними користувача сайту.

У директорії wp-content зберігаються практично всі файли користувача, крім файлу конфігурації wp-config.php (це невід'ємна частина ядра). Тут знаходяться плагіни, теми, файли плагінів, тем та вмісту сайту. Тут прийнято зберігати всі файли пов'язані з розширенням можливостей WordPress. Відповідно в WordPress, wp-content містить один файл index.php і 3 папки: plugins, themes, languages [4].

Файл wp-content/index.php завжди повинен існувати і повинен мати такий вміст:

Рисунок 1.2 – Вміст файлу index.php

Цей файл забороняє бачити список файлів у папці. Якщо index.php не існує, а ваш веб-сервер дозволяє дивитися файли в директоріях, пройшовши за посиланням http://example.com/wp-content, можна побачити всі файли та папки в цій директорії. Це можуть використовувати хакери, щоб отримати доступ до файлів ключів, що дозволить зламати сайт. Наприклад, якщо у вас встановлений вразливий плагін, то сайт можна буде легко перевірити на наявність цього вразливого плагіна, а далі атакуючий без особливих зусиль зможе зламати сайт.

Найважливішим файлом у будь-якій установці WordPress є файл wp-config.php. Цей файл містить абсолютно усі параметри підключення до бази

даних, включаючи ім’я бази даних, ім’я користувача та пароль для доступу вашу базу даних MySQL. Цей файл також зберігає додаткову базу даних та інші розширені WordPress налаштування [4].

Файл wp-config.php зазвичай зберігається в кореневому каталозі WordPress. Крім того, ви можете перемістіть файл wp-config.php із кореневого каталогу WordPress у батьківський каталог.

Також важливий файл .htaccess. Файл .htaccess використовується в основному для створення постійних посилань і URL-адрес із введенням ключових слів для вашого сайту в основному формується на заголовках сторінок, пості, категорій, та інше. WordPress за замовчуванням створює потворні URL-адреси у вигляді рядка запиту, зазвичай з ідентифікатором присутні, наприклад http://example.com/?p=45. Ці URL-адреси повністю функціональні, але не дуже гарно для пошукових систем і відвідувачів сайту. Увімкнувши постійні посилання на основі заголовків, тегів чи архівів [4].

Використання постійних посилань дає багато переваг, наприклад:

- Пошукова оптимізація (SEO) – ключові слова у вашій URL-адресі можуть використовуватися на вашому веб-сайту для підвищення SEO. Пошукові системи використовуватимуть ці ключові слова у своєму алгоритмі для позиціонування в своїх результати пошуку;
- Зручність використання – незручні для відвідувачів URL-адреси ID роблять так само незручним обмін посиланням із друг. Важко відрізнити вміст URL-адрес, керованих ідентифікаторами;
- Спільний доступ – у цю Інтернет-еру соціальних мереж обмін є природним розширенням вашого онлайн присутність. Ключові слова в URL-адресі зроблять пошук вашого посилання надзвичайно простим передати безпосередній контекст для вмісту, що боти мали змогу гарно зчитувати інформацію, та видавати його в результату пошуку [4].

Файл functions.php містить головні функції WordPres. Цей файл використовується для легкого налашування теми плагіну в WordPress за

допомогою стандартизованого методу. Плагіни, теми та WordPress core всі можуть використовують ці функції, чи подібні:

- current_time() – Отримує поточний час на основі вказаного типу;
- force_ssl_login() – потрібен SSL (https) вхід у WordPres;
- wp_nonce_field() –додає одноразове приховане поле для форм. Поле nonce використовується для перевірки під час надсилання та обробки даних у WordPress. Це критично важливий крок захист вашого коду;
- absint() –перетворює значення на невід'ємне ціле число;
- wp_die() – Зупиняє виконання WordPress і відображає повідомлення про помилку HTML.

Файл user.php містить функції API користувача WordPress, наприклад:

- get_users() – повертає список користувачів, які відповідають наданим критеріям
- add_user_meta(), get_user_meta(), delete_user_meta() – використовується для створення, отримання та видалити метадані користувача;
- username_exists() – перевіряє, чи існує ім'я користувача;
- email_exists() – Перевіряє, чи існує адреса електронної пошти;
- wp_insert_user() і wp_update_user() – Створення й оновлення облікового запису користувача.

Mu-plugins - обов'язкові плагіни. У WordPress є "Обов'язкові плагіни", вони знаходяться в директорії wp-content/mu-plugins.

Коротко про обов'язкові плагіни: обов'язкові для використання плагіни (Must-use plugins) – це плагіни, які встановлюються у спеціальну папку mu-plugins у каталозі контенту wp-content та активуються автоматично (завжди активні) для сайту та сайтів мережі. Ці плагіни не видно серед звичайних плагінів. В адмін-панелі вони відображаються у верхньому інформаційному рядку і їх неможливо відключити, крім видалити файл плагіна з каталогу wp-content/mu-plugins.

/plugins – плагіни. Плагіни знаходяться у директорії wp-content/plugins. Плагін може являти собою один файл або кілька файлів усередині папки. Будь-

які файли в директорії /plugins скануються WordPress, щоб визначити, чи файл є файлом плагіна. Якщо файл визначається як плагін, він з'являється в адмін-панелі розділу «Плагіни» і готовий до активації.

1.2 Схема бази даних Wordpress

Стандартна інсталяція WordPress містить 12 таблиць бази даних. В WordPress є дуже легка і гарно пропрацьована база даних. Якщо вона не напружена додатковими плагінами, наприклад WooCommerce. Структура бази даних розроблена таким чином, щоб бути дуже мінімальною, але при цьому допускати безмежну гнучкість під час розробки та дизайн для WordPress. Щоб зрозуміти схему бази даних, це допоможе переглянути базу даних діаграма. На рисунку 1.2 наведено огляд структури бази даних WordPress і таблиць створений під час стандартної інсталяції WordPress. Майте на увазі, що плагіни та теми мати можливість створювати власні таблиці. WordPress Multisite також створює додаткові таблиці тому ваша база даних WordPress може містити більше таблиць, ніж просто WordPress за замовчуванням таблиці [5].

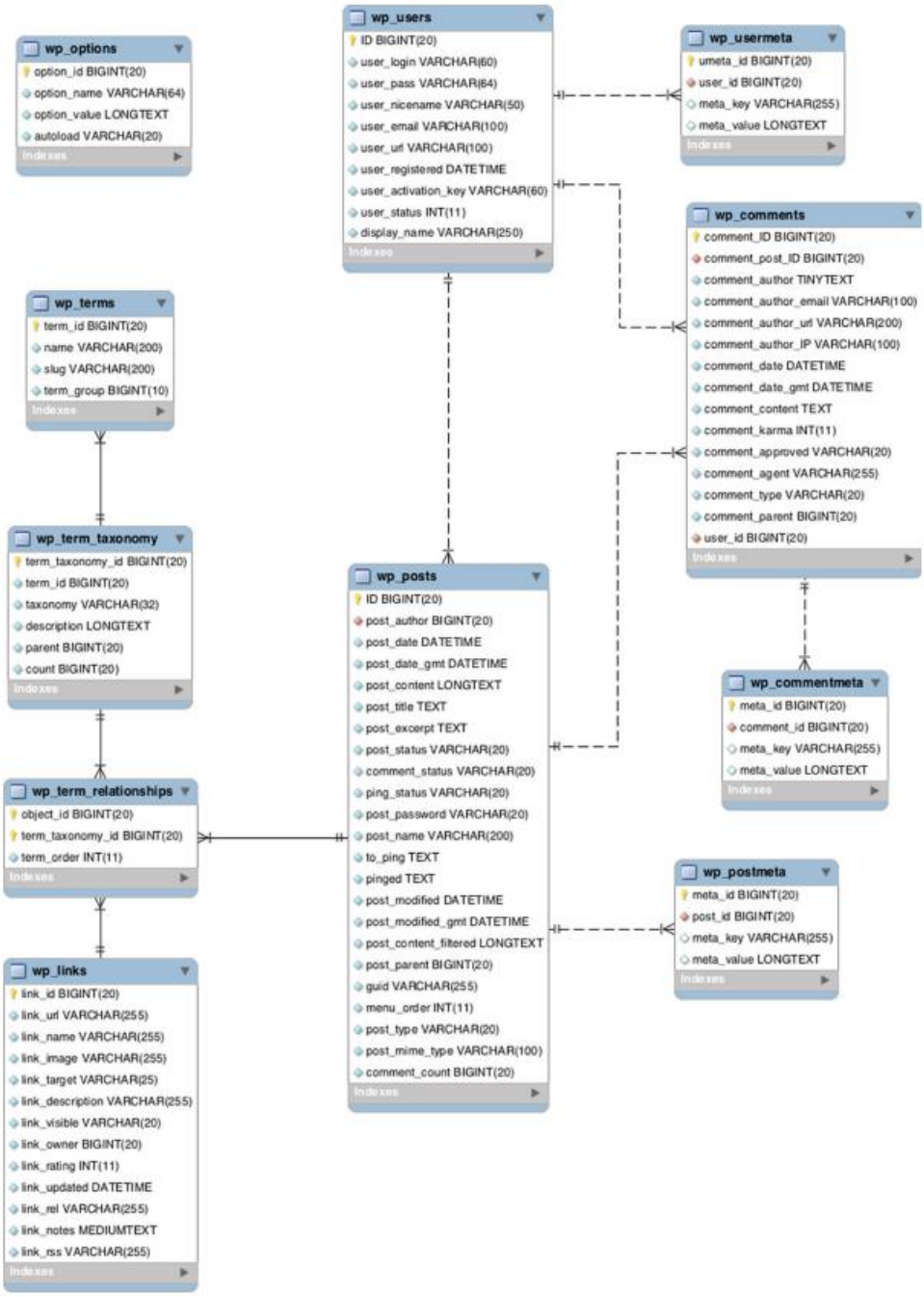


Рисунок 1.3 – Wordpress діаграма бази даних

Структура таблиці в WordPress дуже послідовна база даних. Кожна таблиця у вашій базі даних містить унікальне поле ID, яке є первинним ключем таблиці. Кожна таблиця також містить один або кілька індексів на полів, що удосконалює швидкість отримання даних під час виконання запитів до даних.

Найважливішим полем у кожній таблиці є поле унікального ідентифікатора. Це поле не завжди має назву ID, але це поле з автоматичним збільшенням, яке використовується для надання кожному запису в таблиці унікального ідентифікатора. Ось для прикладу, коли ви вперше встановлюєте WordPress, створюється повідомлення за замовчуванням під назвою «Hello world!» Тому що це перша публікація, створена в таблиці wp_posts, ідентифікатор цієї публікації дорівнює 1. Кожній публікації надається унікальний ідентифікатор яке можна використовувати для завантаження постспецифічної інформації, а також як поле об'єднання інші таблиці в базі даних. Є одне застереження щодо цього, яке стосується редакцій дописів, вкладень і користувацьких типів дописів. Кожен із цих записів зберігається як новий запис у таблиці wp_posts, тому кожен з них отримує власний унікальний ідентифікатор, що означає, що ваші ідентифікатори опублікованих публікацій можуть не бути послідовними. Наприклад, ваш перший пост може мати ідентифікатор 1, тоді як ваша друга публікація може мати ідентифікатор 15. Все залежить від того, скільки між кожною публікацією було створено додаткові записи [6].

На даний момент для WordPress було створено 12 таблиць бази даних. Нижче наведені список цих таблиць і докладно про те, які дані вони зберігають:

- wp_commentmeta – містить усі метадані для коментарів;
- wp_comments – містить усі коментарі в WordPress. Окремі коментарі посилаються повернутися до публікацій через ідентифікатор публікації;
- wp_links – містить усі посилання;
- wp_options – зберігаються всі параметри веб-сайту, визначені на екрані налаштувань. Також зберігає плагін параметри, активні плагіни та теми тощо.
- wp_postmeta – містить усі метадані публікації (настроювані поля);
- wp_posts – містить дописи всіх типів (за замовчуванням і спеціальні типи дописів), сторінки, медіа-записи, та перегляди. Зазвичай у більшості випадків це найбільша таблиця в базі даних;

- wp_terms – містить усі таксономічні терміни, визначені для вашого веб-сайту, зіставляючи їхній текст описи до номерів термінів, які можна використовувати як унікальні індекси в інших таблицях;
- wp_term_relationships – об’єднує терміни таксономії з вмістом, забезпечуючи членство стіл. Він відображає такий термін, як тег або назва категорії, на сторінку чи публікацію, яка на нього посилається;
- wp_term_taxonomy – визначає таксономію, до якої віднесено кожен термін. Ця таблиця дозволяє мати категорії та теги з однаковими назвами, розміщуючи їх у різних назвах таксономії;
- wp_users – містить усіх користувачів, створених на вашому сайті (логін, пароль, електронна пошта, та інші дані);
- wp_usermeta – містить метадані для користувачів (ім’я/прізвище, псевдонім, тощо).

2 ПОСТАНОВКА ЗАДАЧІ ДЛЯ СТВОРЕННЯ ВЕБСАЙТУ

2.1 Постановка задачі

Дана кваліфікаційна робота має на меті реалізувати вебсайт для компанії яка володіє одним із найбільшим асортиментом шин та дисків в Україні «ShinaHub». Замовником даного вебсайту є компанія під назвою «ShinaHub». Даний вебсайт повинен забезпечувати наявність інформації про наявність автомобільних шин та дисків, його послуги, інформацію про товари, можливість замовлення послуги та товарів, можливість підтримки ведення бухгалтерського обліку.

В даному вебсайті повинна бути змога додавання нових товарів за допомогою завантаження їх у вигляді таблиці Exel. Даний вебсайт повинен бути размещений на хостингу. Вебсайт повинен задовольняти весь набір функціональних побажань замовника, які прописані у технічному завданні.

Сайт повинен бути доступним в мережі Інтернет, щоб пошукові роботи видавали його в результаті запросу. Сайт повинен мати чітку структуру, та весь перерахований функціонал.

Для підтримки працездатності інформаційної системи потрібні базові навички роботи з даною інформаційною системою. Також, користувач повинен мати загальні навички роботи з персональним комп'ютером і стандартними web-браузерами.

Стиль сайту має бути лаконічним та сучасним, кольорове рішення – це чорний та жовтий колір. Фон має бути білий, щоб не відволікав від потенційно важливої інформації. Вебсайт повинен бути простим в користуванні для користувача. Всі фото, які будуть розміщені в інформаційній системі повинні мати гарну якість, та бути в новітньому форматі (.webp).

Інформаційна система повинна складатися з наступних розділів:

- сторінка Головна – має містить фільтр товарів, рекомендовані товари, бренди, блок консультації, відгуки клієнтів, контактні дані;
- сторінка Доставка і оплата– містить інформацію про доставку і оплату;
- сторінка Товар – містить інформацію про товари та ціну;
- сторінка Шини та Диски – містить всі товари;
- сторінка Наша історія – містить дані про компанію замовника;
- сторінка Відгуки – містить всі коментарі клієнтів;
- сторінка Контакти – містить контактні дані компанії;
- сторінка Гарантії – містить опис гарантій;
- сторінка Поверненя – містить опис повернення товару;
- сторінка Кошик – містить інформацію про кількість обраного товару.

Програмне забезпечення клієнтської частини має підтримувати наступним вимогам: Web-браузер: Internet Explorer 9.1 і вище, або Firefox 3.6 і вище, або Opera 11 і вище, або Safari 3 і вище, або Chrome 11 і вище.

Вебсайт повинен надвати наступні функціональні вимоги:

- підбір необхідного товару за характеристиками;
- перегляд актуального прайсу;
- перегляд товарів;
- змога замовлення товару;
- перегляд контактної інформації;
- ведення статистики продаж;
- гнучкий контент під телефон, планшет та десктопу;

Всі необхідні матеріали для наповнення сайту має надати замовник у відповідний термін. А саме: таблиці з даними для наповнення ними товарів в магазині, тексти для контактної інформації, та макет сайту.

2.2 Дизайн сайту

Головна сторінка сайту містить: шапку сайту, фільтир для товарів, рекомендовані товари, бренди з якими співпрацює компанія, зворотній зв’язок, переваги компанії, відгуки клієнтів, контактна інформація, та футер сайту.

Сторінка каталогу товарів містить: сам каталог товарів ,та фільтр.

Сторінка “Доставки” містить текстову інформацію, яка стосується доставки.

Сторінка “Про нас” містить: текстове наповнення, та блок з брендами з якими співпрацює клієнт.

Сторінка “Відгуки” містить відгуки клієнтів, та картинки брендів з якими співпрацює клієнт.

Сторінки “Гарантії” та “Поверненя” мають тільки текстове наповненя, тож дизайн сторінки надається тільки для сторінки “Гарантії”.

2.3 Вибір для реалізації вебсайту

Після постановки основних задач та функцій вебсайту було обрано засоби реалізації. Під час обрання системи управління базою даних було розглянуто два варіанти: MySQL та MariaDB. Для співпраці з вебсайтом необхідна швидка та проста база даних, яку можна налаштувати та керувати, щоб вона нам відповідала всім необхідним. База даних повинна бути добре захищеною, надійною та зрозумілою. Саме цьому, була обрана база даних реляційна система управління базами даних MySQL. Оскільки вона максимально підходить для розробки вебсайту та задовольняє всім вищеперерахованим критеріям та майже всі хостинги підтримують цю базу даних. MySQL є однією з найпопулярнішою у світі базою даних з відкритим вихідним кодом [8].

Слід замітити, що вона забезпечує комплексну підтримку для будь-яких потреб додатка. Слід відзначити, що база даних MariaDB є

багатофункціональною базою даних, яка може обробляти більш складніші запити та масивні бази даних.

Для реалізації вебсайту були обрані популярні інструменти веб-розробки: HTML5, CSS3, JavaScript, PHP. HTML - скорочення від "HyperText Mark-up Language" - перекладається як "Мова розмітка гіпертексту" (Гіпертекст - це текст, що не послідовно зв'язаний з іншими документами, тобто у вас є змога з першої сторінки документу перейти на останню). Іншими словами HTML - це мова розмітки, або ще один спосіб зберігання інформації. CSS (абревіатура від Cascading Style Sheets, що в перекладі означає каскадні таблиці стилів) - це спеціальна мова (мова стилів), за допомогою якої описують вигляду документів (як і де відображати елементи веб-сторінки), написаних мовами розмітки даних. Найчастіше CSS використовується для документів, котрі розмічені мовою HTML, XHTML. JavaScript – мова програмування, яка найчастіше використовується для створення інтерактивних Web-сторінок. Дозволяє реалізувати ряд складних рішень в Web-документах і надає можливість на боці клієнта взаємодіяти з користувачем. PHP (англ. PHP: Hypertext Preprocessor - "PHP: препроцесор гіпертексту") - скриптова мова програмування, застосовується для розробки додатків та сайтів. На даний час він підтримується майже всіма хостинг-компаніями і є одним із лідерів серед мов програмування, які застосовуються для написання динамічних веб-сторінок.

2.3 Вибір програмного забезпечення

Для комфортної розробки вебсайту на базі Wordpress були обрані наступні програмне забезпечення:

- Local – це програмний застосунок, який допомагає розвернути локально вебсайт. Він був обраний для комфортної розробки в локальному простору;
- PHPStorm – комерційне крос-платформове інтегроване середовище розробки для PHP, HTML, CSS, JS та інші, яке розробляється компанією JetBrains на основі платформи IntelliJ IDEA.

Це необхідний мінімум для розробки вебсайту. На рисунку 4.1 зображений програмний застосунок Local.

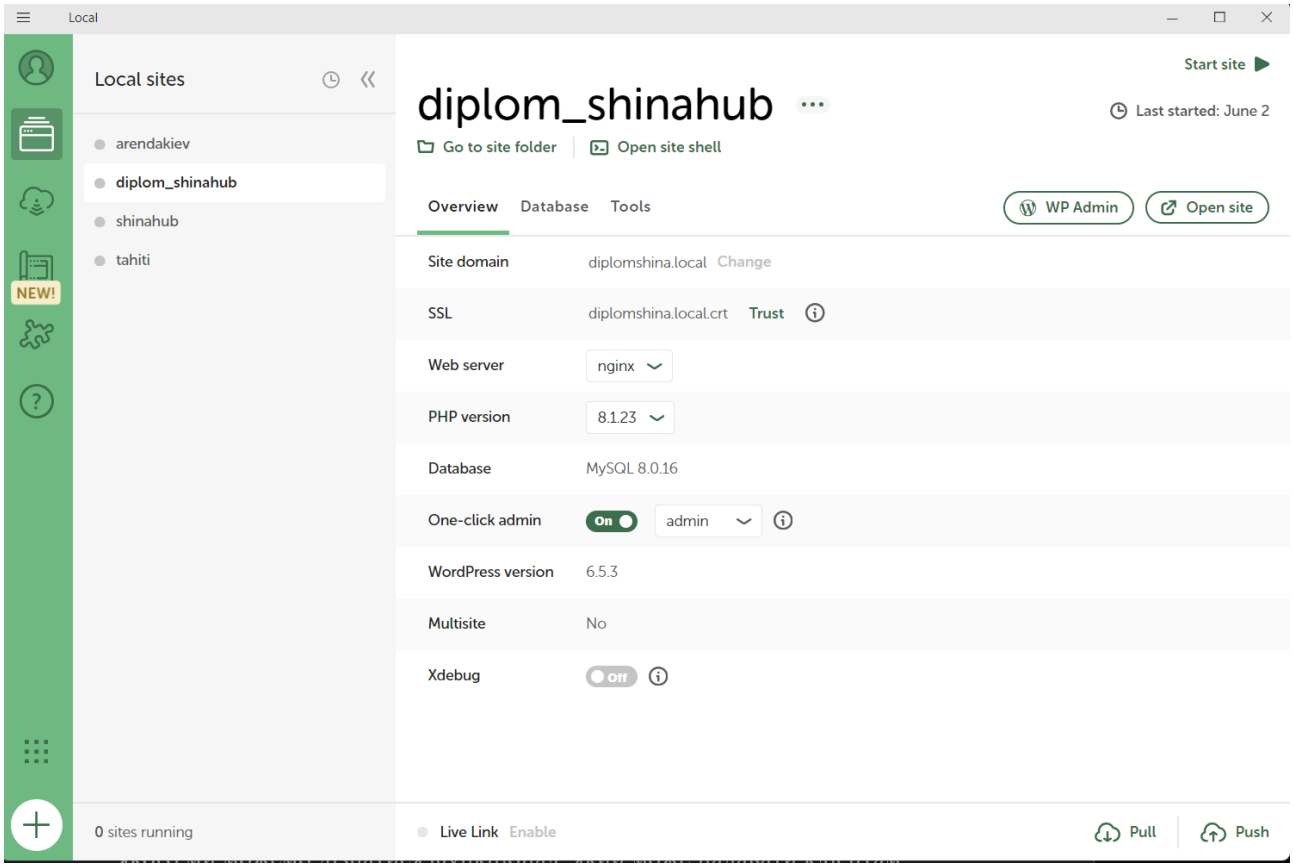


Рисунок 4.1 – Програмний застосунок Local

PHPStorm спеціалізується саме на PHP, що забезпечує повну підтримку мови, також можемо підключити бібліотеку з Wordpress функціями, що прискорить розробку, включаючи інтелектуальне автозаповнення коду, навігацію по коду, рефакторинг та інші функції, які значно спрощують розробку. Інтерфейс даного програмного застосунку зображено на рис. 4.2.

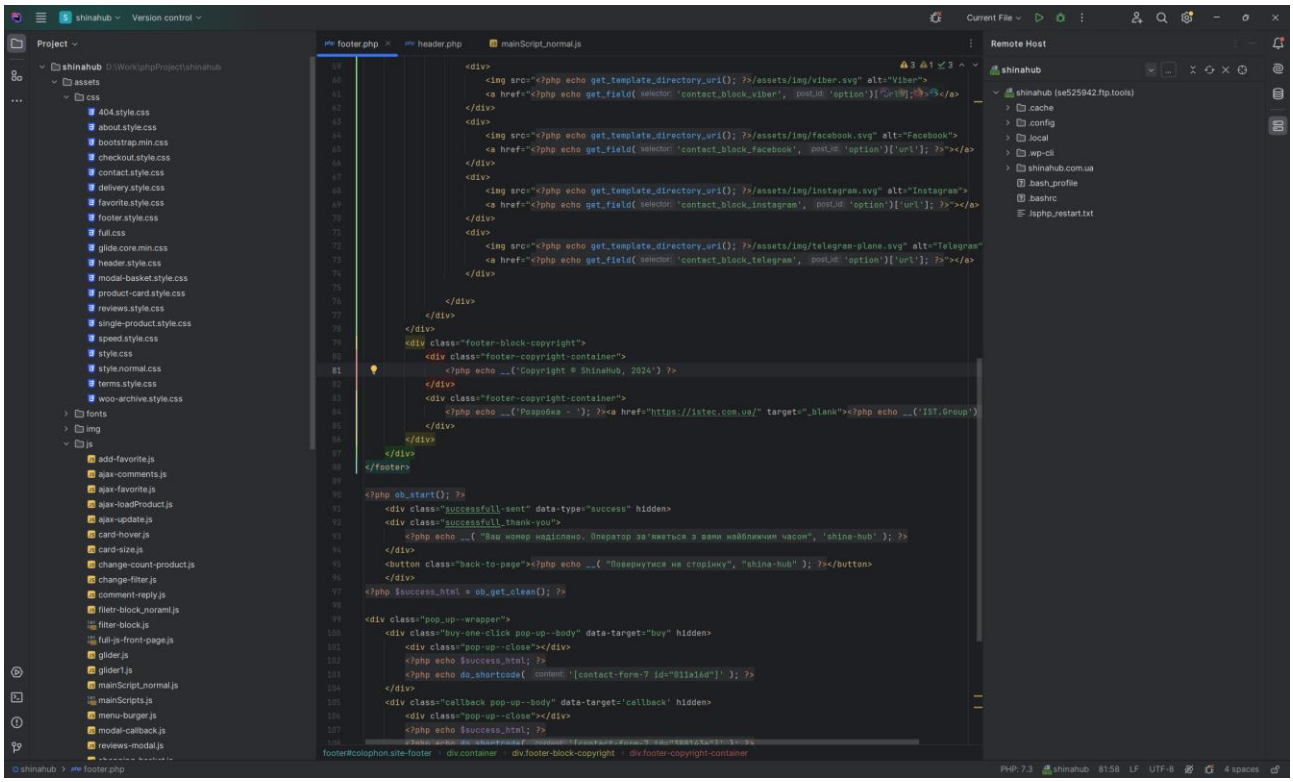


Рисунок 4.2 – Інтерфейс програмного застосунку PHPStorm

Для розробки кастомної теми для вебсайту на базі WordPress потрібні деякі інстументи для комфортної роботи. Та також технічне завдання, макет сайту, з якого ми можемо дізнатися функціонал, який може надавати клієнтам.

В нашому випадку це сайт, який буде використовувати плагін WooCommerce. Більш детально про WooCommerce. WooCommerce – це один із найпопулярніших плагінів для WordPress, який перетворює вебсайт на потужний інтернет-магазин. Він надає всі необхідні інструменти для управління електронною комерцією, від управління товарами до обробки замовлень та платежів.

3 МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ ПІДТРИМКИ ДІЯЛЬНОСТІ ВЕБСАЙТУ

3.1 Моделювання інформаційної системи в IDEF0

Після закінчення етапу визначення основних функцій та задач кваліфікаційної роботи магістра було виконано структурно-функціональне моделювання вебсайту для підтримки діяльностіцього проекту. Під час структурно-функціонального моделювання була розроблена діаграма в IDEF0.

Головним принципом IDEF0 є покрокове розбиття основного процесу створення вебсайту на окремі підпроцеси. Дана діаграма повинна дати смогу враховувати всі зв’язки між різними підпроцесами. Одним з головних додаткових опцій розробки структурно-функціонального моделювання, що під час побудови діаграми ми можемо знайти слабкі місця вебсайту та виправити їх, до переносу вебсайту на клієнтський хостинг.

Проаналізувавши розроблювану інформаційну систему було визначено наступні дані:

- Вхід: характеристики товару.
- Вихід: вид сформованного замовлення;
- Процес: замовлення товару;
- Управління: вимоги замовника, інформація про товар;
- Механізми: база даних.

Згідно з отриманими даними була розроблена діаграма IDEF0 (рис.3.1) та її декомпозиція (рис. 3.2).

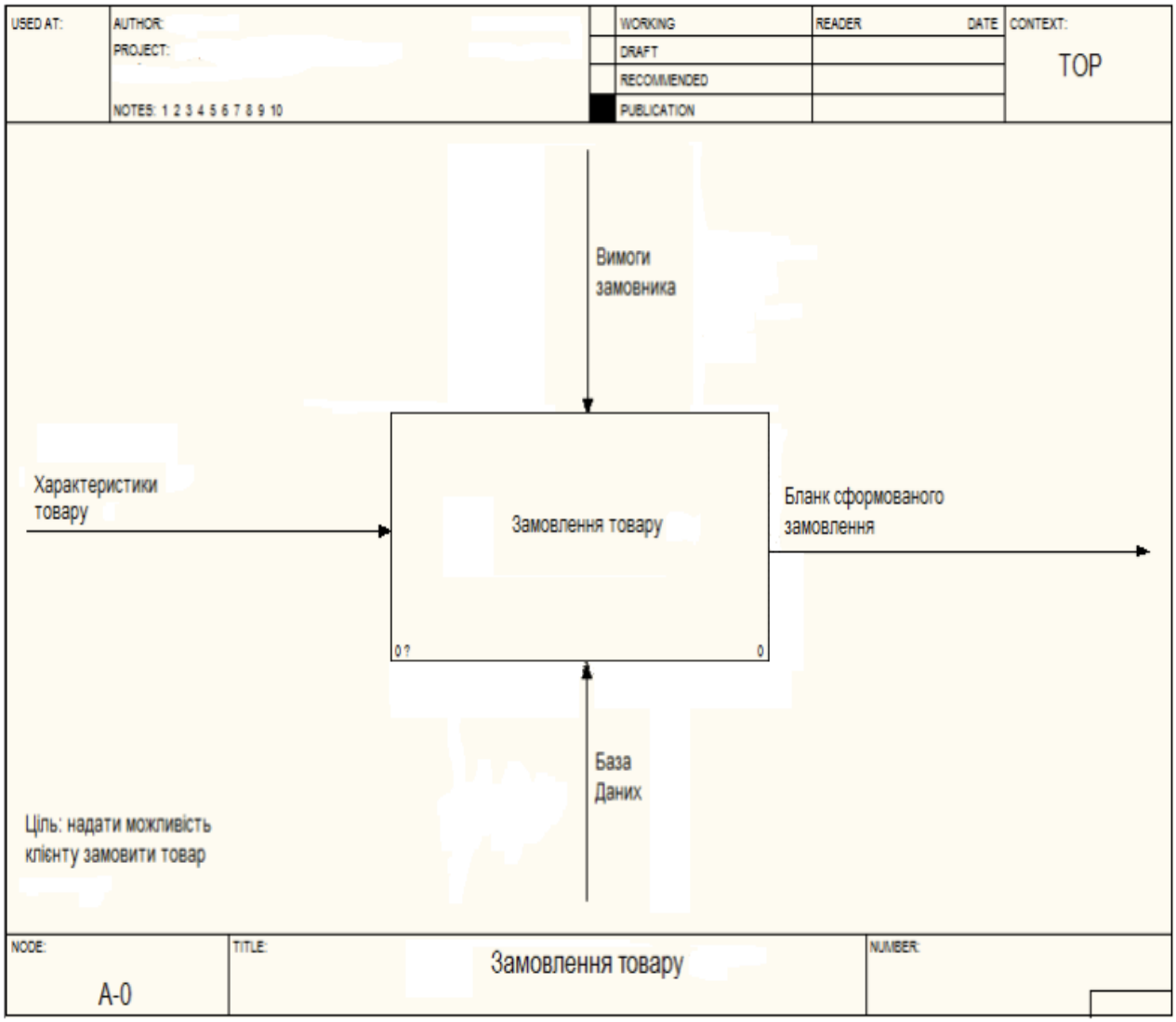


Рисунок 3.1 – Функціональне моделювання інформаційної системи в IDEF0

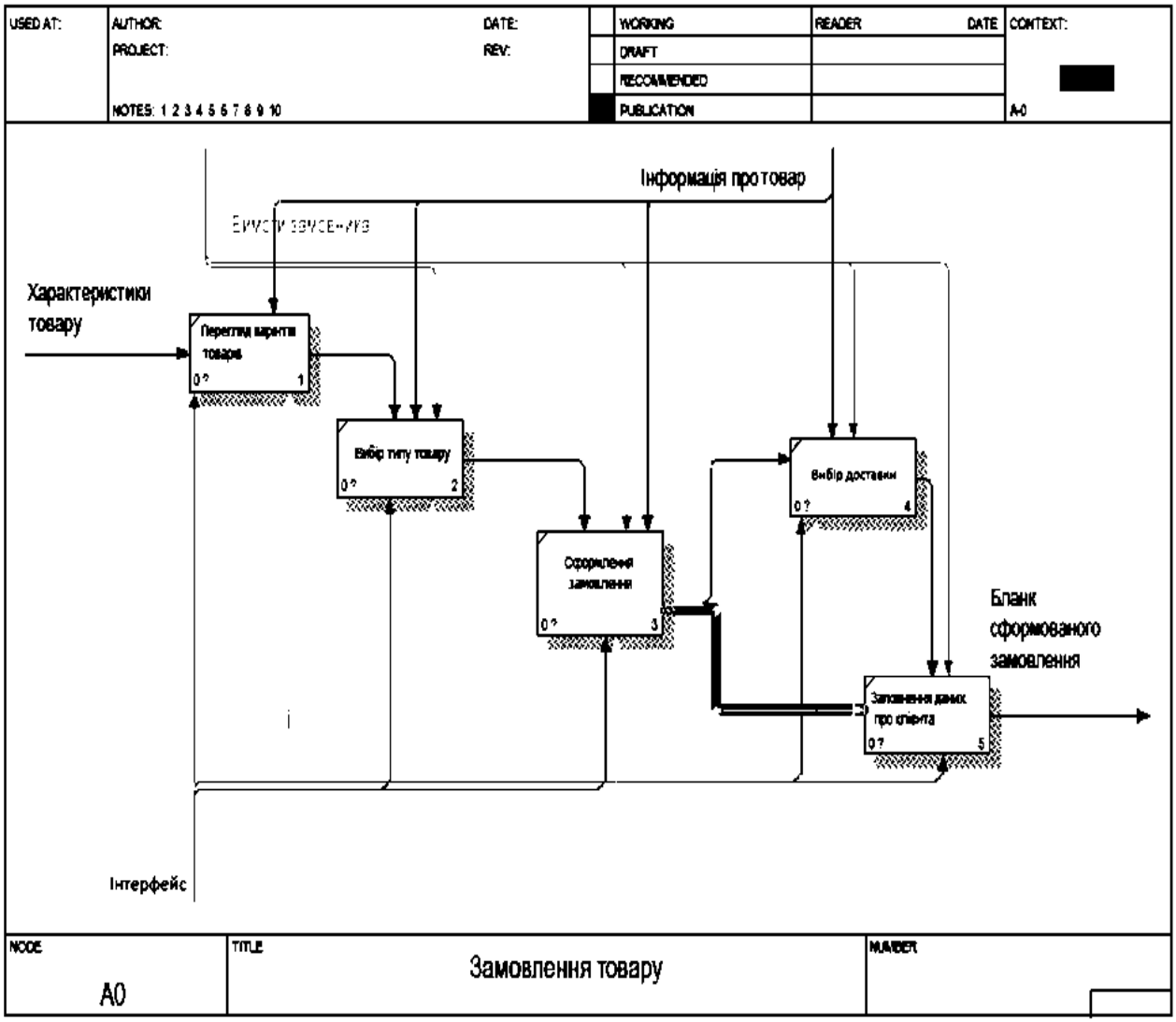


Рисунок 3.2 – Декомпозиція діаграм IDEF0

3.2 Моделювання всіх варіантів використання

Під час розробки кваліфікаційної роботи магістра була розроблена діаграма всіх варіантів використання для вебсайту. Дана діаграма допомагає зрозуміти роботу вебсайту. Наприклад, зображення всіх можливих взаємодій клієнтів сайту, та замовника з вебсайтом та безпосередньо з відвідувачем вебсайту. Діаграма варіантів використання проілюстрована на рис. 3.3.

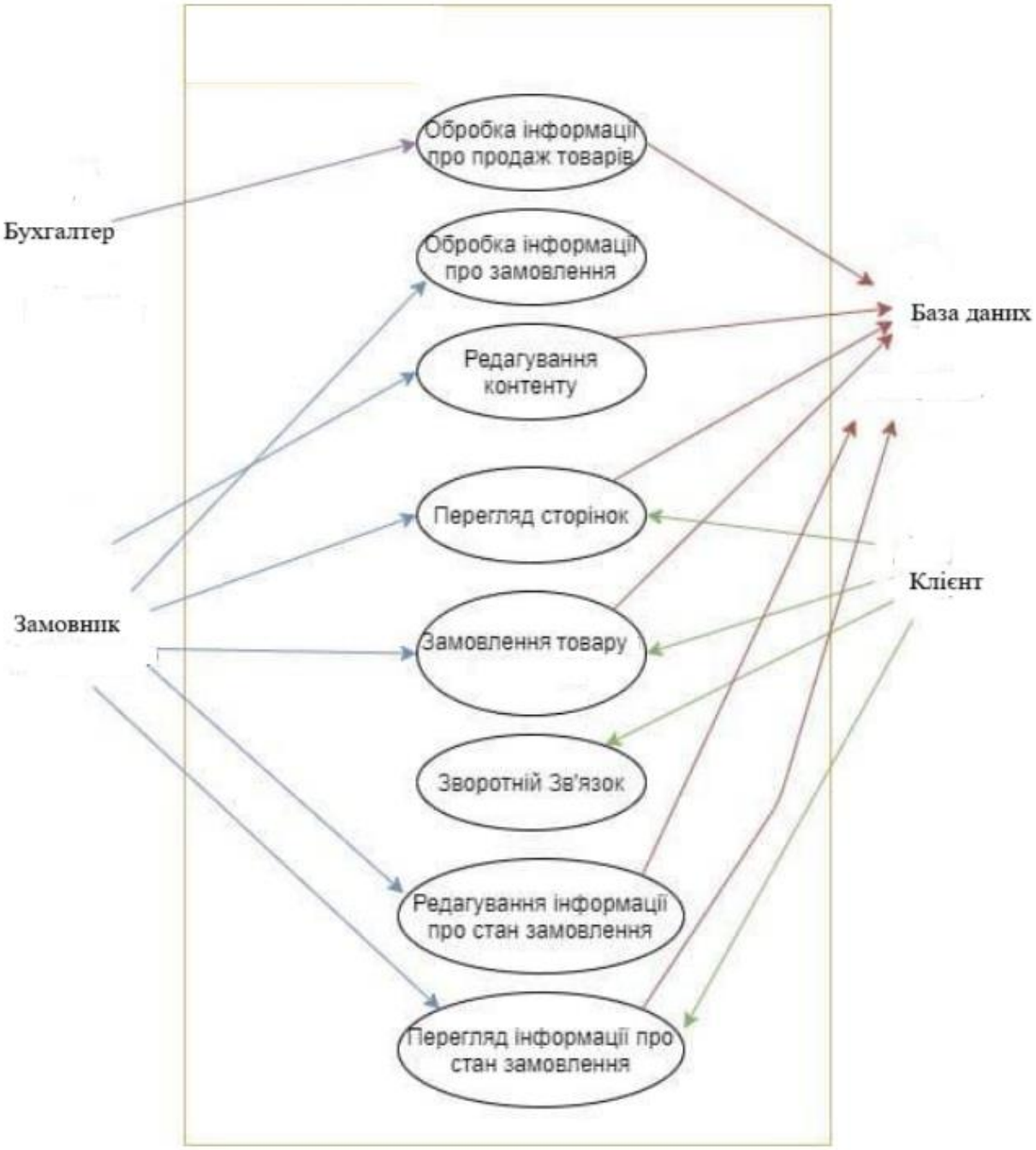


Рисунок 3.3 – Діаграма варіантів використання вебсайту різними відвідувачами

Для даної інформаційної системи було визначено наступних акторів:

- Адміністратор – має доступ до всього функціоналу вебсайту;
- База Даних – зберігає інформацію;
- Клієнт – має доступ до клієнтського функціоналу сайту;
- Бухгалтер – має доступ до інформації про замовлення.

4 РОЗРОБКА ВЕБСАЙТУ

4.1 Розробка хедеру сайту та футеру сайту

Для розробки хедеру та футеру сайту нам потрібно мати навички HTML, CSS, JavaScript, та PHP. Результати які ми отримали після розробки цих елементів зображені на рис. 4.1 – 4.2.

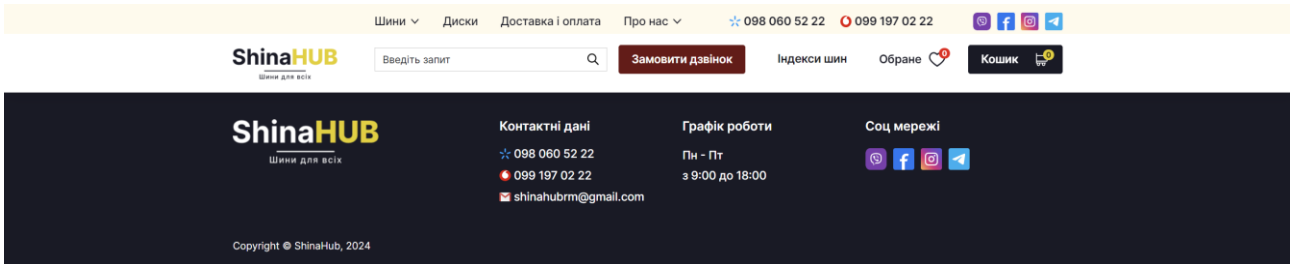


Рисунок 4.1 – Хедер та футер сайту на десктопі

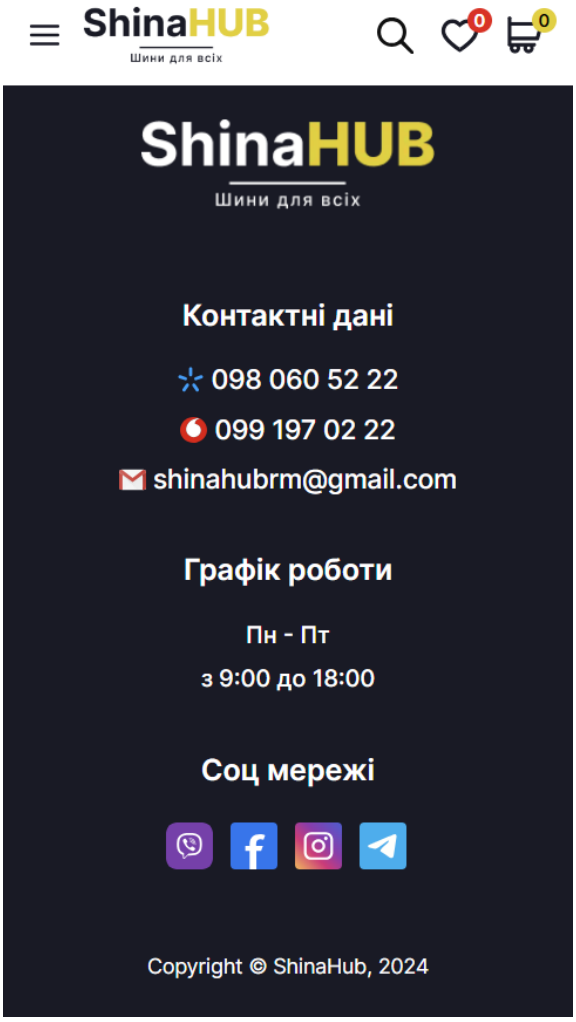


Рисунок 4.2 – Хедер та футер на мобільному пристрої

Для отримання такого результату був розроблений код, який зображений на рис. 4.3 – 4.5.

```php
<?php
1 usage
function my_theme_enqueue_styles()

{
    wp_enqueue_style( handle: 'bootstrapMin',  src: get_template_directory_uri() . '/assets/css/bootstrap.min.css');
    wp_enqueue_style( handle: 'headerStyle',  src: get_template_directory_uri() . '/assets/css/header.style.css');
    wp_enqueue_style( handle: 'footerStyle',  src: get_template_directory_uri() . '/assets/css/footer.style.css');
    wp_enqueue_style( handle: 'glideCore',  src: get_template_directory_uri() . '/assets/css/glide.core.min.css');
}

add_action('wp_enqueue_scripts', 'my_theme_enqueue_styles');


?>

<?php wp_head(); ?>
```

Рисунок 4.3 – Підключення стилів в шапку сайту

Підключення стилів до сайту за допомогою php. Також можна підключити до function.php, але в такому вигляді легше контролювати підключення.

```php
<header class="header">
    <div class="header-container">
        <div class="header-menu-containerUpBg">
            <div class="container">
                <div class="header-menu-container-up">
                    <?php wp_nav_menu(array(
                        'theme_location' => 'head_menu',
                        'menu_class' => 'header-menu',
                        'menu_id' => 'header-menu-id-up'
                    )); ?>
                    <div class="header-phone">
                        <a href="<?php echo get_field( selector: 'contact_block_number_1',  post_id: 'option')['url']; ?>">
                            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/logo_kievstar.svg"
                                 alt="Київстар">
                            <?php echo get_field( selector: 'contact_block_number_1',  post_id: 'option')['title']; ?></a>
                        <a href="<?php echo get_field( selector: 'contact_block_number_2',  post_id: 'option')['url']; ?>">
                            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/logo_vodafone.svg"
                                 alt="Водафон">
                            <?php echo get_field( selector: 'contact_block_number_2',  post_id: 'option')['title']; ?></a>
                    </div>
                    <div class="header-block-social">
                        <div class="contact-social-card">
                            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/viber.svg" alt="Viber">
                            <a href="<?php echo get_field( selector: 'contact_block_viber',  post_id: 'option')['url']; ?>"></a>
                        </div>
                        <div class="contact-social-card">
                            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/facebook.svg"
                                 alt="facebook">
                            <a href="<?php echo get_field( selector: 'contact_block_facebook',  post_id: 'option')['url']; ?>"></a>
                        </div>
                        <div class="contact-social-card">
                            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/instagram.svg"
                                 alt="instagram">
                            <a href="<?php echo get_field( selector: 'contact_block_instagram',  post_id: 'option')['url']; ?>"></a>
                        </div>
                        <div class="contact-social-card">
                            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/telegram-plane.svg"
                                 alt="Telegram">
                            <a href="<?php echo get_field( selector: 'contact_block_telegram',  post_id: 'option')['url']; ?>"></a>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
```

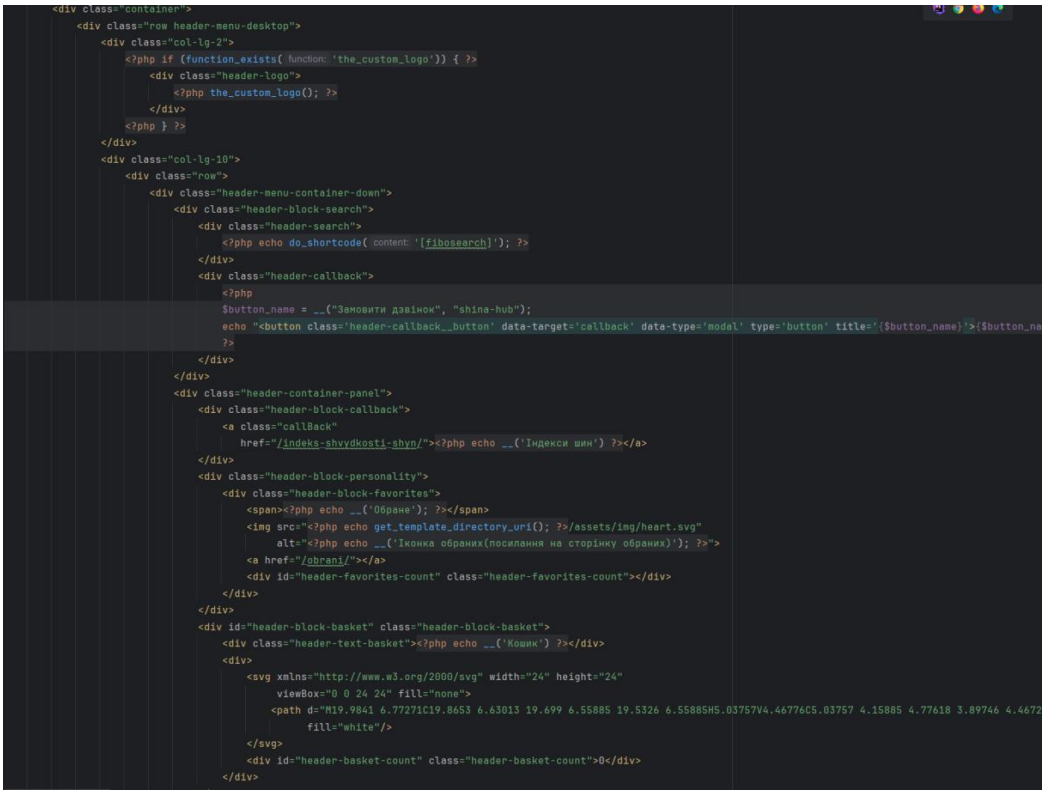
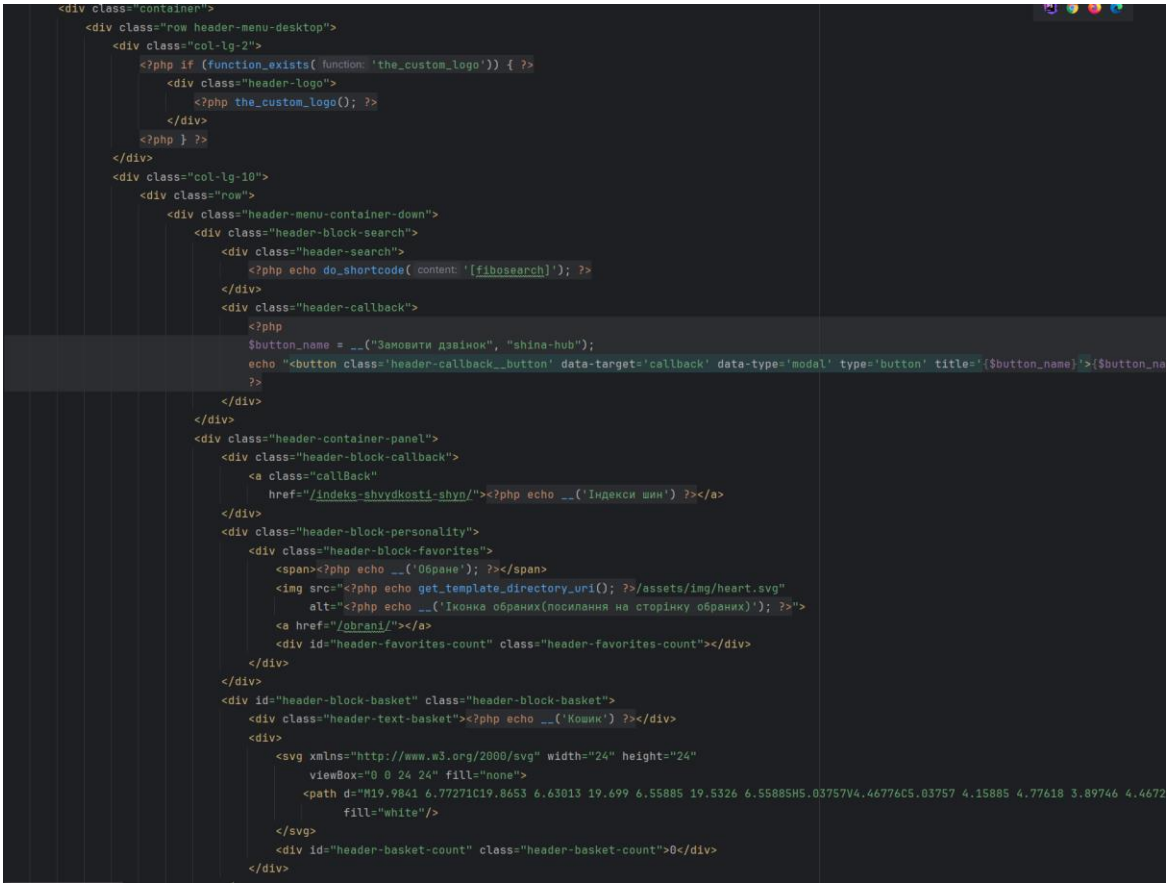


Рисунок 4.4 – Частина хедеру сайту



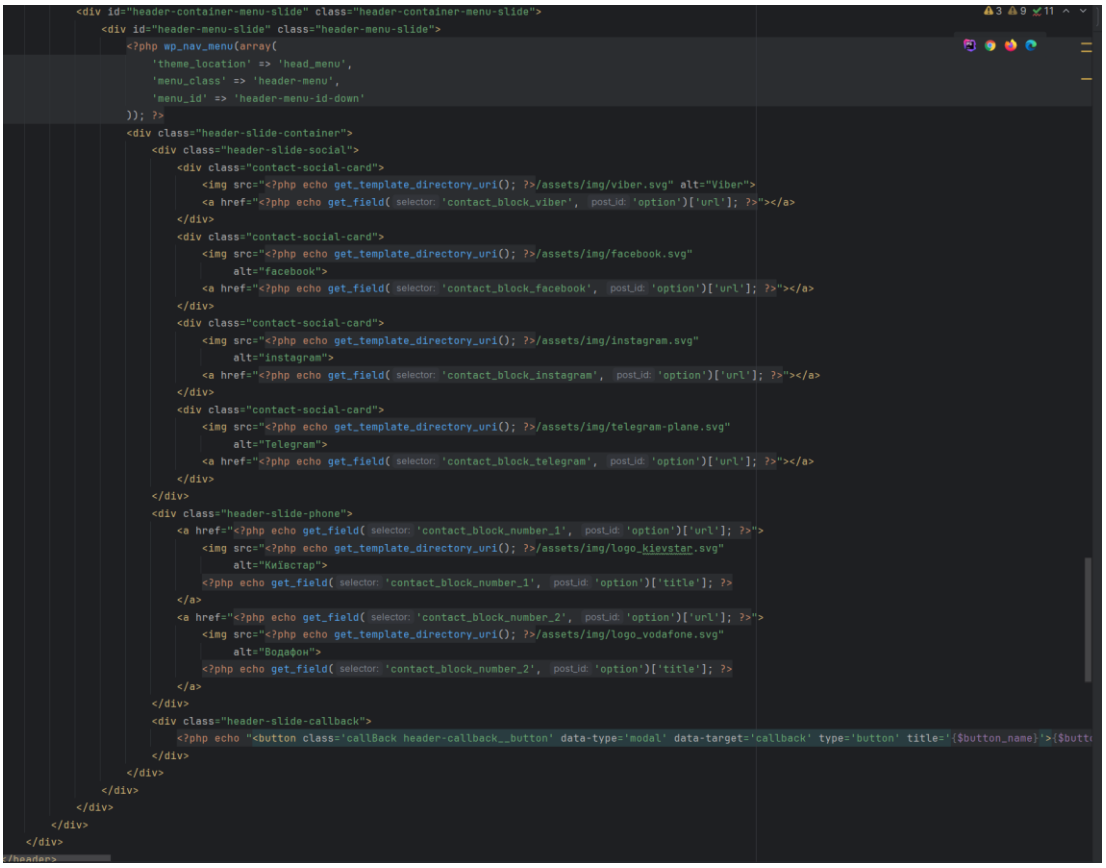


Рисунок 4.5 – Друга частина хедеру сайту

З коду можна зауважити, що використовується стандартий HTML 5 та функції WordPress. А саме використовуються:

- get_template_directory_uri() – використовується для отримання шляху до корня паки з темою, щоб не записувати його статично, так як сайту мсоже змінити домен, та путі перестануть працювати;
- get_field() – використовується для отримання поля, яке створене в плагіні ACF. Він використовується, щоб можна інформацію прописувати в адмінці та виводиити динамічно на сайт;
- wp_nav_menu() – вивід меню, яке створено в адмінці сайту;
- the_custom_logo() – вивід логотипу сайту, яке було додане в адмінку сайту;
- do_shortcode() – вивід шорткоду.

Також в шапці сайту було описано вивід корзини сайту, для зручного контролю кількості товару. Це зображено на рисунку 4.6. Для виводу картки для товарів використовується функція WooCommerce woocommerce_mini_cart.

```
<div id="modal-shopping-basket-container" class="modal-shopping-basket-container">
    <div id="modal-shopping-basket" class="modal-shopping-basket">
        <div class="modal-shopping-header">
            <h2 class="modal-shopping-title"><?php echo __('Кошик'); ?></h2>
            <div id="modal-shopping-close" class="modal-shopping-close">
                <svg xmlns="http://www.w3.org/2000/svg" width="28" height="28" viewBox="0 0 28 28" fill="none">
                    <path d="M26 2L14 14M2 26L14 14M14 14L2 2M14 14L26 26" stroke="#191A26" stroke-width="2"
                          stroke-linecap="round"/>
                </svg>
            </div>
        </div>
        <div class="modal-shopping-content">
            <?php woocommerce_mini_cart(); ?>
        </div>
    </div>
</div>
```

Рисунок 4.6 – Вивід корзини

Також для правильної роботи корзини необхідно написати JavaScript. Для откривання копзини та додавання товарів за допомогою AJAX запросів. AJAX (Asynchronous JavaScript And XML) – це такий підхід до створення користувацьких інтерфейсів вебзастосунків, за яких вебсайт, не перезавантажуючись, у фоновому режимі надсилає запити на сервер і сама звідти отримує відповідь, та обробляє цю відповідь. У нас необхідно реалізувати. Додавання, та зменшення кількості товарі. Також до цього обновляти загальну суму товарів, та ціну за товар в залежності від ціни. Отриманий візуальну частину зображено на рис. 4.7.

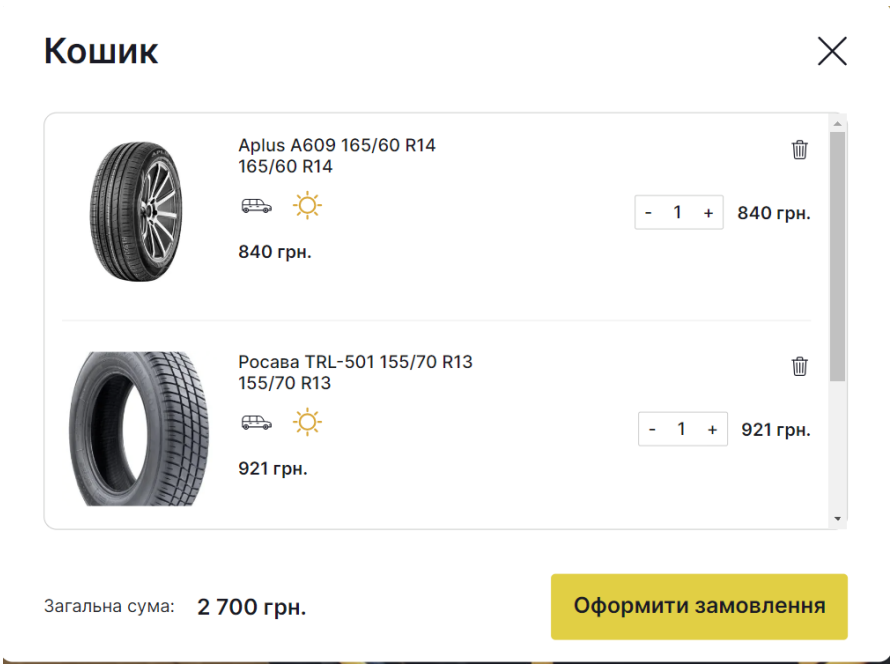


Рисунок 4.7 – Зображженя корзини сайту

Також, маємо змогу видалити товар повністю з корзини. Написаний код для цього модулю зображенно на рис. 4.8- 4.10. Для цього потрібно написати код на JavaScript. Щоб відправити запит на сервер, та відповідь з сервера обробити, та надати клієнту. Та також написати серверну частину. Необхідно додати нові товари до корзини, оновити всі ціни, для вибраного товару, оновити кількість товарів, та оновити загальну ціну, та відправити результат на клієнтську частину.

```
modalCallBack.addEventListener( type: 'click', listener: (event : Event ) : void => {
    if (event.target.classList.contains('modal-callback-container')) {
        modalCallBack.classList.remove( tokens: 'modal-callback-visibility');

        document.body.classList.remove( tokens: 'modal-callback-open');
    }
});
```

```
function updateData(keyProduct, amount, element) : void {
    const xhr : XMLHttpRequest = new XMLHttpRequest();

    xhr.open( method: 'POST', url: '/wp-admin/admin-ajax.php', async: true);

    xhr.setRequestHeader( name: 'Content-Type', value: 'application/x-www-form-urlencoded');

    xhr.onload = function () : void {
        if (xhr.status === 200) {
            element.classList.remove( tokens: 'disable-btn-count');

            const response = JSON.parse(xhr.responseText);

            const newSumProduct : Element = document.querySelector(
                selectors: '.woocommerce-mini-cart__total .woocommerce-Price-amount'
            );

            newSumProduct.innerHTML = response.cart_total;
        } else {
            console.log('Произошла ошибка при выполнении запроса');
        }
    };

    const data : string =
        'action=update_cart_quantity' +
        '&product_key=' +
        keyProduct +
        '&new_quantity=' +
        amount;

    xhr.send(data);
}
```

Рисунок 4.8 – Клієнська обробка написана на чистому JavaScript

```php
2 usages
function update_cart_quantity_callback()
{
    $product_key = sanitize_text_field($_POST['product_key']);
    $new_quantity = intval($_POST['new_quantity']);
    WC()->cart->set_quantity($product_key, $new_quantity);
    $cart_contents = WC()->cart->get_cart();
    $total_amount = 0;
    foreach ($cart_contents as $cart_item) {
        // $count_product = $cart_item['data']->stock_quantity;
        $total_amount += floatval($cart_item['data']->get_price()) * $cart_item['quantity'];
    }

    $response = array(
        'cart_total' => wc_price($total_amount),
    );

    echo json_encode($response);
    die();
}



add_action('wp_ajax_update_cart_quantity', 'update_cart_quantity_callback');

add_action('wp_ajax_nopriv_update_cart_quantity', 'update_cart_quantity_callback');
```

Рисунок 4.9 – Серверна частина написана на PHP

Наступним кроком це написання футеру сайту (рис.4.10).

```php
<footer id="colophon" class="site-footer">
    <div class="container">
        <div class="footer-container">
            <div class="footer-logo">
                <img src="<?php echo get_template_directory_uri(); ?>/assets/img/logo-footer.webp" alt="Лого">
                <a href="<?php echo home_url(); ?>"></a>
            </div>

            <div class="footer-menu-social">
                <div class="footer-submenu-phone">
                    <h3 class="footer-phone-title"><?php echo __('Контактні дані'); ?></h3>
                    <div class="footer-phone-number">
                        <div>
                            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/logo_kievstar.svg" alt="Київстар">
                            <?php echo get_field( selector: 'contact_block_number_1', post_id: 'option')['title']; ?>
                            <a href="<?php echo get_field( selector: 'contact_block_number_1', post_id: 'option')['url'] ?>"></a>
                        </div>
                        <div>
                            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/logo_vodafone.svg" alt="Водафон">
                            <?php echo get_field( selector: 'contact_block_number_2', post_id: 'option')['title']; ?>
                            <a href="<?php echo get_field( selector: 'contact_block_number_2', post_id: 'option')['url'] ?>"></a>
                        </div>
                        <div>
                            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/gmail.svg" alt="Gmail">
                            <?php echo get_field( selector: 'contact_block_email', post_id: 'option')['title']; ?>
                            <a href="<?php echo get_field( selector: 'contact_block_email', post_id: 'option')['url'] ?>"></a>
                        </div>

                    </div>
                </div>
            </div>

            <div class="footer-menu-social footer-menu-social-2">
                <div class="footer-submenu-schedule">
                    <h3 class="footer-schedule-title"><?php echo __('Графік роботи') ?></h3>
                    <div class="footer-schedule-block">
                        <div><span><?php echo get_field( selector: 'contact-block-day', post_id: 'option') ?></span></div>
                        <div><span><?php echo get_field( selector: 'contact-block-time', post_id: 'option') ?></span></div>
                    </div>
                </div>
            </div>
```

```php
<div class="footer-submenu-social">
    <h3 class="footer-social-title"><?php echo __('Соц мережі'); ?></h3>
    <div class="footer-social-block">
        <div>
            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/viber.svg" alt="Viber">
            <a href="<?php echo get_field( selector: 'contact_block_viber', post_id: 'option')['url']; ?>"></a>
        </div>
        <div>
            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/facebook.svg" alt="Facebook">
            <a href="<?php echo get_field( selector: 'contact_block_facebook', post_id: 'option')['url']; ?>"></a>
        </div>
        <div>
            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/instagram.svg" alt="Instagram">
            <a href="<?php echo get_field( selector: 'contact_block_instagram', post_id: 'option')['url']; ?>"></a>
        </div>
        <div>
            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/telegram-plane.svg" alt="Telegram">
            <a href="<?php echo get_field( selector: 'contact_block_telegram', post_id: 'option')['url']; ?>"></a>
        </div>

    </div>
</div>
</div>
<div class="footer-block-copyright">
    <div class="footer-copyright-container">
        <?php echo __('Copyright © ShinaHub, 2024') ?>
    </div>
    <div class="footer-copyright-container">
        <?php echo __('Розробка - '); ?><a href="https://istec.com.ua/" target="_blank"><?php echo __('IST.Group') ?></a>
    </div>
</div>
</div>
</footer>
```

Рисунок 4.10 – Написний код для футеру сайту

Також в футеру сайту було реалізовано вбиваюче вікно для зворотного зв’язку. Воно було перенесене, щоб не заважати завантаження сайту, щоб швидкість була якомога швидше. Для цього було написано HTML розмітка, та логіка була описана за допомогою JS. Цей модуль продемонстрований на рис. 4.13, та код продемонстровано на рис. 4.11 – 4.13.

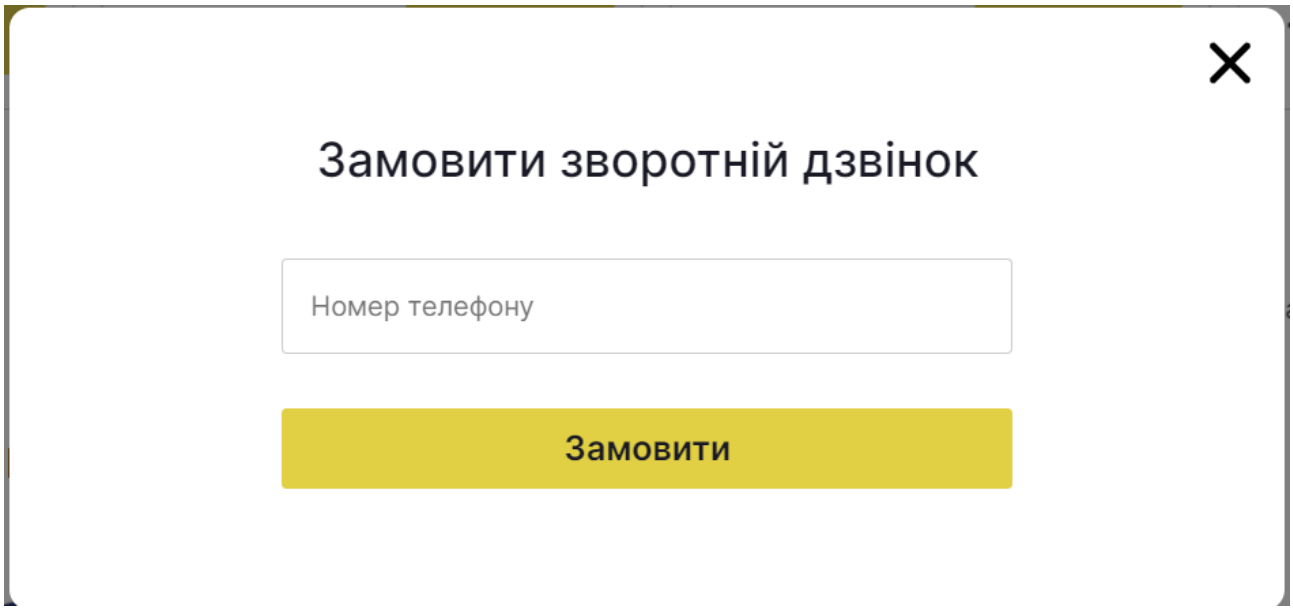


Рисунок 4.11 – Візуальна частина коду

```
<div class="pop_up--wrapper">
    <div class="buy-one-click pop-up--body" data-target="buy" hidden>
        <div class="pop-up--close"></div>
        <?php echo $success_html; ?>
        <?php echo do_shortcode( content: '[contact-form-7 id="011a16d"]' ); ?>
    </div>
    <div class="callback pop-up--body" data-target='callback' hidden>
        <div class="pop-up--close"></div>
        <?php echo $success_html; ?>
        <?php echo do_shortcode( content: '[contact-form-7 id="388143a"]' ); ?>
    </div>
</div>
```

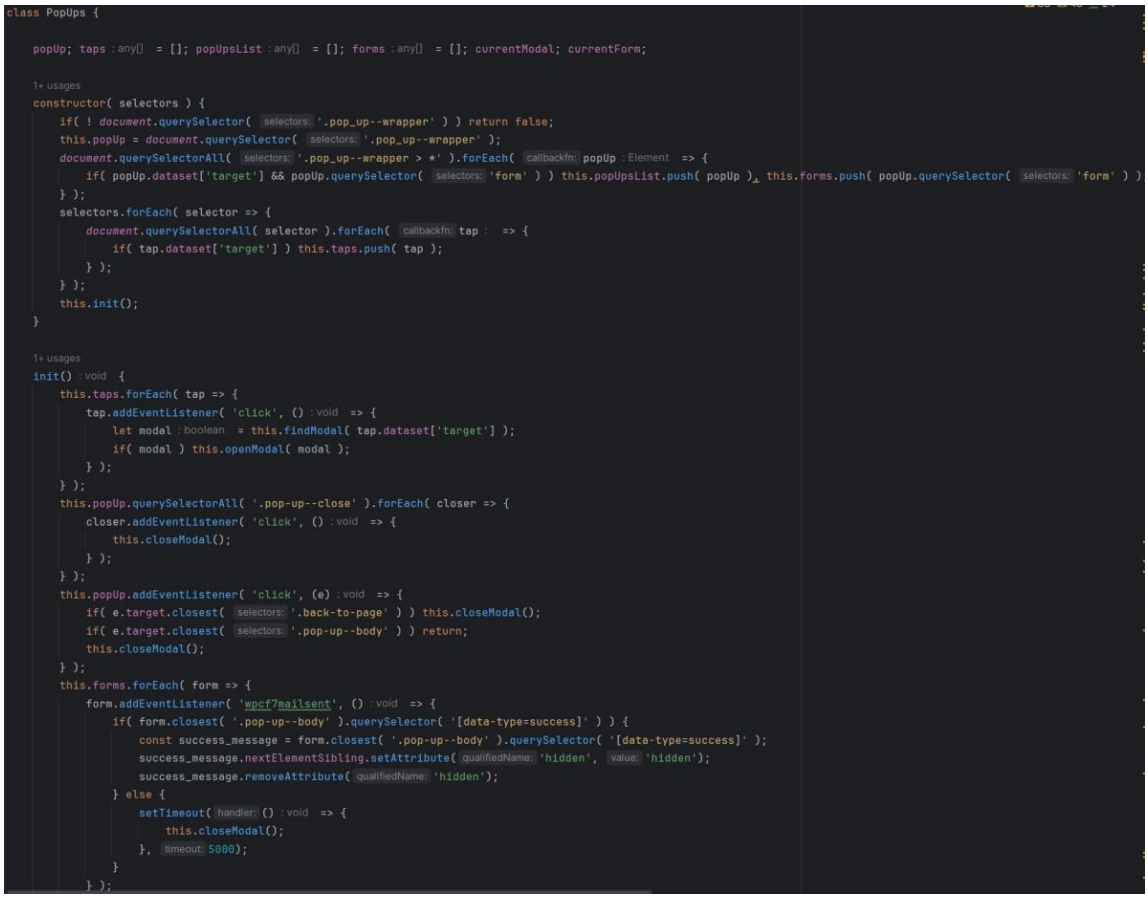
Рисунок 4.12 - HTML розмітка для блоку зворотного зв'язку

```
class PopUps {

    popUp; taps : any[] = []; popUpsList : any[] = []; forms : any[] = []; currentModal; currentForm;

    1+ usages
    constructor( selectors ) {
        if( ! document.querySelector( selectors: '.pop_up--wrapper' ) ) return false;
        this.popUp = document.querySelector( selectors: '.pop_up--wrapper' );
        document.querySelectorAll( selectors: '.pop_up--wrapper > *' ).forEach( callbackfn: popUp : Element => {
            if( popUp.dataset['target'] && popUp.querySelector( selectors: 'form' ) ) this.popUpsList.push( popUp ), this.forms.push( popUp.querySelector( selectors: 'form' ) )
        } );
        selectors.forEach( selector => {
            document.querySelectorAll( selector ).forEach( callbackfn: tap => {
                if( tap.dataset['target'] ) this.taps.push( tap );
            } );
        } );
        this.init();
    }

    1+ usages
    init() : void {
        this.taps.forEach( tap => {
            tap.addEventListener( 'click', () : void => {
                let modal : boolean = this.findModal( tap.dataset['target'] );
                if( modal ) this.openModal( modal );
            } );
        } );
        this.popUp.querySelectorAll( '.pop-up--close' ).forEach( closer => {
            closer.addEventListener( 'click', () : void => {
                this.closeModal();
            } );
        } );
        this.popUp.addEventListener( 'click', (e) : void => {
            if( e.target.closest( selectors: '.back-to-page' ) ) this.closeModal();
            if( e.target.closest( selectors: '.pop-up--body' ) ) return;
            this.closeModal();
        } );
        this.forms.forEach( form => {
            form.addEventListener( 'wpcf7mailsent', () : void => {
                if( form.closest( '.pop-up--body' ).querySelector( '[data-type=success]' ) ) {
                    const success_message = form.closest( '.pop-up--body' ).querySelector( '[data-type=success]' );
                    success_message.nextElementSibling.setAttribute( qualifiedName: 'hidden', value: 'hidden');
                    success_message.removeAttribute( qualifiedName: 'hidden');
                } else {
                    setTimeout( handler: () : void => {
                        this.closeModal();
                    }, timeout: 5000);
                }
            } );
```



```
class PopUps {

    popUp; taps : any[]  = []; popUpsList : any[]  = []; forms : any[]  = []; currentModal; currentForm;

    1+ usages
    constructor( selectors ) {
        if( ! document.querySelector( selectors: '.pop_up--wrapper' ) ) return false;
        this.popUp = document.querySelector( selectors: '.pop_up--wrapper' );
        document.querySelectorAll( selectors: '.pop_up--wrapper > *' ).forEach( callbackFn: popUp : Element  => {
            if( popUp.dataset['target'] && popUp.querySelector( selectors: 'form' ) ) this.popUpsList.push( popUp ), this.forms.push( popUp.querySelector( selectors: 'form' ) )
        } );
        selectors.forEach( selector => {
            document.querySelectorAll( selector ).forEach( callbackFn: tap :  => {
                if( tap.dataset['target'] ) this.taps.push( tap );
            } );
        } );
        this.init();
    }

    1+ usages
    init() : void  {
        this.taps.forEach( tap => {
            tap.addEventListener( 'click', () : void  => {
                let modal : boolean  = this.findModal( tap.dataset['target'] );
                if( modal ) this.openModal( modal );
            } );
        } );
        this.popUp.querySelectorAll( '.pop-up--close' ).forEach( closer => {
            closer.addEventListener( 'click', () : void  => {
                this.closeModal();
            } );
        } );
        this.popUp.addEventListener( 'click', (e) : void  => {
            if( e.target.closest( selectors: '.back-to-page' ) ) this.closeModal();
            if( e.target.closest( selectors: '.pop-up--body' ) ) return;
            this.closeModal();
        } );
        this.forms.forEach( form => {
            form.addEventListener( 'wpcf7mailsent', () : void  => {
                if( form.closest( '.pop-up--body' ).querySelector( '[data-type=success]' ) ) {
                    const success_message = form.closest( '.pop-up--body' ).querySelector( '[data-type=success]' );
                    success_message.nextElementSibling.setAttribute( qualifiedName: 'hidden', value: 'hidden');
                    success_message.removeAttribute( qualifiedName: 'hidden');
                } else {
                    setTimeout( handler: () : void  => {
                        this.closeModal();
                    }, timeout: 5000);
                }
            } );
```

Рисунок 4.13 – Код написаний за допомогою JS

4.2 Розробка головної сторінки сайту

Для розробки першої секції головної сторінки нам необхідно HTML, JS, та PHP. Це все необхідно для фільтрації товарів. В нас задача показувати тільки актуальні атрибути товарів, які були обрані клієнтом сайту. Це необхідно для зручності відображення доступних атрибутів товарів. Та щоб клієнт не зміг обрати таку комбінації атрибутів та потрапити на пусту сторінку каталогу. Для цього нам потрібно:

- після завантаження всього контенту, відправити запит, щоб отримати всі доступні атрибути товарів, для певної категорії товарів;
- потім після обрання певного атрибути шин, чи дисків, нам знову потрібно відправити запит до серверу, щоб отримати всі доступні атрибути для цього категорії товару, та певного атрибути, і таким чином відбувається ципочка всіх настипниї обраних товарів. З кожнем разо передається все більше атрибутів, і кількість доступних товарів такими чином зменшується;

- якщо клієнт змінює категорію товарів, також відправляється новий запит з новою обраною категорією, і таким же чином відбувається пошук товарів, як в попередньому пункті.

Цей блок зображено на рис. 4.14.

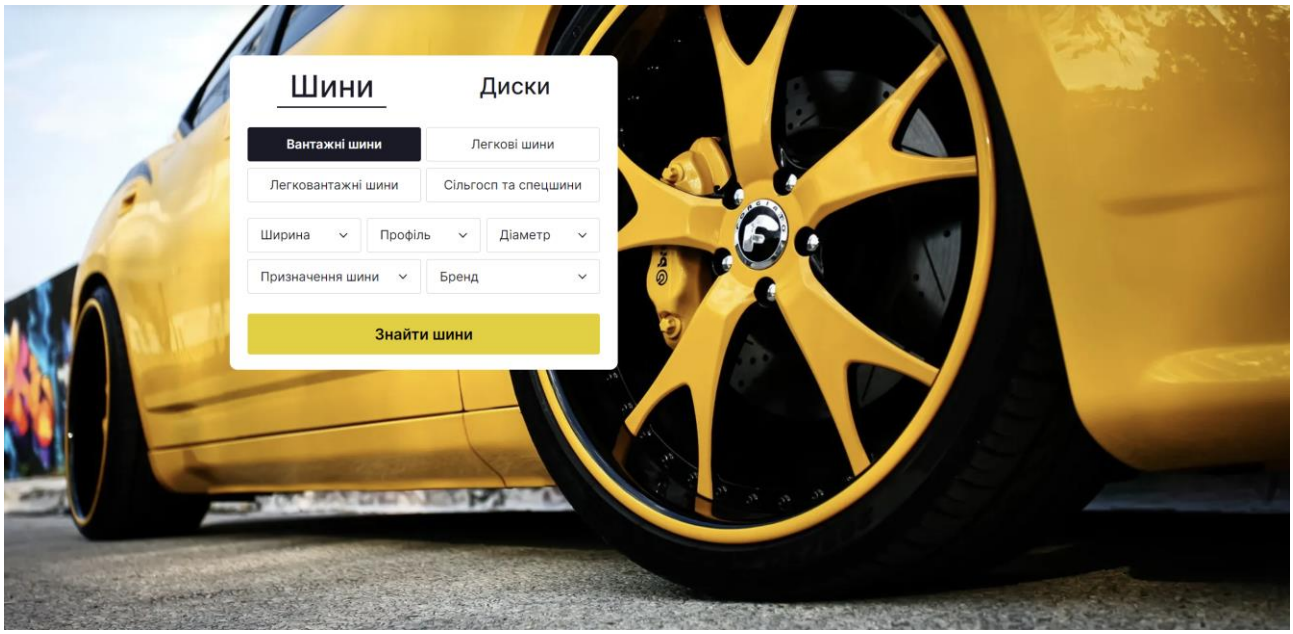


Рисунок 4.14 – Візуальна частина фільтру

На рис. 4.15 – 4.18 зображено необхідний код для реалізації цього функціоналу. На рис. 4.15 – 4.17 зображено код на JS, та на рис. 4.18 відповідно зображено код на PHP.

Були написанні додаткові функції, щоб спростити загальну функцію, та розбити їх на функції, щоб вони відповідали за різні етапи. Першим етапом є знаходження необхідних товарів, наступним етапом це створення JSON для всіх атрибутів для передачі на клієнтську частину сайту.

Наступний блок виводиться за допомогою, стандартної функції WooCommerce. Ця функція виводь рекомендовані товари, які можна обрати в адмінці сайту, де знаходяться всі товари. Візуальну складова зображена на рис. 4.19. Також для виводу було використано бібліотеку слайдеру під назвою Glider. Він був використаний, тому що:

- він має гарну функціональність;
- має невелику вагу для сайту, що забезпечить високу швидкість сайту;
- заощадить велику кількість часу на розробку слайдеру;
- має відкритий код, що додає видалити непотрібний функціонал, що зменшить обсяг.

```
//Фільтрація товарі на головному екрані
const addComments : Element  = document.querySelector( selectors: '.reviews-wait');

setTimeout( handler: () : void  => {
    const selectElements : NodeListOf<Element>  = document.querySelectorAll(
        selectors: '.ist-search-form__field__select'
    ); //Bci select

    let category : null  = null;

    const activeCategory : Element  = document.querySelector( selectors: '.btn-active');

    if (activeCategory.textContent === 'Диски') {
        category = activeCategory.getAttribute( qualifiedName: 'data-value');
    } else {
        const categoryTires : Element  = document.querySelector( selectors: '.search-block-tires');

        const activeCategoryTires : Element  = categoryTires.querySelector( selectors: '.active-category');

        category = activeCategoryTires.getAttribute( qualifiedName: 'data-value');
    }

    addComments.classList.add('reviews-wait-active');

    let selectedOptions : {}  = {};

    const xhr : XMLHttpRequest  = new XMLHttpRequest();

    xhr.open( method: 'POST',  url: '/wp-admin/admin-ajax.php',  async: true);

    xhr.setRequestHeader(
        name: 'Content-Type',

        value: 'application/x-www-form-urlencoded; charset=UTF-8'
    );

    xhr.onreadystatechange = function () : void  {
        if (xhr.readyState === 4 && xhr.status === 200) {
            let response = JSON.parse(xhr.responseText);
```

Рисунок 4.15 – Код логіки написаний на JS частина 1

```
        data = response.form;

        let modifiedData : {}  = {};

        Object.keys(data).forEach(function (key : string ) : void  {
            let newKey : string  = key.replace( searchValue: 'pa_',  replaceValue: '');

            modifiedData[newKey] = data[key];
        }); // убрали pa_ з ключа
        const form : Element  = document.querySelector(
            selectors: '.search-block-filter:not(.search-hidden)'
        );

        const selects : NodeListOf<Element>  = form.querySelectorAll( selectors: '.ist-search-form__field__select');
        selects.forEach( callbackfn: (el : Element ) : void  => {
            let option : NodeListOf<HTMLOptionElement>  = el.querySelectorAll( selectors: 'option');
            let nameSelect = el.name;

            option.forEach( callbackfn: (opt : HTMLOptionElement ) : void  => {
                // if (modifiedData[nameSelect] !== undefined) {...
                if (opt.value) opt.remove();
                if (opt.textContent == '') opt.remove();
            });
            let allOption : any[]  = [];
            if (modifiedData[nameSelect]) {
                Object.entries(modifiedData[nameSelect]).map(([key : string , value]) : void  => {
                    let option : HTMLOptionElement  = document.createElement( tagName: 'option');
                    option.setAttribute( qualifiedName: 'value', key);
                    option.innerHTML = value;
                    allOption.push(option);
                    //if (val == key) option.selected = true;
                });
                allOptionNumber = allOption.filter(
                    (item) => !isNaN(Number(item.textContent))
                );
                allOptionString = allOption.filter((item) : boolean  =>
                    isNaN(Number(item.textContent))
                );
                allOptionNumber = allOptionNumber.sort(
                    compareFn: (a, b) => Number(a.textContent) - Number(b.textContent)
                );
                allOptionString = allOptionString.sort( compareFn: (a, b) =>
                    a.textContent.localeCompare(b.textContent)
                );
                allOption = allOptionNumber.concat(allOptionString);
                allOption.forEach((opt) : void  => {
                    if (opt.textContent !== '') {
                        el.appendChild(opt);
```

```
    xhr.send(
        body: 'action=update_attributes&selected_options=' +
        JSON.stringify(selectedOptions) +
        '&selected_category=' +
        JSON.stringify(category)
    );

    selectElements.forEach( callbackfn: function (selectElement : Element ) : void  {
        selectElement.addEventListener( type: 'change',  listener: function (el : Event ) : void  {
            const activeCategory : Element  = document.querySelector( selectors: '.btn-active');

            if (activeCategory.textContent === 'Диски') {
                category = activeCategory.getAttribute( qualifiedName: 'data-value');
            } else {
                const categoryTires : Element  = document.querySelector( selectors: '.search-block-tires');

                const activeCategoryTires : Element  =
                    categoryTires.querySelector( selectors: '.active-category');

                category = activeCategoryTires.getAttribute( qualifiedName: 'data-value');
            }
            // selectedOptions = {};
            selectedOptions[el.target.name] = el.target.value;

            const selectedElement : EventTarget  = el.target;

            xhr.open( method: 'POST',  url: '/wp-admin/admin-ajax.php',  async: true);

            xhr.setRequestHeader(
                name: 'Content-Type',

                value: 'application/x-www-form-urlencoded; charset=UTF-8'
            );

            xhr.onreadystatechange = function () : void  {
                if (xhr.readyState === 4 && xhr.status === 200) {
                    let response = JSON.parse(xhr.responseText);

                    updateForm(response, selectedElement, selectedOptions);
                }

                addComments.classList.remove( tokens: 'reviews-wait-active');
            };

            addComments.classList.add('reviews-wait-active');
```

Рисунок 4.16 – Код логіки написаний на JS частина 2

```
selects.forEach( callbackfn: (el : Element ) : void  => {
    let option : NodeListOf<HTMLOptionElement>  = el.querySelectorAll( selectors: 'option');
    let val = el.value;
    let nameSelect = el.name;

    if (selectedOption[nameSelect] === '' || nameSelect !== select.name) {
        option.forEach( callbackfn: (opt : HTMLOptionElement ) : void  => {
            // if (modifiedData[nameSelect] !== undefined) {...

            // if (el.value === '') {...
            if (opt.value) opt.remove();
            if (opt.textContent == '') opt.remove();
        });
        let allOption : any[]  = [];
        if (modifiedData[nameSelect]) {
            Object.entries(modifiedData[nameSelect]).map(([key : string , value]) : void  => {
                let option : HTMLOptionElement  = document.createElement( tagName: 'option');
                option.setAttribute( qualifiedName: 'value', key);
                option.innerHTML = value;
                allOption.push(option);
                if (val == key) option.selected = true;
            });
            // allOption = allOption.sort((a, b) =>
            //   !isNaN(Number(a.textContent))
            //     ? Number(a.textContent) - Number(b.textContent)
            //     : a.textContent.localeCompare(b.textContent)
            // );
            allOptionNumber = allOption.filter(
                (item) => !isNaN(Number(item.textContent))
            );
            allOptionString = allOption.filter((item) : boolean  =>
                isNaN(Number(item.textContent))
            );
            allOptionNumber = allOptionNumber.sort(
                 compareFn: (a, b) => Number(a.textContent) - Number(b.textContent)
            );
            allOptionString = allOptionString.sort( compareFn: (a, b) =>
                a.textContent.localeCompare(b.textContent)
            );

            allOption = allOptionNumber.concat(allOptionString);
            allOption.forEach((opt) : void  => {
                if (opt.textContent !== '') {
                    el.appendChild(opt);
                }
            });
        }
    }
```

Рисунок 4.17 – Код логіки написаний на JS частина 3

```php
function update_attributes_callback()
{
    $selected_options = json_decode(stripslashes($_POST['selected_options']), associative: true);
    $selected_category = json_decode(stripslashes($_POST['selected_category']), associative: true);

    $product_filter = get_products_by_attributes($selected_options, $selected_category);

    $all_attributes = get_all_attributes_values($product_filter);
    wp_send_json(['form' =>  $all_attributes]);
    die;
}

add_action('wp_ajax_ update_attributes', 'update_attributes_callback');
add_action('wp_ajax_nopriv_update_attributes', 'update_attributes_callback');
```

Рисунок 4.18 – Код серверу оброблення запросу

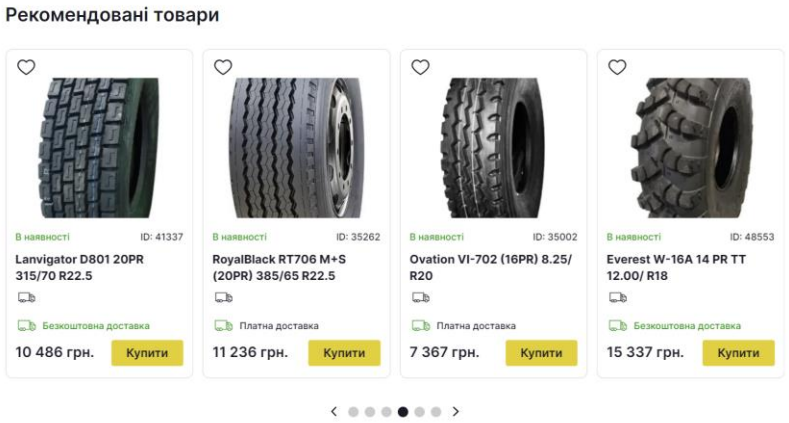


Рисунок 4.19 – Візуальна складова блоку

На рис. 4.20 було зображено код, що реалізовує цей блок. Він написаний за допомогою HTML та PHP.

Наступний блок реалізована за допомогою планігу ACF. Цей блок опишемо більш дктально, щоб зрозуміти, як він створюється, бо більшість таких блоків створюється майже однаково, тільки відрізняється кількостями полями, які ми регіструємо, та типом полів, в наступних блоках не будемо загострювати увагу на такі блокию Він додає змогу нам регіструвати нові поля та виводити їх клієнтам, щоб дати змогу адміну сайту керувати вмістом сайту. На рис. 4.21 зображено візуальну частину блоку. На рис. 4.22 зображено код, який виводить цю інформацію. Також на рис. 4.23 зображено поля, які регіструється в адмінці. Та на рис. 4.24 зображення, як наповнюються ці поля контентом.

```php
<section class="section-recommendation">
    <h2 class="recommendation-title">Рекомендовані товари</h2>
    <div class="recommendation-product">
        <div id="recommendation" class="glide">
            <div class="recommendation-block">
                <div class="glide__track" data-glide-el="track">
                    <ul class="glide__slides">
                        <?php
                        $products = wc_get_products( array(
                            'status'     => 'publish',
                            'tag'        => 'rekomendovanyj-tovar',
                            )
                        );
                        if ( $products ) {
                            foreach ( $products as $product_id ) {
                                global $product;
                                $product = wc_get_product($product_id->id);
                                if( $product_id != null && $product->is_visible() ) { ?>
                                    <li class="glide__slide" >
                                        <?php
                                            $product = wc_get_product($product_id->id);
                                            get_template_part( slug: 'template-parts/content',  name: 'historyProduct');
                                        ?>
                                    </li><?php
                                }
                            }
                        }
                        ?>
                    </ul>
                    <div class="container-btn-bullets">
                <div data-glide-el="controls" class="recommendation-arrows">
                    <button data-glide-dir="<" class="recommendation-arrow viewed-arrow-left">
                        <img src="<?php echo get_template_directory_uri(); ?>/assets/img/arrow-left.svg" alt="arrow-left" />
                    </button>

                    <div id="recommendation-bullets" class="glide__bullets" data-glide-el="controls[nav]">
                    </div>
                    <button data-glide-dir=">" class="recommendation-arrow recommendation-arrow-right">
                        <img src="<?php echo get_template_directory_uri(); ?>/assets/img/arrow-right.svg" alt="arrow-right" />
                    </button>
                </div>
            </div>
                </div>
            </div>
        </div>
    </div>
</section>
```

Рисунок 4.20 – Реалізація блоку

Рисунок 4.21 – Зображенно візуальну складову блоку

```php
<?php if (get_field( selector: 'brendy',  post_id: 'option')) { ?>
    <section class="section-block-brands">
        <h2 class="brands-title">
            <?php echo __('Бренди з якими співпрацюємо'); ?>
        </h2>
        <div class="brands-block-glider">
            <div id="brands" class="glide">


                <div class="glide__track" data-glide-el="track">
                    <ul class="glide__slides">
                        <?php
                        $brandsGlider = get_field( selector: 'brendy',  post_id: 'option');
                        foreach ($brandsGlider as $value) {
                        ?>
                            <li class="glide__slide">
                                <?php echo wp_get_attachment_image($value['kartynka_brendu']['id'],  size: '312*100'); ?>
                            </li>
                        <?php
                        }
                        ?>
                    </ul>
                </div>


                <div class="container-btn-bullets">
                    <div data-glide-el="controls" class="recommendation-arrows">
                        <button data-glide-dir="<" class="recommendation-arrow brands-arrow-left">
                            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/arrow-left.svg" alt="arrow-left" loading="lazy" />
                        </button>
                        <div class="glide__bullets" data-glide-el="controls[nav]">
                            <?php
                            foreach ($brandsGlider as $index => $value) {
                            ?>
                                <button class="glide__bullet" data-glide-dir="=<?php echo $index ?>"></button>
                            <?php } ?>
                        </div>
                        <button data-glide-dir=">" class="recommendation-arrow brands-arrow-right">
                            <img src="<?php echo get_template_directory_uri(); ?>/assets/img/arrow-right.svg" alt="arrow-right" loading="lazy" />
                        </button>
                    </div>
                </div>
            </div>
        </div>
    </section>
<?php } ?>
```

Рисунок 4.22 – Код який виводить цей блок

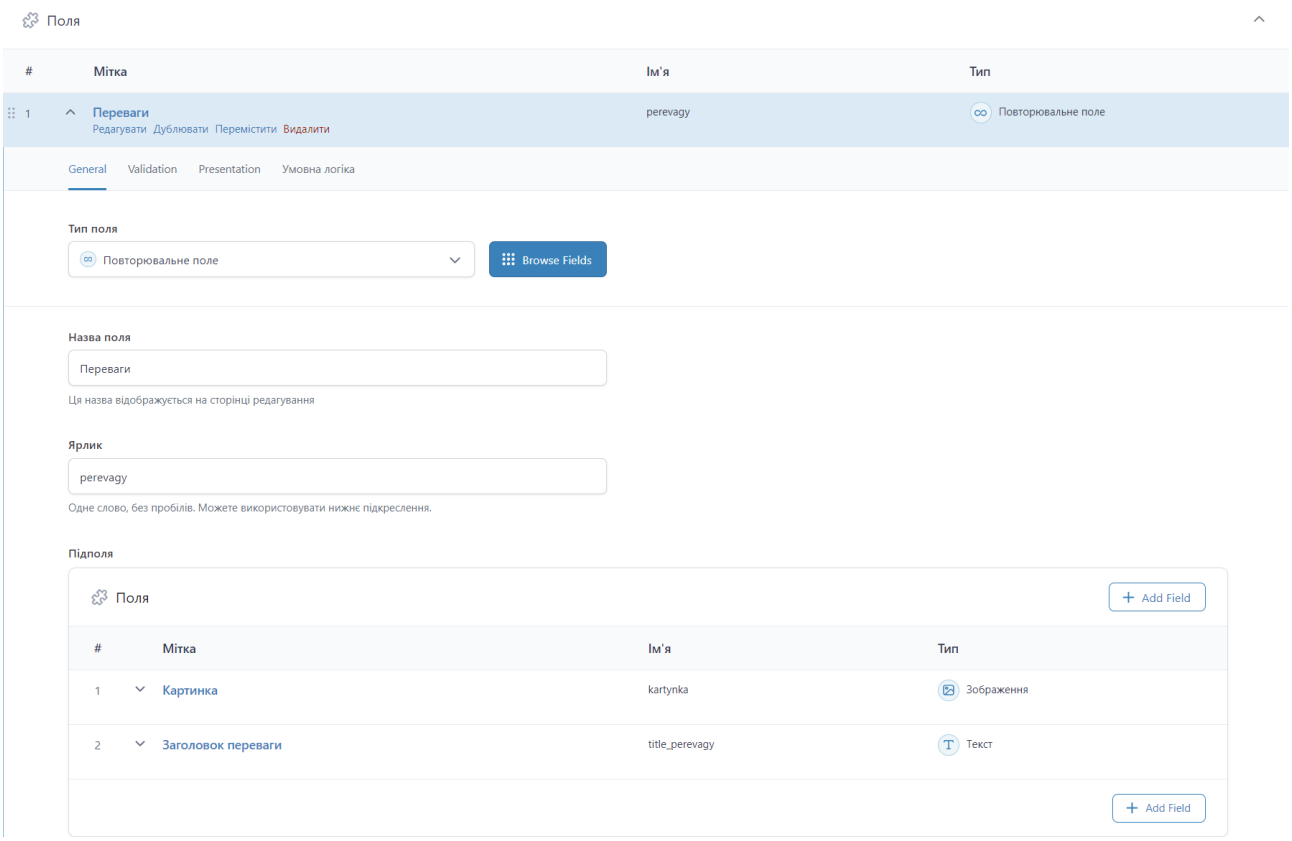


Рисунок 4.23 – Додавання нових полів до ACF

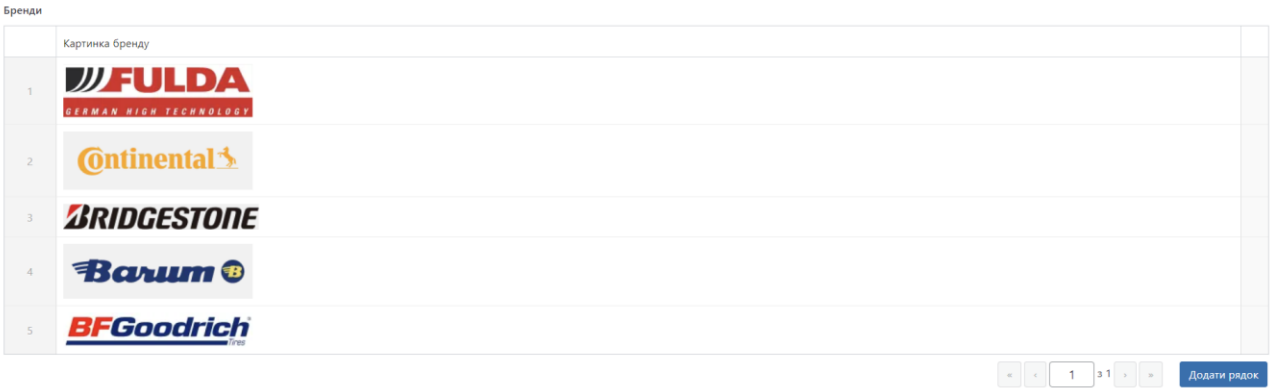


Рисунок 4.24 – Наповнення контентом полів в адмінці сайту

Наступний блок реалізовано за допомогою плагіну Contact from 7. Це безплатний плагін, який допомагає в створенні форм, налаштуванні їх, та допомагає в обробці цих запитів. Цей плагін відсилає на пошту інформацію з форми, а також дублює цю інформацію в адмінку сайту. Візуальну частину зображено на рис. 4.25. Та на рис. 4.26 зображенно візуальну складову плагіну, який допомагає виводити цю форму.

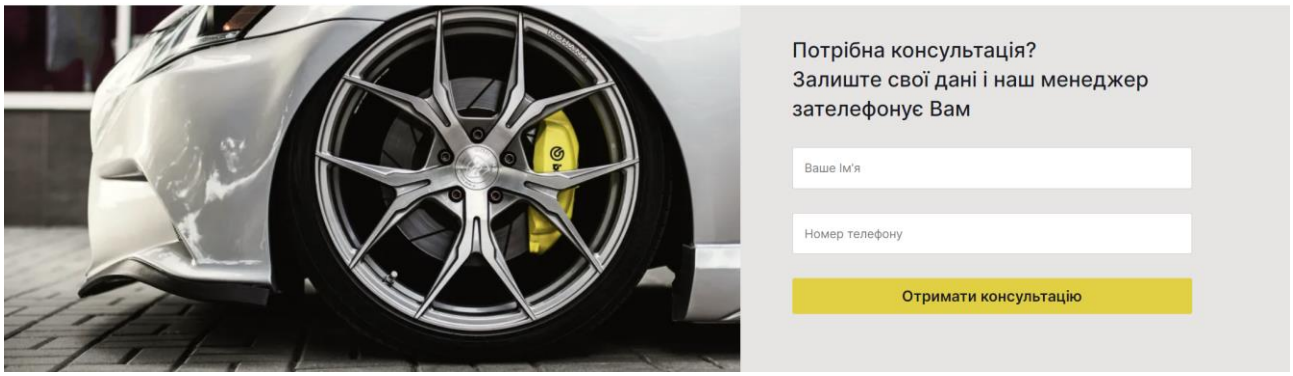


Рисунок 4.25 – Зображено візуальну частину блоку

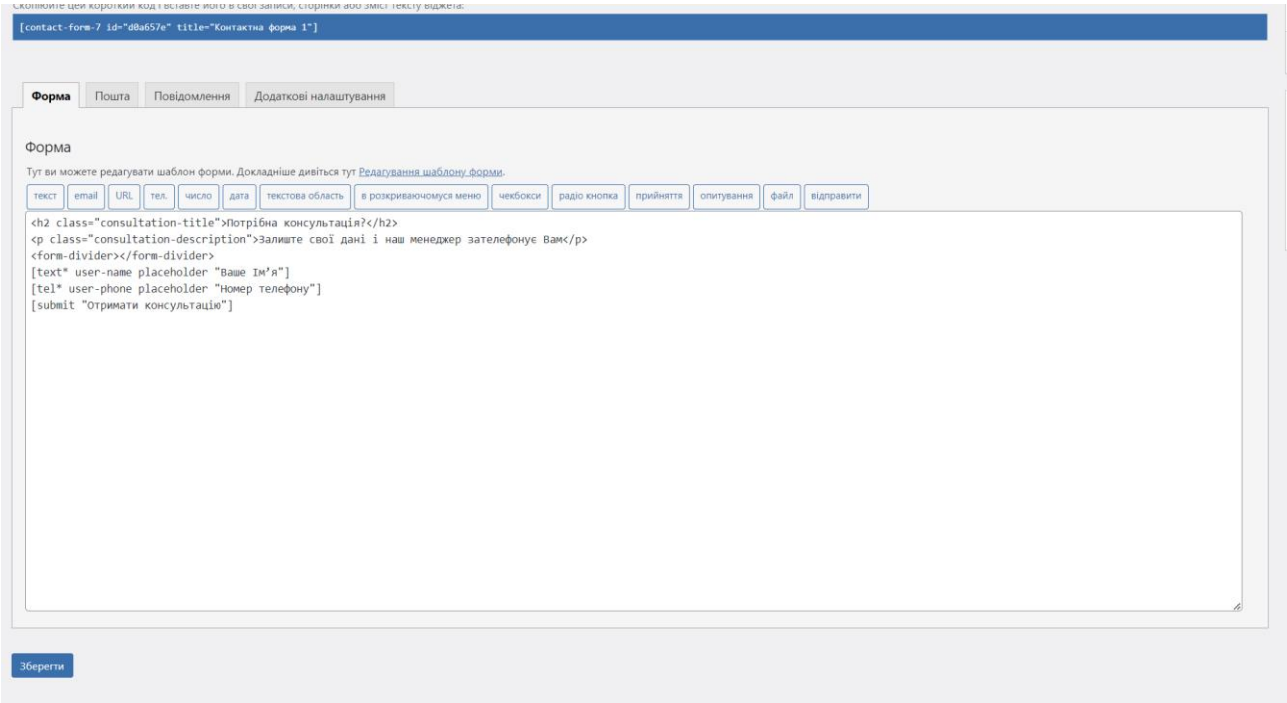


Рисунок 4.26 – Відображення форми, яку можна редагувати в адмінці сайту

Ми можемо додавати нові поля до цієї форми, та необхідний HTML код, для правильного візуально відображення. Також можна було використати плагін Gravity Form. Він кращій з візуальної частини, він легший для освоєння людям, які не мають навичок в HTML, але він являється платним, та не підійшов нашим замовгикам.

Наступний блок являє собою ACF поля, які заповнюються адміном сайту. Код подібний до попереднього блок, тож представляємо візуальну частину блоку, який зображено на рис. 4.27.

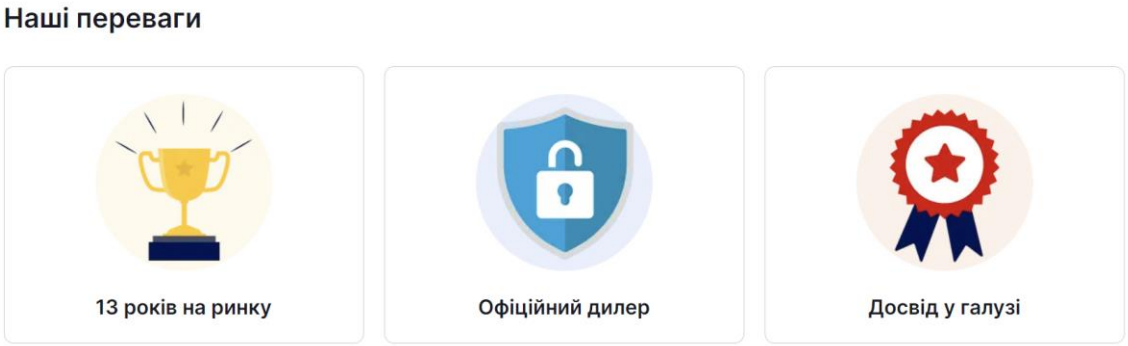


Рисунок 4.27 – Зображення блоку 4.26

Наступний блок – вивід коментарів. Він виводить останні 5 коментарів , які адмін підтвердив. Також для кращого візуального оформлення було знову

використано слайдер. На рис. 4.28 зображено візуальну складову блоку, та на рис. 4.29 відображено код, що виводить ці коментарі.

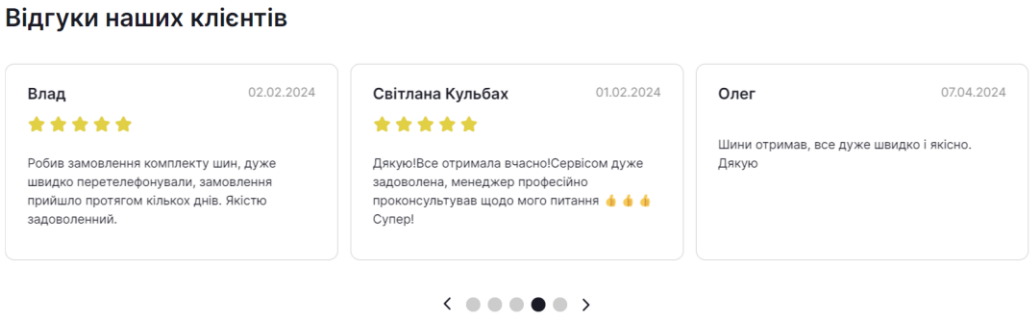
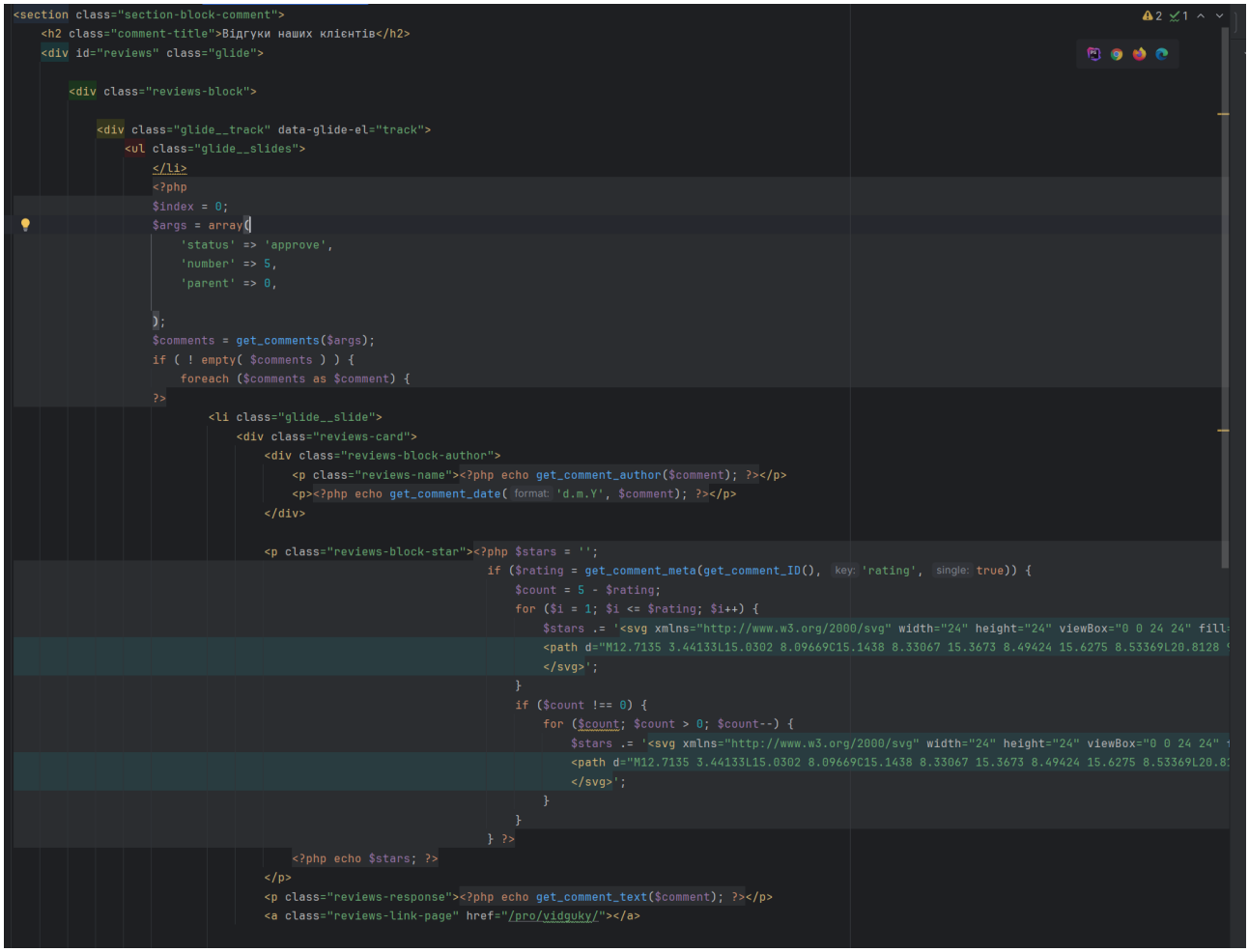


Рисунок 4.28 – Зображено візуальну частину блоку



```
                    </li>
                <?php
                        $index++;
                    }
                } else {
                    echo 'Пока нет комментариев.';
                }
                ?>
            </ul>
        </div>

    </div>
    <div class="container-btn-bullets">
        <div data-glide-el="controls" class="recommendation-arrows">
            <button data-glide-dir="<" class="recommendation-arrow viewed-arrow-left">
                <img src="<?php echo get_template_directory_uri(); ?>/assets/img/arrow-left.svg" alt="arrow-left" />
            </button>
            <div class="glide__bullets" data-glide-el="controls[nav]">
                <?php
                for ($i = 0; $i < $index; $i++) { ?>
                    <button class="glide__bullet" data-glide-dir="=<?php echo $i; ?>"></button>
                <?php } ?>
            </div>
            <button data-glide-dir=">" class="recommendation-arrow reviews-arrow-right">
                <img src="<?php echo get_template_directory_uri(); ?>/assets/img/arrow-right.svg" alt="arrow-right" />
            </button>
        </div>
    </div>
</div>
</section>
```

Рисунок 4.29 – Зображено код для виводу блоку коментарів

Наступним блоком є відображення контактної інформації, вона старенна також на основі плагіну ACF, щоб клієнт мав змогу швидко змінювати телефон, посилання на соціальні мережі, та інше. Цей блок зображено на рис. 4.31.

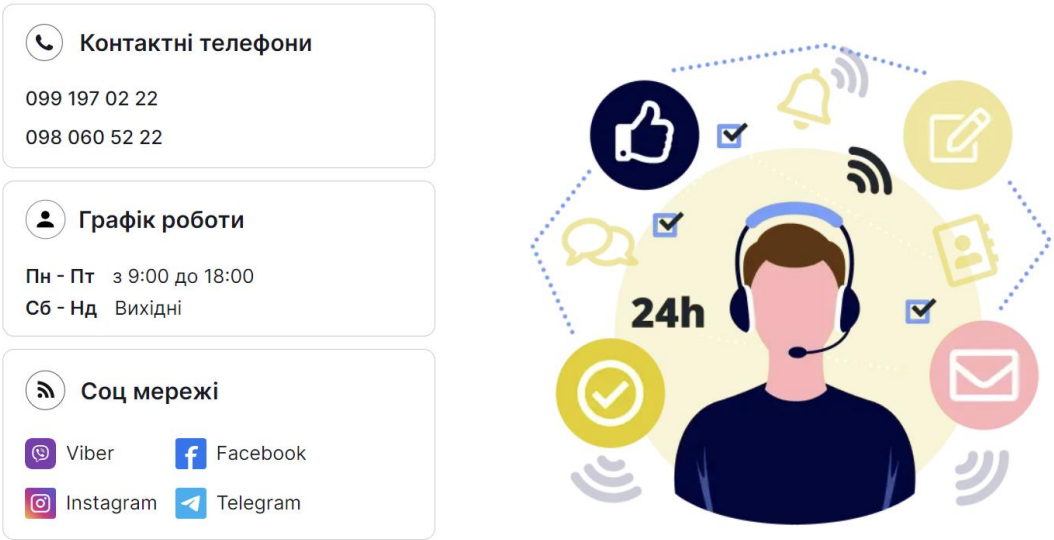


Рисунок 4.30 – Вивід контактної інформації

4.3 Розробка сторінок товарів

Сторінки для товарів будуть використовувати один шаблон, тож нам необхідно створити, тільки один шаблон для цього. На рис. 4.30 зображено створений шаблон для сторінок.

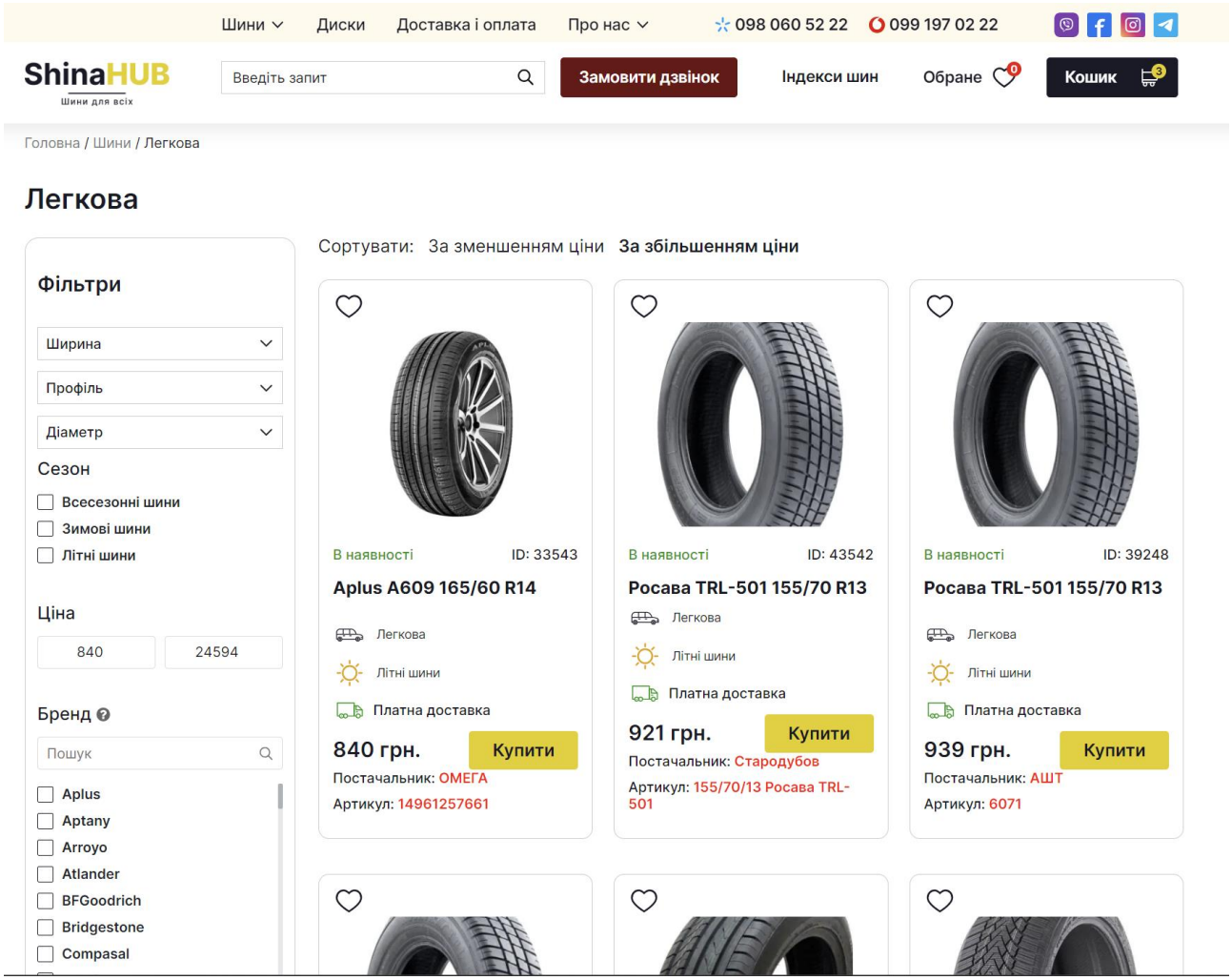
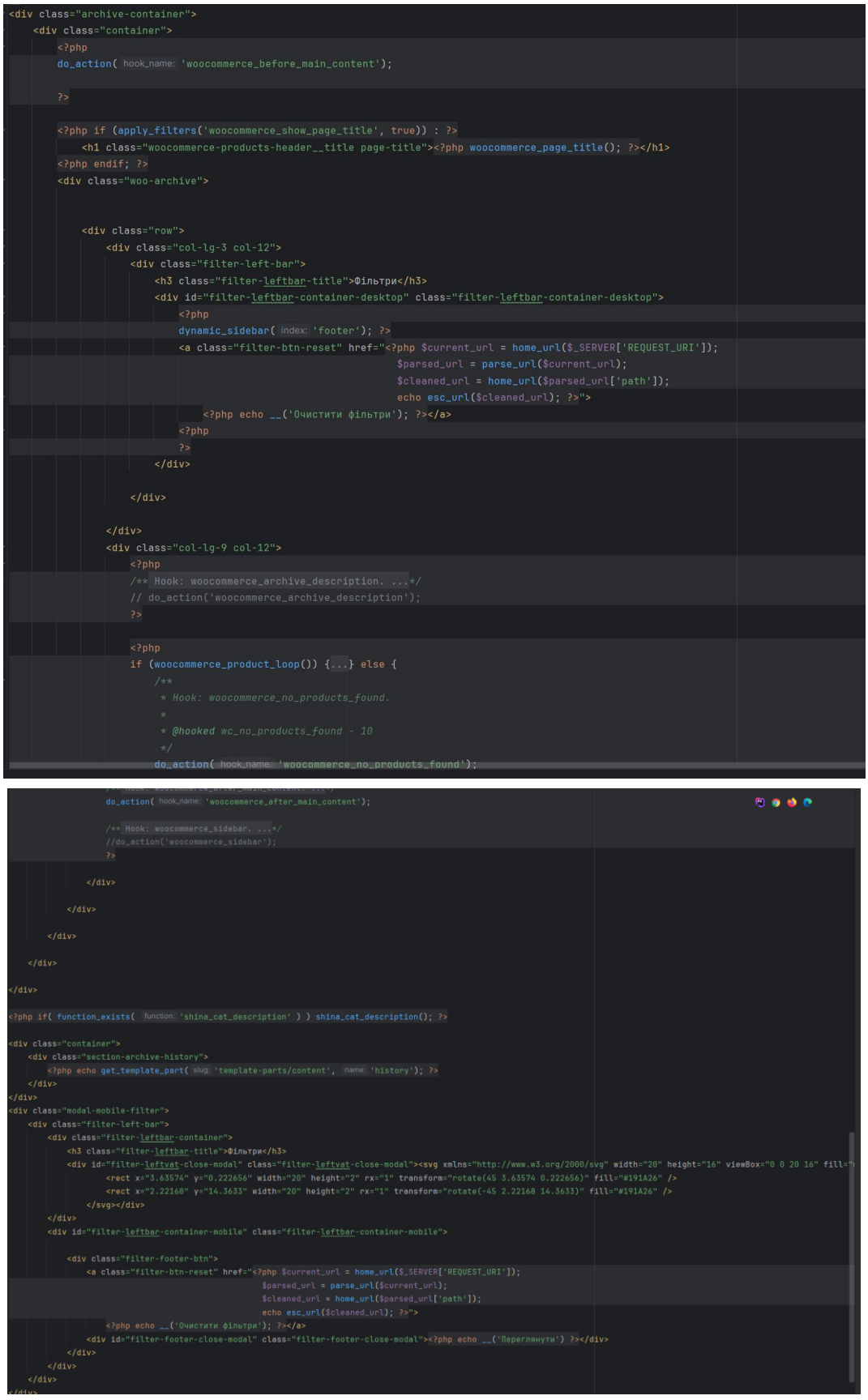


Рисунок 4.30 – Шаблон товарів

Також нам було необхідно додати фільтр, щоб не розробляти свій кастомний фільтр, було знайдено плагін, який нам заощадить часи розробки цього фунціоналу, та він нам повністю підходить по всім параметрам. Він має змогу завантажувати товар за допомогою AJAX, та має весь необхідний функціонал. На рис. 4.31 зображено розроблений шаблон для виведення товарів, він розроблений на базі стандартного шаблону WooCommerce.



```
<div class="archive-container">
    <div class="container">
        <?php
        do_action( hook_name: 'woocommerce_before_main_content');

        ?>

        <?php if (apply_filters('woocommerce_show_page_title', true)) : ?>
            <h1 class="woocommerce-products-header__title page-title"><?php woocommerce_page_title(); ?></h1>
        <?php endif; ?>
        <div class="woo-archive">


            <div class="row">
                <div class="col-lg-3 col-12">
                    <div class="filter-left-bar">
                        <h3 class="filter-leftbar-title">Фільтри</h3>
                        <div id="filter-leftbar-container-desktop" class="filter-leftbar-container-desktop">
                            <?php
                            dynamic_sidebar( index: 'footer'); ?>
                            <a class="filter-btn-reset" href="<?php $current_url = home_url($_SERVER['REQUEST_URI']);
                                                                $parsed_url = parse_url($current_url);
                                                                $cleaned_url = home_url($parsed_url['path']);
                                                                echo esc_url($cleaned_url); ?>">
                                <?php echo __('Очистити фільтри'); ?></a>
                            <?php
                            ?>
                        </div>

                    </div>

                </div>
                <div class="col-lg-9 col-12">
                    <?php
                    /** Hook: woocommerce_archive_description. ...*/
                    // do_action('woocommerce_archive_description');
                    ?>

                    <?php
                    if (woocommerce_product_loop()) {...} else {
                        /**
                         * Hook: woocommerce_no_products_found.
                         *
                         * @hooked wc_no_products_found - 10
                         */
                        do_action( hook_name: 'woocommerce_no_products_found');
```



```
                    do_action( hook_name: 'woocommerce_after_main_content');

                    /** Hook: woocommerce_sidebar. ...*/
                    //do_action('woocommerce_sidebar');
                    ?>

                </div>

            </div>

        </div>

    </div>

</div>

<?php if( function_exists( function: 'shina_cat_description' ) ) shina_cat_description(); ?>

<div class="container">
    <div class="section-archive-history">
        <?php echo get_template_part( slug: 'template-parts/content', name: 'history'); ?>
    </div>
</div>
<div class="modal-mobile-filter">
    <div class="filter-left-bar">
        <div class="filter-leftbar-container">
            <h3 class="filter-leftbar-title">Фільтри</h3>
            <div id="filter-leftvat-close-modal" class="filter-leftvat-close-modal"><svg xmlns="http://www.w3.org/2000/svg" width="20" height="16" viewBox="0 0 20 16" fill="
                <rect x="3.63574" y="0.222656" width="20" height="2" rx="1" transform="rotate(45 3.63574 0.222656)" fill="#191A26" />
                <rect x="2.22168" y="14.3633" width="20" height="2" rx="1" transform="rotate(-45 2.22168 14.3633)" fill="#191A26" />
            </svg></div>
        </div>
        <div id="filter-leftbar-container-mobile" class="filter-leftbar-container-mobile">

            <div class="filter-footer-btn">
                <a class="filter-btn-reset" href="<?php $current_url = home_url($_SERVER['REQUEST_URI']);
                                                    $parsed_url = parse_url($current_url);
                                                    $cleaned_url = home_url($parsed_url['path']);
                                                    echo esc_url($cleaned_url); ?>">
                    <?php echo __('Очистити фільтри'); ?></a>
                <div id="filter-footer-close-modal" class="filter-footer-close-modal"><?php echo __('Переглянути') ?></div>
            </div>
        </div>
    </div>
</div>
```

Рисунок 4.31 – Розроблений шаблон для виводу товарів

4.4 Розробка сторінки замовлення товарів

WooCommerce надає цю стандартну сторінку для оформлення товарів, але для замовника вона не повністю підходила, по дизайну. Тож, було переписано цей шаблон, відповідно до нашого дизайну. Переписавши цей шаблон отримали таку візуальну складову, зображену на рис. 4.32 та на рис. 4.33 зображено код цього шаблону.

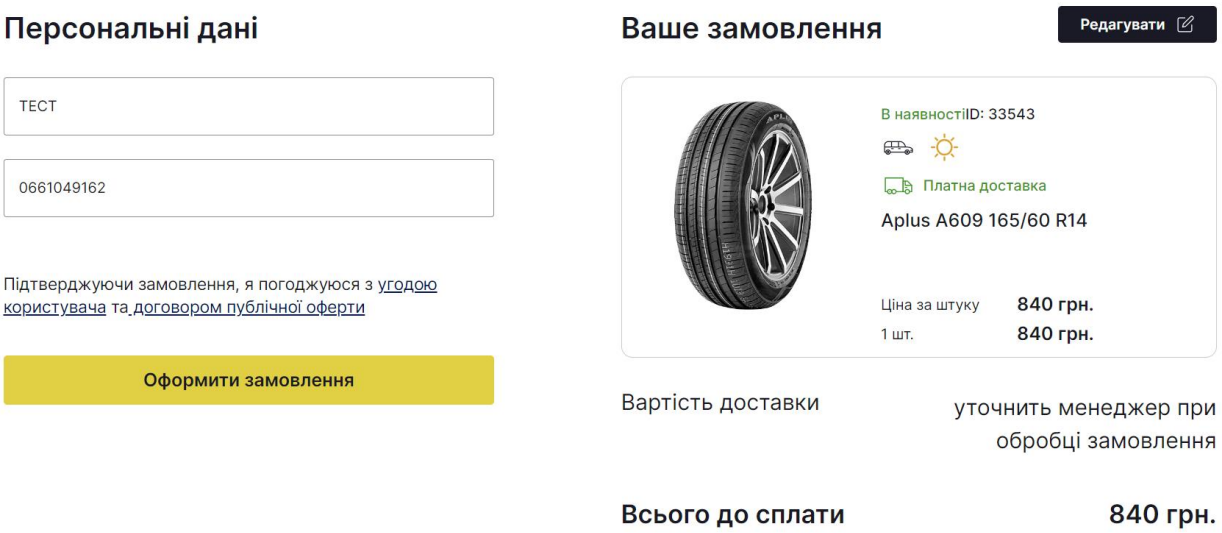


Рисунок 4.32 – Шаблон замовлення товару

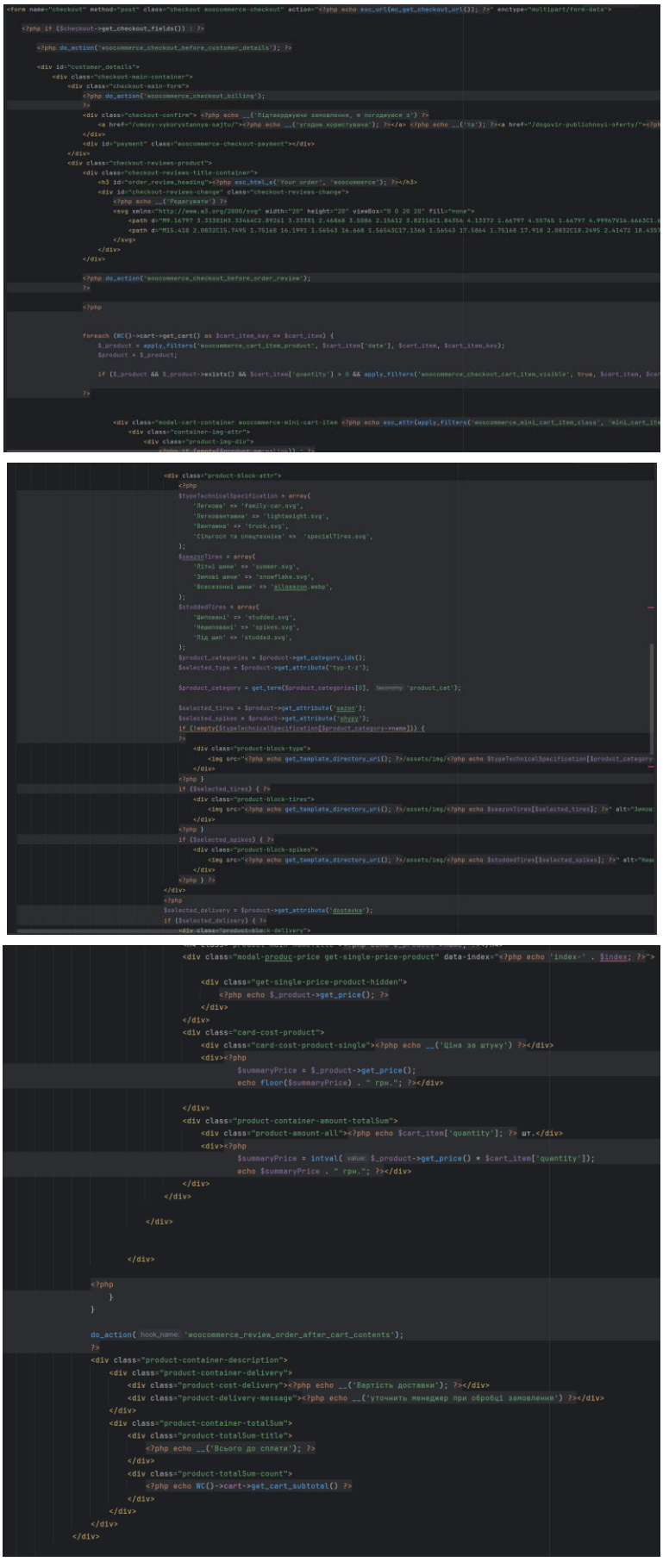


Рисунок 4.33 – Шаблон оформлення товару

Також після оформлення товару, нам потрібно змінити шаблон виведення товару, та коментаря до цього замовлення. Зображення візуальної частини коду представлено на рис. 4.34 та код для шаблону на рис.4.35.

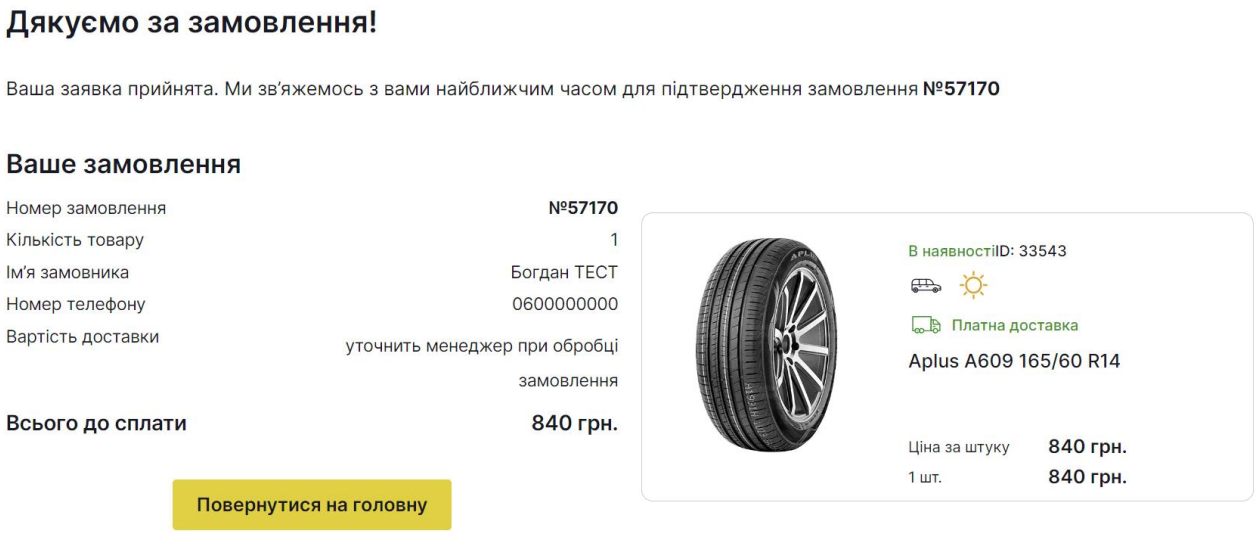


Рисунок 4.34 – Візуальна складова оформленого замовлення

```php
<div class="order-all-details-container">
    <div class="order-all-details">
        <h2 class="order-details-title"><?php echo __('Ваше замовлення') ?></h2>
        <div class="order-details-numberOrder">
            <span><?php echo __('Номер замовлення'); ?></span>
            <strong>№<?php echo $order->get_order_number(); ?></strong>
        </div>
        <div class="order-details-amount-product">
            <span><?php echo __('Кількість товару'); ?></span>
            <span><?php echo $order->get_item_count(); ?></span>
        </div>
        <div class="order-details-name">
            <span><?php echo __('Ім'я замовника') ?></span>
            <span><?php echo $order->get_billing_first_name(); ?></span>
        </div>
        <div class="order-details-phone">
            <span><?php echo __('Номер телефону'); ?></span>
            <span><?php echo $order->get_billing_phone(); ?></span>
        </div>
        <div class="order-details-delivery">
            <span><?php echo __('Вартість доставки'); ?></span>
            <span class="order-details-message"><?php echo __('уточнить менеджер при обробці замовлення'); ?></span>
        </div>
        <div class="order-details-totalSum">
            <span><?php echo __('Всього до сплати'); ?></span>
            <span><?php echo $order->get_total(); ?> <?php echo __('грн.') ?></span>
        </div>

        <a class="back-to-home-button" href="<?php echo get_home_url(); ?>">
            <?php echo __('Повернутися на головну'); ?>
        </a>
    </div>
    <div class="order-product">
        <?php $items = $order->get_items();
        foreach ($items as $item_id => $item) {
            $_product = $item->get_product();
            $product = $_product;
        ?>
```

```php
<div class="checkout-reviews-product">
    <div class="modal-cart-container woocommerce-mini-cart-item <?php echo esc_attr(apply_filters('woocommerce_mini_cart_item_class', 'mini_cart_item',
        <div class="container-img-attr">
            <div class="product-img-div">
                <?php if (empty($product_permalink)) : ?>
                    <?php $image = wp_get_attachment_url($_product->image_id);
                    if ($image) {
                    ?>
                        <img src="<?php echo $image; ?>" alt="<?php echo __('Картинка шини'); ?>">
                    <?php } else { ?>
                        <img src="<?php echo get_template_directory_uri(); ?>/assets/img/no_img.webp" alt="<?php echo __('Картинка відсутня'); ?>">
                    <?php } ?>
                <?php else : ?>
                    <a href="<?php echo esc_url($product_permalink); ?>">
                        <?php echo $thumbnail; // phpcs:ignore WordPress.Security.EscapeOutput.OutputNotEscaped
                        ?>
                    </a>
                <?php endif; ?>
            </div>
            <div class="product-main-content">
                <div class="product-container-instock-article">
                    <?php if ($_product->is_in_stock()) { ?>
                        <div class="product-instock"><?php echo 'В наявності'; ?> <span>ID: <?php echo $product->id; ?></span></div>
                    <?php
                    } else { ?>
                        <div class="product-instock"><?php echo 'Товар тимчасово відсутній'; ?></div>
                    <?php
                    }
                    ?>
                    <div class="product-main-article">Артикул <?php echo $_product->get_sku(); ?>
                    </div>
                </div>
```

Рисунок 4.35 – Шаблон для виведення оформленого замовлення

4.5 Розробка плагіну для завантаження товарів

Потрібно розробити плагін, який буде додавати нові товари до сайту. Список товарів сформований в Exel, тобто нам потрібно розробити парсер, який буде розбивати цей файл на товари, та додавати товари нам до бази даних. Також, цей плагін повинен не тільки додавати нові товари, а й оновлювати ціни уже в існуючих.

При розробці цього плагіну було деякі вагомі проблеми. Перша проблема це розбитя файлу на товари. JavaScript не можна розбити файл з розиренням .xls та .xlsx. Це можна зробити за допомогою бібліотек для JavaScript. Але змоги додати бібліотеку в нас не було. Тож було вирішено, конвертувати ці формати в .csv, його можна з легкістю розбити на товари та колонки за допомогою стандартного функціоналу JavaScript.

Наступна проблема це кількість товарів. В тестових режимах при кількості товарах до 500 штук, було все добре, але при збільшені кількості товарів до 2000 штук, серверна частина не витримувала, і сайт падав, навіть на локальному серверу. На хостингу кількість товарів була ще менша, тому що виділених ресурсів було в рази менше. Необхідно було щоб плагін міг завантажити більше 10000 товарів, щоб не ділити файли з товарами. Тому було реалізовано ділення по 100 товарів за допомогою JavaScript, та відправки на сервер. В тестовому режимі було вибрано 100 товарів, є самим оптимальним, так як сервер оброблює таку кількість товарів дуже швидко, приблизно до 10 секунд. Та ще було встановлено затримку на відправку нових товарів, після відповіді від сервера, що товари успішно додані до бази даних. Затримка в експериментальному режимі було обрано в 5 секунд, щоб навантаження на сервері падало майже до 0.

Таке велике навантаження на сервер через те, що для завантаження товару, нам необхідно кожен товар перевірити в наявності базі даних по ID товару. А це кожен раз йде запит на наявність такого товару в базі даних, якщо його не має, то ми його створюємо та наповнюємо його всіма необхідними атрибутами, які надані у файлі. Якщо він уже створений, то нам необхідно тільки оновити ціну для нього.

На рисунку 4.36 представлено візуальну складову плагіну, та його відповіді про завантаження товару.

Рисунок 4.36 – Візуальна складова плагіну

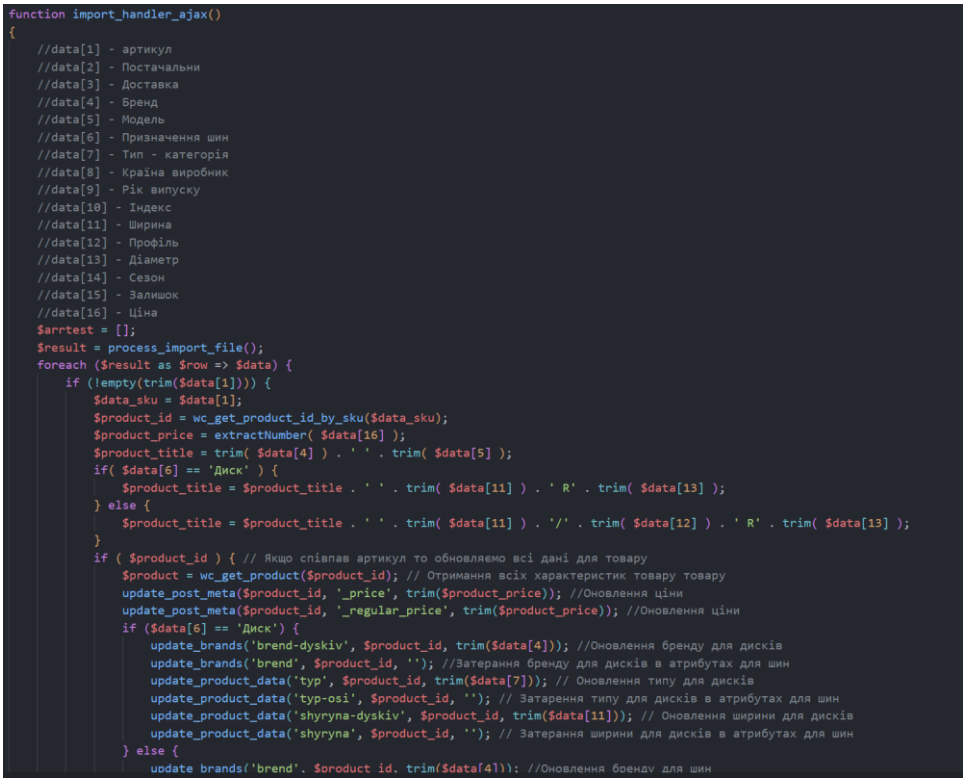
В тестовому режимі було завантажено 400 товарів. Як бачимо ми маєм відповідь від серверу, про успішність завантаженні товарів. Якщо виникли проблеми з завантаженням товарів, нам це напише плагін, але щоб дізнатися, що саме за помилка завадила завантаженню товарів, нам необхідно перевірити журнал помилок серверу, він може знаходися на сервері в візуальному вигляді, або .txt форматі, де будуть записані останні помилки. На рис. 4.37 продемонстровано, як повинен виглядати файл, який буде завантажуватися.

№з/п	Артикул	Постачальник	Доставка	Бренд	Модель	Тип	Призначення шини	Країна виробник	Рік випуску	Індекс	Ширина	Профіль	Діаметр	Сезон	Залишок	Ціна
1	1000992		Безкоштовна	Bridgestone	M729	Вантажна	Ведуча				235	75	17.5		4	10867.0
2	1001429		Безкоштовна	Bridgestone	M729	Вантажна	Ведуча				285	70	19.5		2	13147.0
3	1001477		Безкоштовна	Bridgestone	M729	Вантажна	Ведуча				295	80	22.5		>12	17542.0
4	1001487		Безкоштовна	Bridgestone	M788	Вантажна	Універсальна				295	80	22.5		>12	20422.0
5	1001614		Безкоштовна	Bridgestone	M729	Вантажна	Ведуча				315	70	22.5		>12	20244.0
6	1001661		Безкоштовна	Bridgestone	M788	Вантажна	Універсальна				315	80	22.5		>12	17255.0
7	1008810		Безкоштовна	Bridgestone	M716	Вантажна	Ведуча				8.5		17.5		2	8952.0
8	1008841		Безкоштовна	Bridgestone	L317	Вантажна	Ведуча				13		22.5		>12	20841.0
9	1008842		Безкоштовна	Bridgestone	L355	Вантажна	Ведуча				13		22.5		>12	25316.0
10	1008844		Безкоштовна	Bridgestone	M840	Вантажна	Універсальна				13		22.5		>12	18355.0
11	1009197		Безкоштовна	Bridgestone	M748	Вантажна	Причіпна				385	65	22.5		>12	22782.0
12	1015872		Безкоштовна	Double Coin	RLB450	Вантажна	Ведуча				315	70	22.5		>12	10560.0
13	1046152		Безкоштовна	GTRadial	GTF78+	Вантажна	Причіпна				385	65	22.5		>12	13732.0
14	1070185		Безкоштовна	Bridgestone	R168	Вантажна	Причіпна				385	65	22.5		>12	19651.0
15	1121591		Безкоштовна	Bridgestone	R249	Вантажна	Рульова				315	70	22.5		>12	20051.0
16	1121593		Безкоштовна	Bridgestone	R249	Вантажна	Рульова				315	60	22.5		>12	21483.0
17	1121600		Безкоштовна	Double Coin	RR905	Вантажна	Причіпна				385	55	19.5		>12	9224.0
18	1121612		Безкоштовна	Double Coin	RLB450	Вантажна	Ведуча				295	60	22.5		>12	9191.0
19	1121627		Безкоштовна	Bridgestone	R249	Вантажна	Рульова				315	80	22.5		>12	20745.0

Рисунок 4.37 – Вигляд файлу для завантаження

Комірки для всіх атрибутів повинні бути не змінні, щоб плагін записував в відповідні атрибути значення. Бо в іншому випадку буде не відповідність до назви атрибуту та значення.

На рис. 4.38 представлена частина коду написана на PHP, що додає нові товари до бази даних, або оновлює ціну товару.



```php
function import_handler_ajax()
{
    //data[1] - артикул
    //data[2] - Постачальни
    //data[3] - Доставка
    //data[4] - Бренд
    //data[5] - Модель
    //data[6] - Призначення шин
    //data[7] - Тип - категорія
    //data[8] - Країна виробник
    //data[9] - Рік випуску
    //data[10] - Індекс
    //data[11] - Ширина
    //data[12] - Профіль
    //data[13] - Діаметр
    //data[14] - Сезон
    //data[15] - Залишок
    //data[16] - Ціна
    $arrtest = [];
    $result = process_import_file();
    foreach ($result as $row => $data) {
        if (!empty(trim($data[1]))) {
            $data_sku = $data[1];
            $product_id = wc_get_product_id_by_sku($data_sku);
            $product_price = extractNumber( $data[16] );
            $product_title = trim( $data[4] ) . ' ' . trim( $data[5] );
            if( $data[6] == 'Диск' ) {
                $product_title = $product_title . ' ' . trim( $data[11] ) . ' R' . trim( $data[13] );
            } else {
                $product_title = $product_title . ' ' . trim( $data[11] ) . '/' . trim( $data[12] ) . ' R' . trim( $data[13] );
            }
            if ( $product_id ) { // Якщо співпав артикул то обновляємо всі дані для товару
                $product = wc_get_product($product_id); // Отримання всіх характеристик товару товару
                update_post_meta($product_id, '_price', trim($product_price)); //Оновлення ціни
                update_post_meta($product_id, '_regular_price', trim($product_price)); //Оновлення ціни
                if ($data[6] == 'Диск') {
                    update_brands('brend-dyskiv', $product_id, trim($data[4])); //Оновлення бренду для дисків
                    update_brands('brend', $product_id, ''); //Затерання бренду для дисків в атрибутах для шин
                    update_product_data('typ', $product_id, trim($data[7])); // Оновлення типу для дисків
                    update_product_data('typ-osi', $product_id, ''); // Затарення типу для дисків в атрибутах для шин
                    update_product_data('shyryna-dyskiv', $product_id, trim($data[11])); // Оновлення ширини для дисків
                    update_product_data('shyryna', $product_id, ''); // Затерання ширини для дисків в атрибутах для шин
                } else {
                    update_brands('brend', $product_id, trim($data[4])); //Оновлення бренду для шин
                    update_brands('brend-dyskiv', $product_id, ''); // Затерання ширини для шин в атрибутах для дисків
                    update_product_typ_osi('typ-osi', $product_id, trim($data[7])); // Оновлення типу для шин
                    update_product_data('typ', $product_id, ''); // Затерання типу для шин в атрибутах для дисків
                    update_product_data('shyryna', $product_id, trim($data[11])); // Оновлення ширини для шин
                    update_product_data('shyryna-dyskiv', $product_id, ''); // Затерання ширини для шин в атрибутах для дисків
                }
                update_provider('postachalnyk', $product_id, trim($data[2])); //Оновлення постачальника
                update_delivery('dostavka', $product_id, trim($data[3])); //Оновлення доставки
                update_product_data('model', $product_id, trim($data[5])); // Оновлення моделю
                update_product_data('krayina-vyrobnyk', $product_id, trim($data[8])); // Оновлення країни виробника
                update_product_data('rik-vyrobnycztva', $product_id, trim($data[9])); // Оновлення року виробництва
                update_product_data('indeks-shvydkosti', $product_id, trim($data[10])); // Оновлення індексу
                update_product_data('profil', $product_id, trim($data[12])); // Оновлення профілю
                update_product_diametr('diametr', $product_id, trim($data[13])); // Оновлення діаметру
                update_product_seazon('sezon', $product_id, trim($data[14])); // Оновлення сезону
                update_product_count($product_id, trim($data[15])); // Оновлення кількісті товару
                update_product_category($product_id, trim($data[6])); // Оновлення категорії товару
                $product->set_name( $product_title );
                $product->save();
            } else {
                $product = new WC_Product();
                $product->set_name($product_title);
                $product->set_status('publish');
                $product->set_regular_price(trim($product_price));
                $product->set_price(trim($product_price));
                $product->set_sku(trim($data[1]));
                $product_id = $product->save();

                if ($data[6] == 'Диск') {
                    update_brands('brend-dyskiv', $product_id, trim($data[4])); //Оновлення бренду для дисків, update_brands потрібен бути першим для створення
                    update_product_data('typ', $product_id, trim($data[7])); // Оновлення типу для дисків
                    update_product_data('shyryna-dyskiv', $product_id, trim($data[11])); // Оновлення ширини для дисків
                } else {
                    update_brands('brend', $product_id, trim($data[4])); //Оновлення бренду для шин, update_brands потрібен бути першим для створення товарів
                    update_product_typ_osi('typ-osi', $product_id, trim($data[7])); // Оновлення типу для шин
                    update_product_data('shyryna', $product_id, trim($data[11])); // Оновлення ширини для шин
                }
                update_provider('postachalnyk', $product_id, trim($data[2])); //Оновлення постачальника
                update_delivery('dostavka', $product_id, trim($data[3])); //Оновлення доставки
                update_product_data('model', $product_id, trim($data[5])); // Оновлення моделю
                update_product_data('krayina-vyrobnyk', $product_id, trim($data[8])); // Оновлення країни виробника
                update_product_data('rik-vyrobnycztva', $product_id, trim($data[9])); // Оновлення року виробництва
                update_product_data('indeks-shvydkosti', $product_id, trim($data[10])); // Оновлення індексу
                update_product_data('profil', $product_id, trim($data[12])); // Оновлення профілю
```

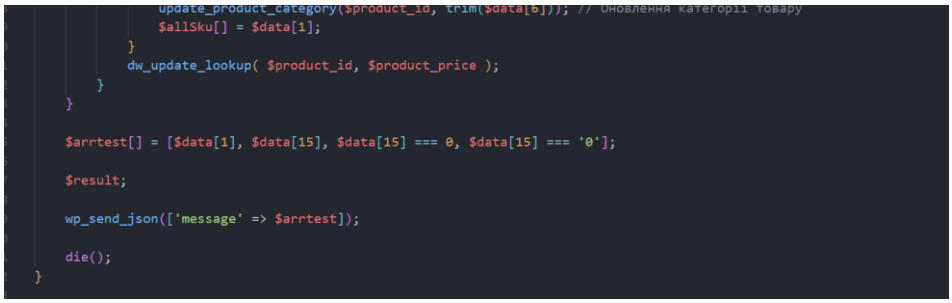


Рисунок 4.38 – Додавання нових товарів, та оновлення існуючих товарів

ВИСНОВКИ

Під час виконання кваліфікаційної роботи магістра було розроблено вебсайт. Вебсайт був розроблений згідно вимог замовника, що зазначені у завданні:

- адаптивна верстка вебсайту;
- відображення актуального асортименту та актуальної ціни товарів;
- управління замовленнями;
- наявність форми оформлення замовлення;
- наявність повної інформації про товари, які надані клієнтом;
- можливість перегляду статусів товарів;
- можливість додавання товару за допомогою файлу, який містить актуальні товари та ціну.

Було проведено детальний аналіз предметної області: визначено питання актуальності проблеми, сформовано мету та задачі для реалізації вебсайту, та детальний план по реалізації, проведено аналіз існуючих технологій розробки.

Також було проведено проектування вебсатйу, що дозволило повноцінно та детально вивчити процес розробки вебсайту. На даному етапі роботи було розроблено діаграму варіантів використання, що відображає головних користувачів цієї системи та їх сценарії дій з вебсайтом.

Під час розробки були виявлені проблеми вебсайту, та виправленні до впровадження вебсайту у використання.

Результати кваліфікаційної роботи магістра роботи були впроваджені в роботу для вебсайту “shinahub.com.ua”.

Апробація результатів дослідження кваліфікаційної роботи опублікована у двох тезах доповіді на конференції: IV Всеукраїнська студентська наукова конференція «Розвиток освіти, науки та бізнесу» і на 28-Й МОЛОДІЖНИЙ МІЖНАРОДНИЙ ФОРУМ «РАДІОЕЛЕКТРОНІКА І МОЛОДЬ В XXI СТОЛІТТІ» Харків.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. А.Г. Микитишин, М.М. Митник, П.Д. Стухляк, В.В. Пасічник. КОМП’ЮТЕРНІ МЕРЕЖІ. – Львів: Магнолія, 2006.

2. В. Чернега, Б. Платтнер. Безпровідні локальні комп'ютерні мережі. Навчальний посібник – Кондор, 2008.

3. Б.Ю. Жураковський, І.О. Зенів. КОМП’ЮТЕРНІ МЕРЕЖІ ЧАСТИНА 1 НАВЧАЛЬНИЙ ПОСІБНИК. – Київ, 2020.

4. Professional WordPress Plugin Development / B. Williams et al. Wiley & Sons, Incorporated, John, 2020. 560 p.

5. Joel M. Murach's MySQL / Murach Joel., 2019. – 646 c. – (Mike Murach and Associates, Incorporated)

6. Riaz A. Full Stack Web Development For Beginners: Learn Ecommerce Web Development Using HTML5, CSS3, Bootstrap, JavaScript.

7. Nixon R. Learning PHP, MySQL & JavaScript 5e : With jQuery, CSS & HTML5 / Robin Nixon. – Sebastopol, United States, 2018. – 800 c. – (O'Reilly Media, Inc, USA).

8. Williams A. WordPress for Beginners 2021 : A Visual Step-by-Step Guide to Mastering WordPress / Andy Williams., 2020. – 256 c. – (Independently Published).

9. Алексенко О. В. Технології програмування та створення програмних продуктів : конспект лекцій / Ольга Василівна Алексенко. – Суми: СумДУ, 2017. – 161 с.

10. Burge S. WooCommerce Explained: Your Step-by-Step Guide to WooCommerce / S. Burge, P. Rauland., 2017. – 311 c.

11. Williams B. Professional WordPress Plugin Development / B. Williams, J. Tadlock, J. James Jacoby. – Hoboken, United States: John Wiley and Sons Ltd, 2020. – 480 c. – (John Wiley and Sons Ltd).

12. Project Management Institute. A guide to the Project Management Body of Knowledge (PMBOK guide) and the Standard for project management / Project

Management Institute. – Newton Square, PA, United States: Project Management Institute, 2021. – 250 с. – (Project Management Institute).

13. Messenlehner B. Building Web Apps with WordPress 2e : WordPress as an Application Framework / B. Messenlehner, J. Coleman. – Sebastopol, United States, 2020. – 400 с. – (O'Reilly Media, Inc, USA).

14. Інструмент створення блогів, платформа для публікацій і CMS - WordPress.org Україна. WordPress.org Україна. URL: https://uk.wordpress.org/

15. WooCommerce - Open Source ecommerce Platform. WooCommerce. URL: https://woocommerce.com/.