Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет *Кібербезпеки*
(повна назва)

Кафедра *Інформаційно-мережної інженерії*
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Рівень вищої освіти *перший (бакалаврський)*

Створення веб-платформи для підприємства «ЄвроМеблі+»
(тема)

Виконав:
здобувач 4 року навчання,
групи *ТРІМІ-22-2*
Данило Кіт
(власне ім'я, прізвище)
Спеціальність *172 Телекомунікації та радіотехніка*
(код і повна назва спеціальності)
Тип програми *Освітньо-професійна*
Освітня програма *Інформаційно-мережна інженерія*
(повна назва освітньої програми)
Керівник *Ст. викл. Олександр Малінін*
(посада, власне ім'я, прізвище)

Допускається до захисту

В.о. завідувача кафедри _____ _____ Дарія ЧЕБОТАРЬОВА
(підпис) (власне ім'я, прізвище)

2026 р.

Харківський національний університет радіоелектроніки

Факультет *Кібербезпеки*
(повна назва)

Кафедра *Інформаційно-мережної інженерії*
(повна назва)

Рівень вищої освіти *перший (бакалаврський)*

Спеціальність *172 Телекомунікації та радіотехніка*
(код і повна назва)

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

Освітня програма *Інформаційно-мережна інженерія*
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри ІМІ_____
(підпис)
"____" _____2026 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Здобувачеві *Кіту Данилу Володимировичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *Створення веб-платформи для підприємства «ЄвроМеблі+»* затверджена наказом університету від *02 червня 2026* року № *466 Ст*
2. Термін подання студентом роботи до екзаменаційної комісії *16 липня 2026 р.*
3. Вихідні дані до роботи
Віртуальна машина на основі Ubuntu, Веб-сервер, База даних, phpMyAdmin, HTML, Серверна мова програмування, CSS, JavaScript, описати структуру та дизайн платформи, частину коду проєкту.
4. Перелік питань, що потрібно опрацювати в роботі
Вступ
1. Опис технологій проєкту
2. Розробка веб-платформи
3. Організування захисту від SQL-ін'єкцій і XSS атак
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, постановка задачі, актуальність роботи, завдання роботи, технології проєкту, розробка веб-платформи, висновки)*

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

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Основна частина	Ст. викл. Малінін О.П.		10.07.2026

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів атестаційної роботи	Строк виконання етапів роботи	Примітка
1	*Ознайомлення із завданням. Уточнення ТЗ*	*25.05.26*	*виконано*
2	*Підбір літератури за темою роботи*	*26.05-28.05.26*	*виконано*
3	*Виконання розділу 1*	*29.05-10.06.26*	*виконано*
4	*Виконання розділу 2*	*10.06-22.06.26*	*виконано*
5	*Оформлення пояснювальної записки*	*23.06-27.06.26*	*виконано*
6	*Оформлення презентаційного матеріалу*	*28.06-30.06.26*	*виконано*
7	*Підготовка до захисту у ЕК*	*10.07-16.07.26*	*виконано*

Дата видачі завдання 25 травня 2026 р.

Здобувач _____
(підпис)

Керівник роботи _____ ст. викл. *Олександр Малінін*
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 76с., 82 рис., 20 джерел, 1 додаток.

Об'єкт дослідження – веб-платформа для підприємства «ЄвроМеблі+».

Мета роботи – описати процес розробки веб-платформи для підприємства «ЄвроМеблі+», тобто, задачі є наступними: проаналізувати технології, компоненти проєкту, описати структуру та дизайн платформи, надати приклад оформлення дизайну платформи, надати частину коду проєкту.

Результати – описано процес створення веб-платформи, проаналізовано і порівняно компоненти, на базі чого зроблено вибір щодо використання технологій реалізації проєкту, надано дизайн сторінки, надано код проєкту.

КЛЮЧОВІ СЛОВА: ВЕБ-ПЛАТФОРМА, МЕБЕЛЬНЕ ПІДПРИЄМСТВО, ПІДПРИЄМСТВО, ЄВРОМЕБЛІ+, РЕЛЯЦІЙНА БАЗА ДАНИХ, LEMP, PHP, JAVASCRIPT, HTML, CSS, PHPMYADMIN.

THE ABSTRACT

Explanatory note: 76 p., 82 fig., 20 sources, 1 app.

The object of the study is a web platform for the company EuroMebli+.

Purpose - to describe the process of developing a web platform for the company “EuroMebli+”, the tasks are as follows: to analyze the technologies and project components, describe the platform's structure and design, provide an example of the platform's visual design, and provide part of the project code.

Deliverables: describe the process of creating web platform page, analyze and compare the components, based on which a choice is made about the use of technologies for the project, provide a page design, provide the project code.

KEYWORDS: WEB PLATFORM, FURNITURE COMPANY, ENTERPRISE, EUROFURNITURE+, RELATIONAL DATABASE, LEMP, PHP, JAVASCRIPT, HTML, CSS, PHPMYADMIN.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ……..…..... 8
ВСТУП ……..…… 9
1 ОПИС ТЕХНОЛОГІЙ ПРОЄКТУ …….……... 10
1.1 Віртуальна машина на основі Ubuntu ..……..... 10
1.2 Веб-сервер ……... 11
1.3 Мова HTML ……. 11
1.4 Мова CSS ……..... 12
1.5 Мова JavaScript …….……... 14
1.6 Серверна мова програмування …….. 15
1.7 Вибір бази даних ……. 16
1.8 Застосунок для адміністрування бази даних phpMyadmin ……. 17
2 РОЗРОБКА ВЕБ-ПЛАТФОРМИ …….. 18
2.1 Постановка завдання …….. 18
2.1.1 Постановка технічного завдання до розробки макету веб-платформи ….. 19
2.1.2 Поставка технічного завдання для написання коду веб-платформи та верстки ……... 19
2.2 Дизайн веб-платформи і верстка ……... 19
2.2.1 Фінальний результат дизайну веб-платформи підприємства «ЄвроМеблі+» ...…… 22
2.2.2 Верстка веб-платформи…… 22
2.3 Конфігурація віртуальной машини і установка необхідних пакетів …….. 22
2.3.1 Підключення до віртуальної машини і оновлення всіх пакетів …….. 22
2.3.2 Встановлення LEMP стеку …….. 23
2.3.3 Конфігурація веб-сервера Nginx для використання процесору PHP …….. 26
2.3.4 Встановлення і конфігурування phpMyadmin ……... 28
2.3.5 Створення бази даних……... 30

2.4 Код веб-платформи…….. 33
2.4.1 Архітектура робочої папки проєкту ……... 33
2.4.2 Код головної сторінки веб-платформи ……... 34
2.4.3 Код сторінки з кошиком …….. 45
2.4.4 Код сторінки з адміністративною панеллю…… 49
ВИСНОВКИ …….. 58
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ ...…… 59
ДОДАТОК А. СЛАЙДИ ПРЕЗЕНТАЦІЇ…… 61

ПЕРЕЛІК СКОРОЧЕНЬ

OS (Operating System або операційна система) - це базове програмне забезпечення, яке керує комп'ютером;

HTML (HyperText Markup Language) – це спеціальна мова розмітки, яка застосовується при створенні сайтів в інтернеті;

CSS (Cascading Style Sheets або каскадні таблиці стилів) – це мова візуального представлення вмісту HTML-документа;

PHP (Hypertext Preprocessor) – це скриптова мова програмування з відкритим вихідним кодом;

БД (База даних) – сукупність даних, організованих відповідно до концепції, яка описує характеристику цих даних і взаємозв'язки між їх елементами;

SQL (Structured Query Language) – це стандартна мова запитів, яка використовується для роботи з реляційними базами даних;

NoSQL (Not Only SQL) – це група баз даних, які відрізняються від традиційних реляційних баз даних (SQL) способом організації, зберігання та доступом до даних;

GUI (Graphical User Interface) – це спосіб взаємодії користувача з комп'ютером із використанням графічних елементів;

XSS (Cross Site Scripting) – тип кібератаки, при якій зловмисники додають або змінюють код сторінки сайту;

UI (User Interface) – це користувацький інтерфейс, тобто всі візуальні та інтерактивні елементи, з якими людина взаємодіє під час користування сайтом;

UX (User Experience або досвід використання) – це те, що людина відчуває при користуванні продуктом, системою чи сервісом;

LEMP (Linux, Nginx, mySQL, PHP) – це комплекс технологій та відкритих інструментів, що використовується для розгортання та хостингу високопродуктивних веб додатків;

PDO (PHP Data Objects або об'єкти даних PHP) – розширення для PHP, що надає розробнику простий і універсальний інтерфейс для доступу до різних баз даних.

ВСТУП

Сучасний стан розвитку ринку меблевої продукції в Україні вимагає від підприємств високої конкурентоспроможності, мобільності і активної присутності у цифровому просторі.

Традиційні методи маркетингу і способи задля залучення нових клієнтів втрачають свою ефективність і актуальність, поступаючись платформам в яких є цифровий варіант оформлення замовлення, та можливість самообслуговування.

Дивлячись на статистику меблевих підприємств, трохи більше ніж 50% меблевих підприємств має свою веб-платформу.

Тому за цим для підприємства «ЄвроМеблі+», з’явилась необхідність у створенні веб-платформи, бо основним напрямком діяльності підприємства є продаж своїх меблів і виготовлення с подальшою їх реалізацією. Також, окрім реалізації своїх меблів, підприємство має намір продавати і просувати меблі інших брендів, тому створення власної веб-платформи є критично важливим рішенням.

Це обумовлено необхідністю надання потенційним клієнтам цілодобового доступу до віртуального каталогу продукції, детальних характеристик матеріалів, фурнітури та готових рішень без обов'язкового відвідування фізичного офісу чи виставкового залу. А продаж і просування меблів інших брендів на своїй майбутній платформі аргументовано тим що підприємство виготовляє не всі типи мебельної продукції, і за допомогою надання розміщення іншим брендам отримує можливість як і додаткового заробітку так і активного просування своєї веб-платформи в мережі інтернет.

1 ОПИС ТЕХНОЛОГІЙ ПРОЄКТУ

Розробку сучасної веб-платформи можна здійснити за допомогою різноманітних технологій. Для такого підприємства, як «ЄвроМеблі+», вибір відповідних технологій є критично важливим для створення рішення, яке відповідатиме стандартам нашого часу та зможе забезпечити кінцевий результат, який ставить замовник підприємства.

Отже, в цьому розділі описані технології проєкту які були використані під час розробки веб-платформи.

1.1 Віртуальна машина на основі Ubuntu

Віртуальна машина – це програмний комп’ютер, який працює всередині справжньої фізичної системи. Вона має власні віртуальні процесори, пам’ять та жорсткий диск [1].

Необхідність використання технологій віртуалізації у даному проекті обумовлена технічними обмеженнями персонального комп’ютера, на якому здійснювалась розробка веб-платформи. Для вирішення цієї проблеми та створення адекватного середовища розробки, було використано віддалений виділений сервер, на якому було виділено ізольований віртуальний простір, на якому була розгорнута OS Ubuntu. Нижче наведено приклад як працює виділення ресурсів з віддаленого виділеного сервера для віртуальної машини (рис. 1.1).

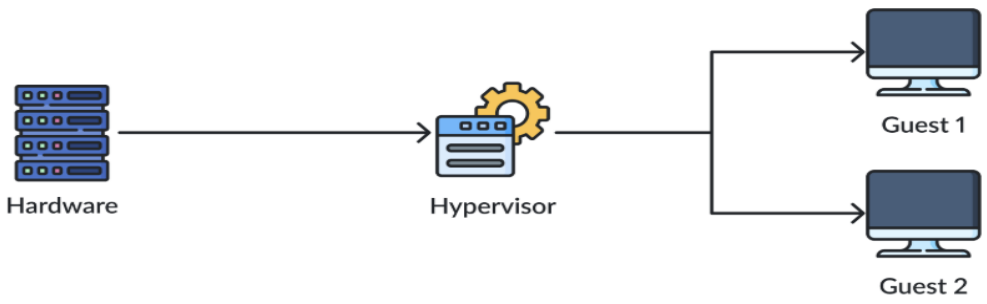


Рисунок 1.1 – Приклад виділення ресурсів з виділеного сервера для віртуальної машини.

Вибір цієї операційної системи, надав переваги у можливості ефективно використовувати ресурси віртуальної машини, бо завдяки перенесенню всіх

серверних процесів на віддалений сервер, локальний ПК розробника використовувався лише задля комунікації з сервером через термінал, та також ця OS на відміну від інших має регулярність і прогнозованість оновлень, що надає велику перевагу у безпеці.

Також такий варіант відповідає реальним умовам, так як більшість сучасних веб-платформ у промисловій експлуатації розміщується саме на серверних OS сімейства Linux (зокрема Ubuntu Server). Розробка саме у цьому середовищі дозволяє уникнути проблем із сумісністю шляхів, прав доступу до кореневих файлів платформи і конфігурацій Nginx.

Таким чином, використання віртуальної машини на віддаленому виділеному сервері з встановленою туди OS Ubuntu, вирішило проблему з технічними обмеженнями ПК розробника, і зробило розробку проєкту наближеним до реальних умов промислових платформ.

1.2 Веб-сервер

В веб просторі є декілька варіантів веб-серверів, наприклад, таких як Apache, Nginx. Історично HTTP-сервер Apache був домінуючою технологією з відкритим вихідним кодом, яку цінували за гнучкість і модульну систему конфігурації. Однак, Apache традиційно працює на керованій процесором моделі, виділяючи незалежний потік або процес для кожного активного мережевого з’єднання. У періоди великого напливу клієнтів такий підхід призводить до масового споживання пам’яті, і є фатальним для будь-якого проєкту який може швидко масштабуватись, та не є гарним вибором під підприємство «ЄвроМеблі+» за рахунок цих проблем [2].

В свою чергу, щоб вирішити проблеми продуктивності, і перейти до архітектури з керованими подіями, у проекті використовувався Nginx, який в свою чергу використовує асинхронну архітектуру яка керується подіями. Замість того, щоб створювати нові системні процеси для окремих HTTP-запитів, Nginx виконує неблокуючі операції в межах фіксованої кількості високооптимізованих робочих процесів [2]. Згідно з даними ринків, Nginx залишається найкращою технологією веб-серверів у всьому світі, забезпечуючи приблизно 31,8% усіх веб-сайтів із відомою реалізацією сервера [3].

Отже, Nginx було обрано як оперативний веб-сервер для підприємства

«ЄвроМеблі+» завдяки практичним перевагам у високій паралельності і стійкості до трафіку, та у многозадачності яка потребує меншу частку ресурсів серверу.

1.3 Мова HTML

Мова розмітки гіпертексту HTML має багато різноманітних версій. Історично веб-розробка спиралася на застарілі стандарти, кожен з яких мав архітектурні обмеження, які безпосередньо впливають на сучасні веб-додатки [4]:

HTML 2.0 та 3.2 – були розроблені в середині 1990-х років, ці стандарти були розроблені в першу чергу для статичної текстової документації. Вони практично не пропонували гнучкості макета, також не мали стандартизації в різних веб-браузерів.

HTML 4.01 – була розроблена до ери динамічних веб-сайтів чи додатків, хоча ця версія успішно відокремила презентацію від структури через CSS. Основним недоліком цієї версії є відсутність рідних мультимедіа та розширена обробка даних, що змушує розробників значною мірою покладатися на важкі, незахищені сторонні плагіни, наприклад такі як Adobe Flash, для інтерактивних функцій.

Сама по собі мова розмітки до 2000 року періодично оновлювалась, і версії такі як 4.01 і 5 вже вийшли після 2010 року (рис. 1.2).



Рисунок 1.2 – Історія випуску версій HTML

Після детального аналізу і зваженню усіх наявних версій, був вибраний оптимальний стандарт HTML5, в якому усунуті недоліки своїх попередників.

Використання саме п'ятої версії стандарту HTML надало переваги у семантичній розмітці тим що має спеціальні семантичні теги (таких як <header>, <nav>, <article>, <footer>, <section>) , ефективній роботі с медіа-контентом (HTML5 дозволяє інтегрувати зображення, адаптивні галереї та відеоогляди), адаптивності та кросплатформеності (теги і елементи HTML5 коректно працюють з усіма сучасними веб-браузерами) [5].

Для платформи «ЄвроМеблі+» важливим елементом є форми зворотнього зв'язку, і HTML5 з цим гарно справляється за допомогою вбудованих атрибутів для перевірки коректності введених даних користувачем (наприклад, поля required, перевірки формату пошти type=”email”).

Таким чином, мова HTML5 забезпечує створення чистого, стандартизованого і підтримуваного коду, який гарантує швидку та стабільну роботу інтерфейсу користувача.

1.4 Мова CSS

CSS – це каскадні таблиці стилів (Cascade Styling Sheets), які працюють в парі з HTML, бо по самій суті HTML виступає в ролі первинного контенту веб-платформи бо надає їй структуру, яка є необробленою спочатку. А CSS в свою чергу обробляє і забезпечує веб-сторінці візуальну складову (рис. 1.3).

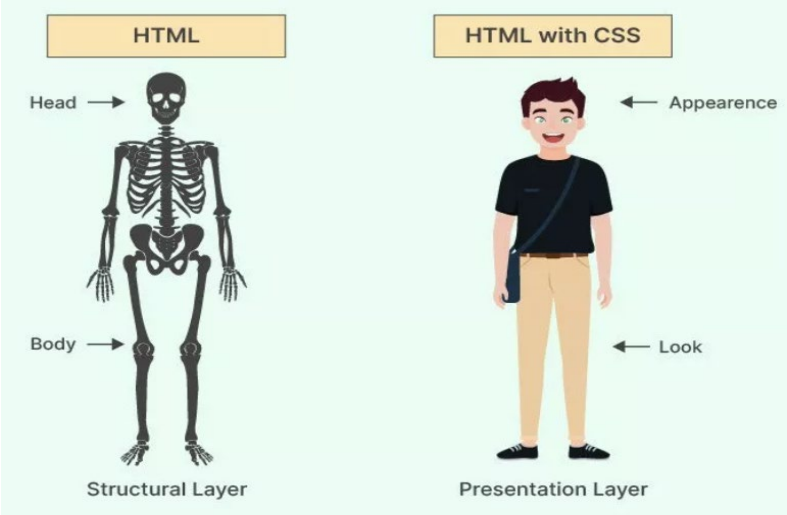


Рисунок 1.3 – Пояснення взаємодії HTML з CSS

Якщо казати про сам CSS, він спочатку оновлювався в масивних і повільних версіях, тобто, одразу багато і дуже рідко (наприклад, CSS 1 і CSS 2.1). Проблему з цим старим підходом полягала в тому, що якщо розробники хотіли нову функцію, їм доводилося роками чекати оновлення всієї глобальної специфікації. Щоб на той час це виправити, творці павутини перестали робити монолітні версії. Сьогодні CSS розбивається на незалежні модулі які оновлюються самостійно. Ця модульна система – це те, що дозволяє використовувати сучасні потужні інструменти проектування, не чекаючи виходу CSS4 чи CSS5 [6]. Оскільки, сучасна CSS використовує цю систему нам не потрібно більше шукати єдине число версії. Замість цього, вибір останньої версії означає вибір найсучасніших, стабільних модулів, доступних сьогодні.

Із популярних модулей і доступних на даний момент, є такі як [7]:

– Модуль Flexbox: використовується для організації менших компонентів на сторінці, таких як навігаційне меню, кнопки та карти продуктів. Це гарантує що елементи ідеально вирівняються та автоматично адаптуються до різних розмірів екрану, не порушуючи макет.

– Модуль CSS Grid: використовується для керування загальним макетом веб-сайту (наприклад, шапкою, розділом каталогу продуктів і нижнього колонтитула). Сітка дозволяє побудувати чистий багатоколонковий макет, який залишається ідеально вирівняним.

Отже, під час роботи над веб-платформою для підприємства «ЄвроМеблі+», використання CSS останніх версій і популярних модулей Flexbox, CSS Grid покращить швидкість розробки платформи, і зробить майбутній проект сучасним.

1.5 Мова JavaScript

JavaScript – це інтерпретована, легка мова програмування яка використовується в основному для обробки логічних операцій на стороні клієнта, які виконуються безпосередньо в браузері користувача, а не на сервері [8]. Приклад роботи JavaScript зображено на фото нижче (рис. 1.4)

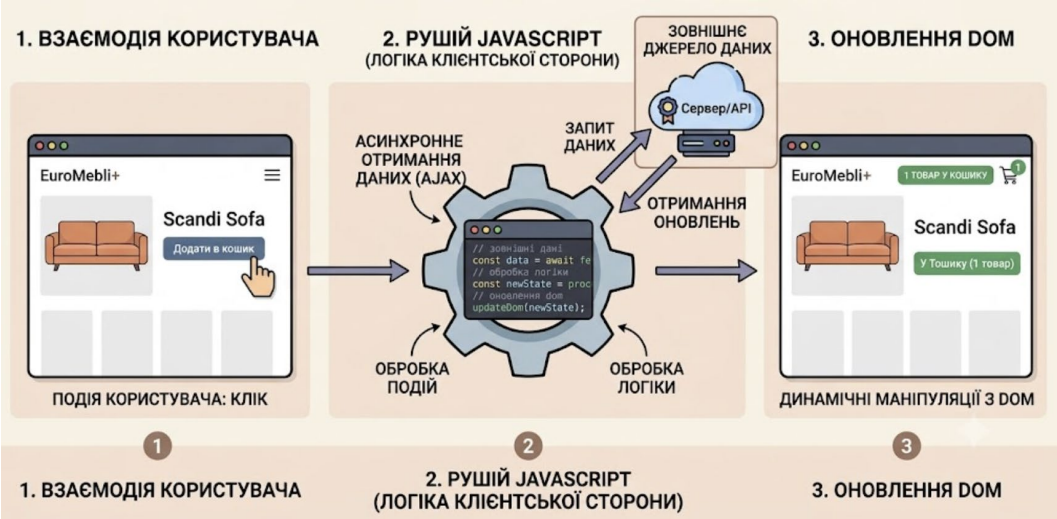


Рисунок 1.4 – Приклад роботи JavaScript

JavaScript діє як міст між користувачем і структурою сторінки. Це дозволяє платформі динамічно оновлювати вміст, не вимагаючи повного оновлення сторінки (наприклад, оновлення кількості кошиків для покупок або фільтрації списків продуктів). Також, мова активно прослуховує дії клієнта – такі як клацання, надсилання форм або вибір різних параметрів розміщених на сайті, і негайно запускає відповідну інтерактивну поведінку (наприклад, заміняє стиль кнопок, дизайну, відправляє спливаюче повідомлення) [8].

За допомогою асинхронного зв’язку (AJAX) браузер може мовчки обмінюватися даними з сервером у фоновому режимі. Це дозволяє платформі отримувати дані в режимі реального часу, такі як – наявність запасів, відгуки клієнтів, тощо. І в цей час не перериваючи взаємодію з користувачем [9].

Отже, мова програмування високого рівня JavaScript була використана у проекті для розроблення клієнтського сценарію та забезпечення високого рівня динамічності і відгукливості інтерфейсу користувача веб-платформи підприємства. Основне призначення JavaScript у цьому проекті є управління поведінкою елементів веб-сторінки у реальному часі безпосередньо у браузері користувача.

1.6 Серверна мова програмування

У сучасному світі на цей час, є багатий перелік мов програмування для бекендної частини. Наприклад, такі як Python – який широко відомий своїм

чистим синтаксисом і можливостями обробки даних у поєднанні з такими фреймворками, як Django та Java, традиційний стандарт для масивних, і в цей час складних корпоративних систем. Крім того, є ще такі середовища як NodeJS, які дозволяють використовувати асинхронні програми, керовані подіями, використовуючи уніфікований JavaScript у стеку.

Незважаючи на ці параметри, мова PHP була обрана як основна серверна мова для цього проєкту. У той час як поява нових фреймворків призвела до помилкової думки про те, що PHP є застарілою і не актуальною мовою, хоча дивлячись на статистику останніх років, PHP все ще у десятці, а іноді і у п’ятірці найпопулярніших мов програмування, і залишається домінуючим механізмом Інтернету, який живить приблизно 71% усіх веб-сайтів відомою мовою на стороні сервера. Крім того, випуск сучасних ітерацій, таких як PHP 8.x, представив об’єктивно-орієнтовані вдосконалення, суворі натяки на тип і компіляцію Just-In-Time, перетворивши її на високоефективне та безпечне середовище для комерційного використання. Також якщо дивитись статистику використовування версій PHP, 8.x є масово використовуваною среди інших версій, що підтверджує те що вона є стабільною (рис. 1.5) [10, 11].

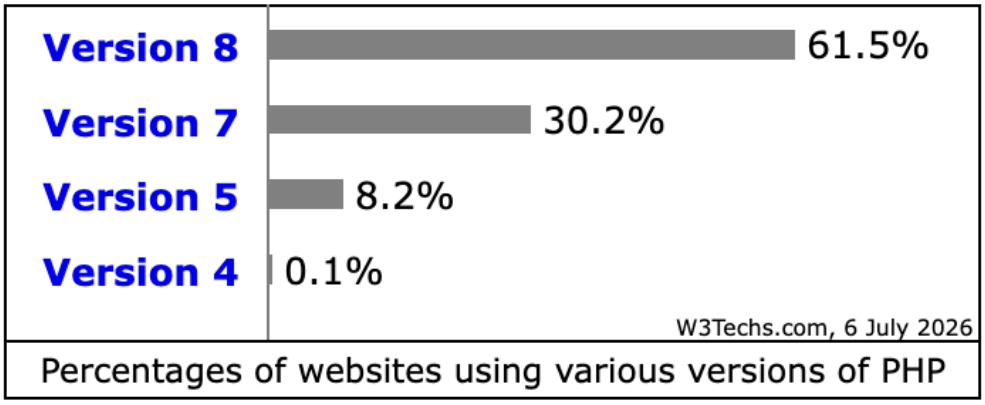


Рисунок 1.5 – Статистика використання версій PHP

Для підприємства «ЄвроМеблі+», PHP надає три ключові практичні переваги: він має зрілу екосистему електронної комерції з попередньо створеними і перевіреними великим часом компонентами для каталогів і кас, забезпечує низькі витрати на інфраструктуру завдяки універсальній сумісності зі стандартним веб-сервером і пропонує величезний пул талантів, який в свою

чергу гарантує швидкість розробки та легке довгострокове обслуговування системи.

1.7 Вибір бази даних

Для збереження даних і структурування і необхідної інформації, веб-платформі необхідна БД. У сучасній розробці програмного забезпечення і веб-сайтів технології збереження даних поділяються на реляційні (SQL) і нереляційні (NoSQL) системи. Нереляційні, такі як MongoDB, організовують дані за допомогою гнучких схем на основі документів, і є оптимізованими для неструктурованих або швидко мінливих моделей даних. В свою чергу, реляційні системи, такі як mySQL чи PostgreSQL, широко визнані за обробку дуже складних аналітичних запитів і пропонують суворе дотримання стандартів SQL. Ключові відмінності між SQL і NoSQL (рис. 1.6) [12].

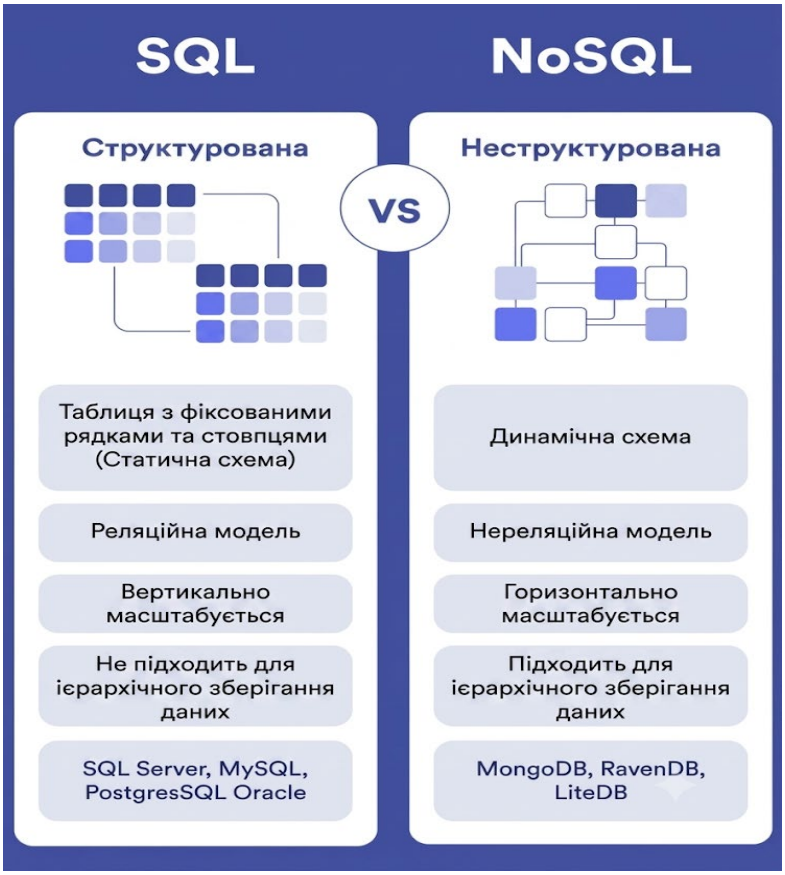


Рисунок 1.6 – Відмінності роботи структури даних SQL та NoSQL

Дивлячись на майбутню веб-платформу для підприємства «ЄвроМеблі+», яка має в свою чергу велику необхідність підв’язувати унікальний ідентифікатор клієнта, наприклад, до рахунку-фактури чи прив’язки певних розмірів до предмету мебелі, вибір зупинився на реляційному типу бази даних.

У реляційному типу баз даних, є два найвідоміших варіанта з відкритим кодом: PostgreSQL, MySQL. PostgreSQL – це об’єктно-реляційна база даних, популярна своїми розширеними аналітичними можливостями і підтримкою складних користувацьких типів даних. Однак, PostgreSQL вимагає більш складної конфігурації та створює більші накладні витрати на пам’ять сервера. А mySQL, навпаки, є оптимізованим для високочитаних веб-додатків, пропонує виняткову швидкість для стандартних запитів до баз даних, має простіше адміністрування і універсальну сумісність зі стеками веб-розробки [13].

Отже, було обрано MySQL як оптимальне рішення для підприємства «ЄвроМеблі+», бо згідно з даними ринку веб-технологій, mySQL залишається найбільш широко розгорнутою технологією баз даних що забезпечує базову інфраструктуру переважної більшості веб-платформ.

1.8 Застосунок для адміністрування бази даних phpMyadmin

Щоб забезпечити комфортне і ефективне керування реляційною базою даних веб-платформи та моніторингу стану БД і швидким зміненням структури таблиць було обрано платформу phpMyAdmin. Це спеціалізований додаток з відкритим вихідним кодом на PHP, який дає можливість виконувати адміністрування базою даних mySQL через графічний інтерфейс користувача (GUI) [14].

Вибір цього додатку серед інших, полягав у тому що, цей продукт має багато функцій для керування базою даних і комфортний і деталізований дизайн з коробки, чим той же Adminer який в свою чергу має бідніший дизайн, і менше кнопок для взаємодії з моніторингом стану БД.

Використання phpMyAdmin у технологічний комплекс розробки дозволяє мінімізувати ризики виникнення синтаксичних помилок при маніпуляції даними за допомогою SQL запитів та забезпечує наочність архітектури бази даних веб-платформи.

2 РОЗРОБКА ВЕБ-ПЛАТФОРМИ

2.1 Постановка завдання

Стоїть задача почати розробку багатофункціональної веб-платформи, яка буде виконувати наступні функції:

– представляти мебельне підприємство «ЄвроМеблі+» в мережах інтернету;

– створити можливість просування нових і готових меблевих виробів у маси за допомогою веб-платформи;

– створити можливість створення інтернет-замовлень через веб-платформу;

– створити можливість продавати мебельну продукцію інших виробників;

2.1.1 Постановка технічного завдання до розробки макету веб-платформи

Макет – це статичний, високоточний візуальний план інтерфейсу веб-сайту. Який на відміну від каркаса (який є базовим ескізом макетів сторінок), макет визначає точні естетичні та структурні компоненти сайту, включаючи кольорові палітрі майбутнього веб-сайту, відстань між елементами та навігаційні структури [15, 16].

Майбутній макет веб-платформи передбачає в собі враховувати усі найважливіші блоки, які далі можуть структурувати інші.

Отже, макет веб-платформи повинен відповідати таким критеріям:

– Використання гнучкої грід-системи, яка буде автоматично обчислювати скільки безпечно елементів може помістится на екрані користувача.

– Наявність мобільної реакції, де незалежно від того, чи відкриває користувач веб-платформу на настільному комп’ютері, планшеті чи мобільному телефоні. Елементи повинні плавно змінювати розмір і переставлятися, щоб текст розміщений на веб-платформі залишався читабельним, а кнопки можна було легко нажимати.

– Використання візуальної ідентичності і логотипу підприємства, а саме наявність зелено-білої кольорової палітри відображаючи цим фірмовий стиль підприємства, та також наявність логотипу підприємства у шапці платформи (header) і нижньої частини платформи (footer).

– Навігація веб-платформою у шапці веб-платформи (header), є одним із найважливіших пунктів, які допоможуть швидко розібратись у платформі майбутнім користувачам. Також біля навігації має бути вибір мови сторінки.

2.1.2 Поставка технічного завдання для написання коду веб-платформи та верстки

Веб-платформа має бути написана, використовуючи наступні технології:

– Віртуальна машина на основі ОС Ubuntu;
– Веб-сервер Nginx;
– Реляційна база даних mySQL;
– phpMyadmin;
– HTML та CSS і JavaScript;
– PHP;

Щоб запобігти вразливості безпеки веб-платформи, усі дані, що надходять із форм користувача (зворотний зв’язок, дані оформлення замовлення та інші), повинні пройти сувору перевірку та фільтрацію перед обробкою. Тобто, під час написання коду веб-платформи за допомогою мови програмування PHP з поєднанням JavaScript, треба дотримуватися усіх безпекових стандартів проти SQL-ін’єкцій, XSS-атак.

2.2 Дизайн веб-платформи і верстка

Дизайн веб-платформи для підприємства створювався за допомогою додатка Figma. Figma – це застосунок для роботи з векторною графікою, що дозволяє створювати різні макети, елементи інтерфейсу, зображення, ілюстрації, проектування і багато інших креативів. Цей інструмент популярний серед веб та UI/UX-дизайнерів, SMM-менеджерів, stories-мейкерів та інших digital-експертів. Бо він зручний для використання, доступний і тут можливо працювати над проєктом віддалено разом з іншими учасниками. Якщо колись для роботи із дизайном потрібно було встановлювати спеціальні програми, як-от Adobe

Photoshop, то тепер у Figma online можливо працювати прямо на сайті й не треба завантажувати окремий застосунок [17].

2.2.1 Фінальний результат дизайну веб-платформи підприємства «ЄвроМеблі+»

Загальні висновки щодо проробленої роботи над дизайном веб-платформи:

– в основі використані 8 кольорів: 1b4332, 000000, 40916c, 74c69d, d8f3dc, ffffff, 4a5568, 1a202c;

– використано шрифт Inter як основний, і Arial як запасний, якщо Inter не буде завантажено;

– виконано 4 сектори на головній сторінці;

Фінальний вигляд дизайну веб-платформи зображено на рис. 2.1 – 2.6.



Рисунок 2.1 – Дизайн заголовку платформи (header)

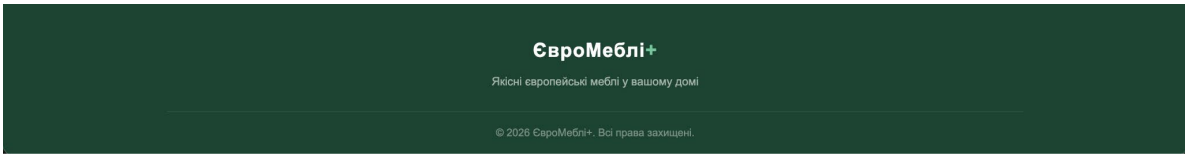


Рисунок 2.2 – Дизайн нижньої частини платформи (footer)



Рисунок 2.3 – Дизайн навігації на платформі

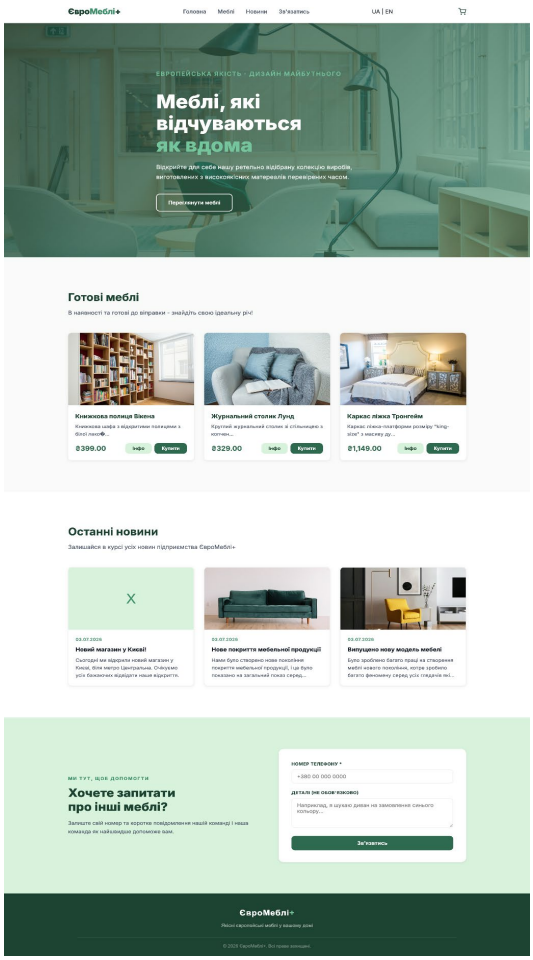


Рисунок 2.4 – Дизайн головної сторінки веб-платформи

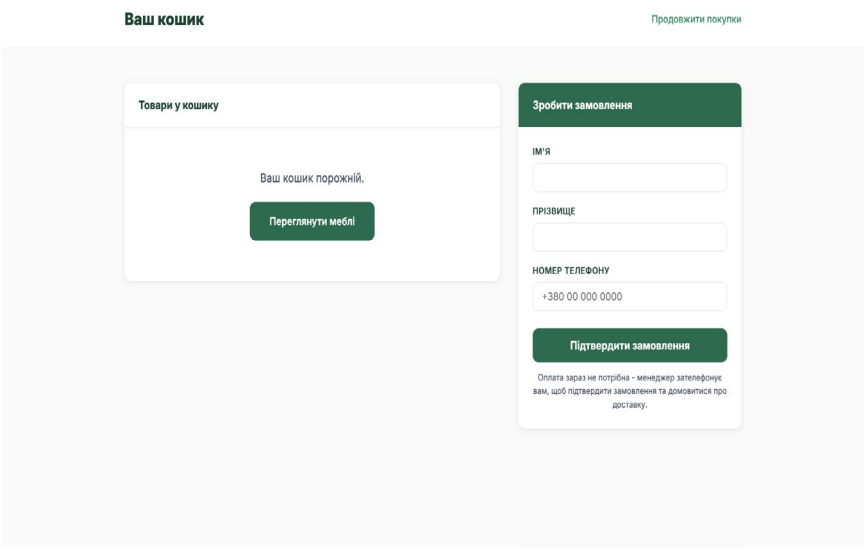


Рисунок 2.5 – Дизайн сторінки «Ваш кошик»

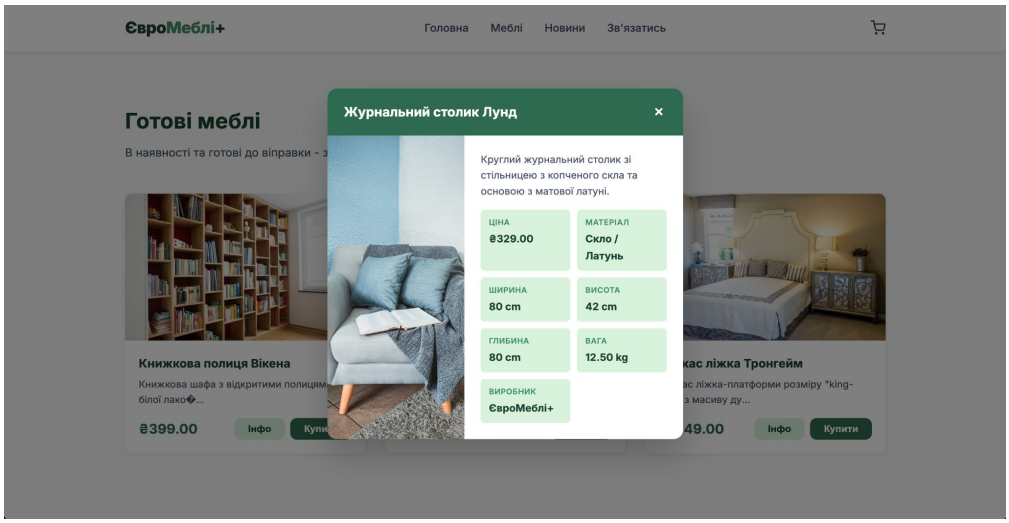


Рисунок 2.6 – Дизайн інфо блока про товар

2.2.2 Верстка веб-платформи

Верстка веб-платформи – це один із найважливіших моментів із розробки веб-платформи, в ході якого дизайн зображений у форматі дизайнерського рішення трансформується у HTML та CSS код.

На стадії повного завершення розробки дизайну веб-платформи, настає час верстки. Береться макет, створений у графічному редакторі, в випадку з розробкою веб-платформи для підприємства – це Figma, і перетворюється на робочий інструмент. З технічної точки зору, верстка передбачає в собі створення коду за допомогою HTML та CSS для кожної сторінки, у сумі це надає розпізнавання і відображення сайту так, як було задумано дизайнером [18].

Верстка також включає в себе створення структури веб-платформи і дизайну усіх рішень на сайті, тобто, в процесі верстки сайту розташовується текст, фотографії, модулі, блоки як було заплановано і це все підкріплюється стилізацією за допомогою каскадних таблиць стилів (CSS).

2.3 Конфігурація віртуальной машини і установка необхідних пакетів

Конфігурація віртуальної машини заключається у оновленні пакетів, у встановленні LEMP стеку який включає в себе встановлення пакетів веб-серверу Nginx, реляційної бази даних mySQL, та PHP.

2.3.1 Підключення до віртуальної машини і оновлення всіх пакетів

Щоб підключитись до віртуальної машини і розпочати конфігурування і установку необхідних пакетів, треба скористатися терміналом і ввести ssh root@IP, замість IP вводиться публічний адрес віртуальної машини. Після успішної авторизації зустрічає віддалене керування віртуальною машиною за допомогою консолі SSH (рис. 2.7).



Рисунок 2.7 – Управління сервером через консоль

Після успішної авторизації, віртуальній машині треба оновити всі пакети до останніх версій, та також усі програми. Для цього було введено такі команди як – apt update -y, apt upgrade -y, які оновлюють списки доступних пакетів з офіційних репозиторіїв. Результати оновлення пакетів зображено на рис. 2.8.

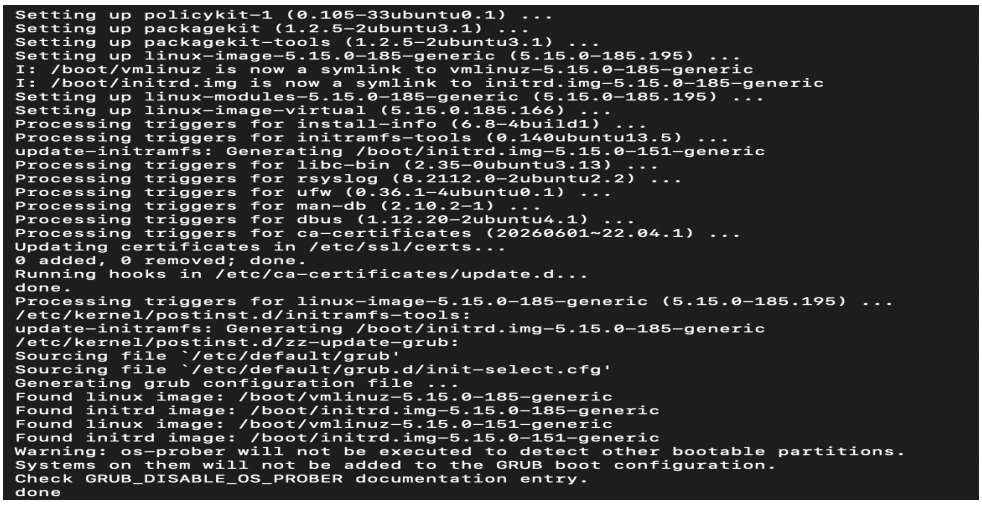


Рисунок 2.8 – Результат оновлення пакетів

2.3.2 Встановлення LEMP стеку

Встановлення веб-серверу Nginx, виконується командою – apt install nginx. Результати встановлення і успішної роботи веб-серверу (рис. 2.9 – 2.10).

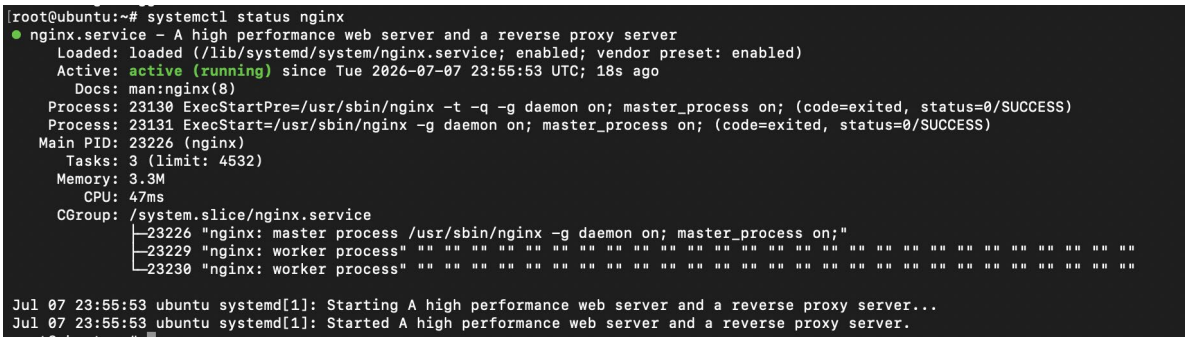


Рисунок 2.9 – Результат встановлення і успішної роботи веб-серверу Nginx

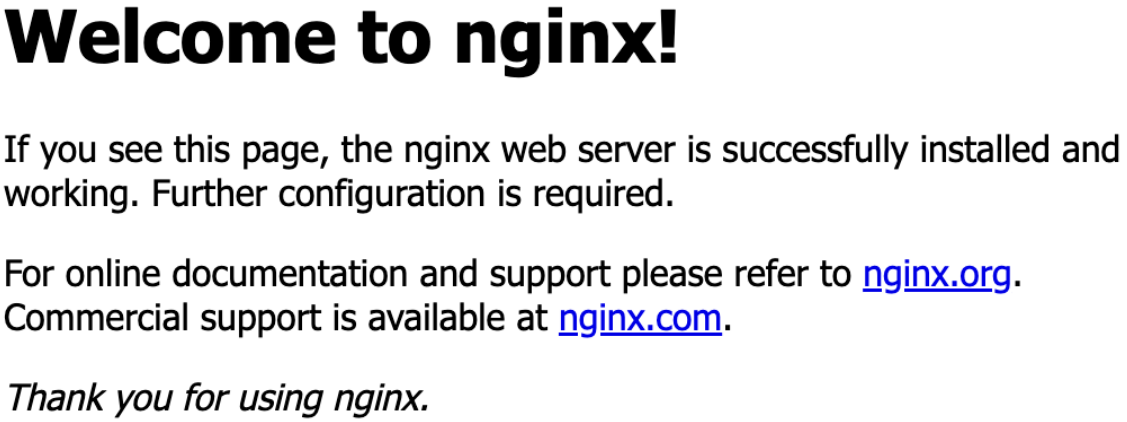


Рисунок 2.10 – Результат встановлення і успішної роботи веб-серверу Nginx

Встановлення пакету mySQL, виконується командою – apt install mysql-server. Результат встановлення і успішної роботи mySQL (рис. 2.11).

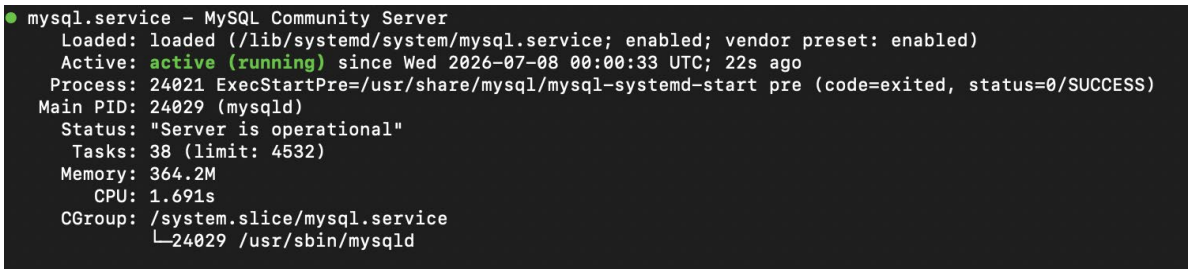


Рисунок 2.11 – Результати встановлення реляційної бази даних mySQL

Після встановлення пакету реляційної бази даних, необхідно зробити безпечний доступ до бази даних за допомогою конфігуруючої команди – mysql_secure_installation. А також необхідно створити першого адміністративного користувача в реляційній базі даних mySQL. Процес конфігурації mySQL, і створення першого адміністративного користувача зображено на рис. 2.12 – 2.14.

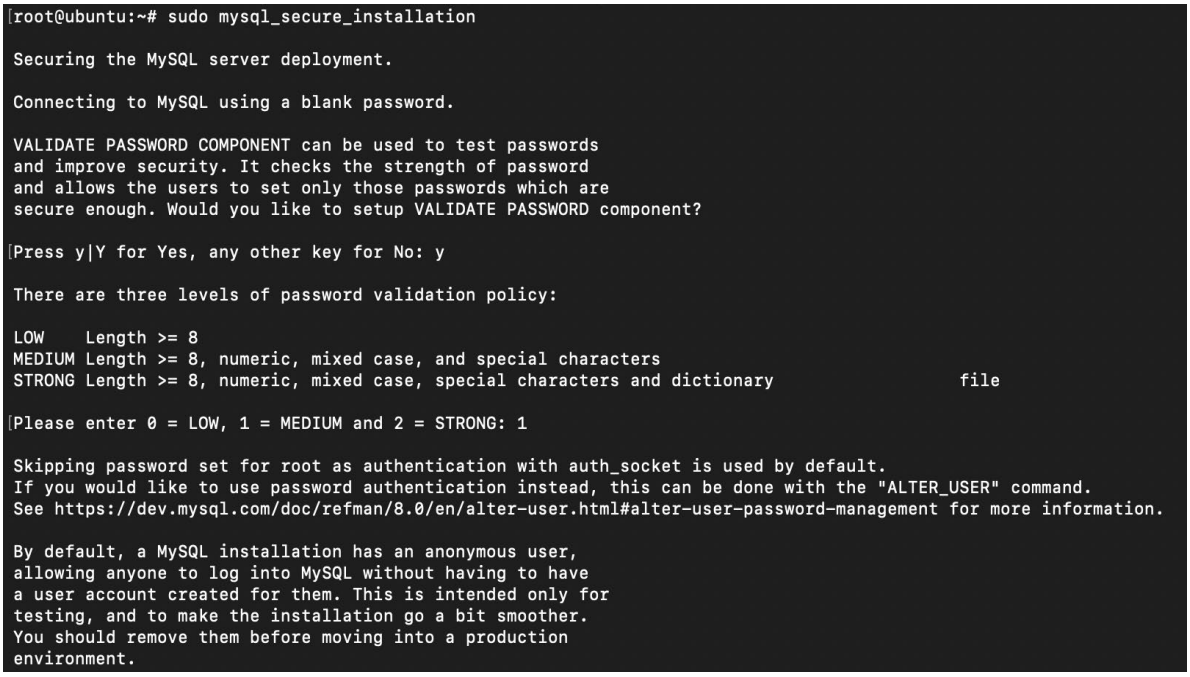


Рисунок 2.12 – Процесс конфігурації реляційної бази даних mySQL

```
Remove anonymous users? (Press y|Y for Yes, any other key for No) : y
Success.


Normally, root should only be allowed to connect from
'localhost'. This ensures that someone cannot guess at
the root password from the network.

Disallow root login remotely? (Press y|Y for Yes, any other key for No) : y
Success.

By default, MySQL comes with a database named 'test' that
anyone can access. This is also intended only for testing,
and should be removed before moving into a production
environment.


Remove test database and access to it? (Press y|Y for Yes, any other key for No) : y
 - Dropping test database...
Success.

 - Removing privileges on test database...
Success.

Reloading the privilege tables will ensure that all changes
made so far will take effect immediately.

Reload privilege tables now? (Press y|Y for Yes, any other key for No) : y
Success.

All done!
```

Рисунок 2.13 – Процес конфігурації реляційної бази даних mySQL

```
root@ubuntu:~# mysql -r
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.46-0ubuntu0.22.04.3 (Ubuntu)

Copyright (c) 2000, 2026, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create user euromebli@localhost identified by 'EuroMebli2026!';
Query OK, 0 rows affected (0.03 sec)

mysql> grant all privileges on *.* to euromebli@localhost with grant option;
Query OK, 0 rows affected (0.01 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)
```

Рисунок 2.14 – Створення першого адміністративного користувача для реляційної бази даних mysQL

Встановлення пакету PHP і всіх йому необхідних пакетів, виконується командою – apt install php-fpm php-mysql. Результат встановлення PHP пакетів зображено на рис. 2.15 – 2.16.

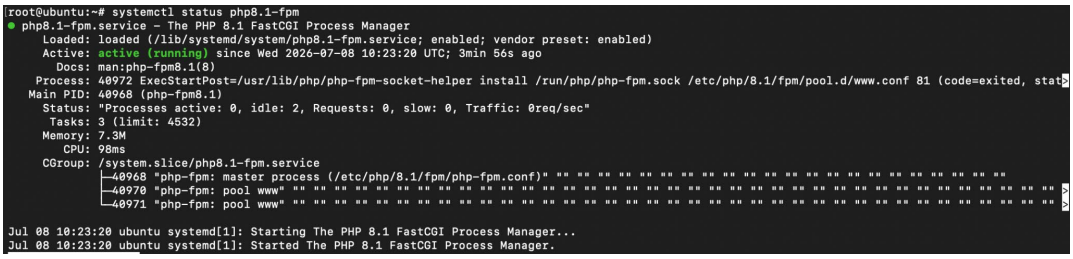


Рисунок 2.15 – Результат встановлення пакету PHP

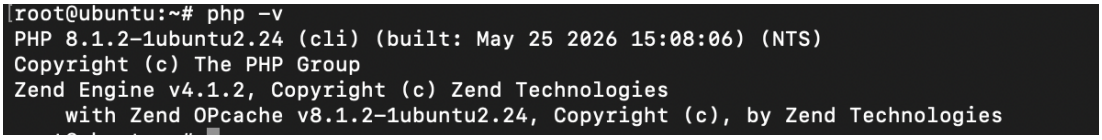


Рисунок 2.16 – Результат встановлення пакету PHP

2.3.3 Конфігурація веб-сервера Nginx для використання процесору PHP

Щоб розпочати конфігурацію веб-серверу для використання процессора PHP, необхідно створити робочу папку для веб-платформи в якій будуть розміщені файли веб-платформи підприємства «ЄвроМеблі+». Папка була створена командою – mkdir /var/www/euromebli. Також на папку було видано права за допомогою команди – chown -R www-data:www-data /var/www/euromebli. В результаті було отримано робочу папку для веб-платформи з необхідними правами (рис. 2.17).

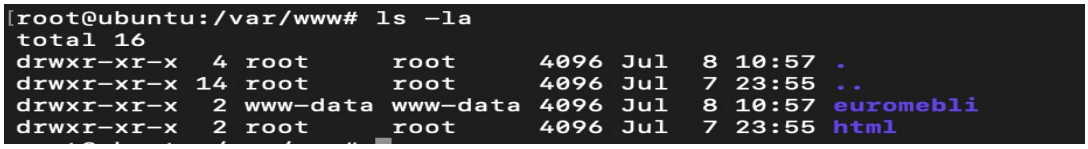


Рисунок 2.17 – Результат створення робочої папки і видання прав доступу на неї

Тепер необхідно створити конфіг у Nginx який буде використовувати процессор PHP для наших файлів розміщених на веб-платформи. Конфіг було создано командою – nano /etc/nginx/sites-available/euromebli.ua, це створило пустий бланк конфігу куди потім було імпортовано стандартний конфіг для веб-платформи (рис. 2.18).

```
server {
    listen 80;
    server_name euromebli.ua www.euromebli.ua;
    root /var/www/euromebli;

    index index.php index.html index.htm;

    location / {
        try_files $uri $uri/ =404;
    }

    location ~ \.php$ {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/run/php/php8.1-fpm.sock;
     }

}
```

Рисунок 2.18 – Створення конфігу Nginx для використання процесору PHP на веб-платформі

Після створення конфігу для Nginx, необхідно його активувати, щоб конфіг був активований необхідно ввести команду – ln -s /etc/nginx/sites-available/euromebli.ua /etc/nginx/sites-enabled/, і також після цього треба відключили конфіг який було встановлено за замовчуванням, це було зроблено командою – unlink /etc/nginx/sites-enabled/default.

Щоб з перезапуском Nginx не було проблем, було перевірено створену конфігурацію на синтаксичні помилки командою – nginx -t, і не побачивши помилок був перезапущений веб-сервер (рис. 2.19).

```
root@ubuntu:/etc/nginx/sites-enabled# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
root@ubuntu:/etc/nginx/sites-enabled# service nginx reload
root@ubuntu:/etc/nginx/sites-enabled#
```

Рисунок 2.19 – Перевірка синтаксичних помилок конфігурації і перезапуск Nginx

Щоб перевірити що все створено правильно, і процесор PHP працює, було створено тестову сторінку info.php, в якій був розміщений код який мав це перевірити, код і результати зображено на рис. 2.20 – 2.21.



Рисунок 2.20 – Код info.php

PHP Version 8.1.2-1ubuntu2.24

System	Linux ubuntu 5.15.0-151-generic #161-Ubuntu SMP Tue Jul 22 14:25:40 UTC 2025 x86_64
Build Date	May 25 2026 15:08:06
Build System	Linux
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.1/fpm
Loaded Configuration File	/etc/php/8.1/fpm/php.ini
Scan this dir for additional .ini files	/etc/php/8.1/fpm/conf.d
Additional .ini files parsed	/etc/php/8.1/fpm/conf.d/10-mysqlnd.ini, /etc/php/8.1/fpm/conf.d/10-opcache.ini, /etc/php/8.1/fpm/conf.d/10-pdo.ini, /etc/php/8.1/fpm/conf.d/20-calendar.ini, /etc/php/8.1/fpm/conf.d/20-ctype.ini, /etc/php/8.1/fpm/conf.d/20-exif.ini, /etc/php/8.1/fpm/conf.d/20-ffi.ini, /etc/php/8.1/fpm/conf.d/20-fileinfo.ini, /etc/php/8.1/fpm/conf.d/20-ftp.ini, /etc/php/8.1/fpm/conf.d/20-gettext.ini, /etc/php/8.1/fpm/conf.d/20-iconv.ini, /etc/php/8.1/fpm/conf.d/20-mysqli.ini, /etc/php/8.1/fpm/conf.d/20-pdo_mysql.ini, /etc/php/8.1/fpm/conf.d/20-phar.ini, /etc/php/8.1/fpm/conf.d/20-posix.ini, /etc/php/8.1/fpm/conf.d/20-readline.ini, /etc/php/8.1/fpm/conf.d/20-shmop.ini, /etc/php/8.1/fpm/conf.d/20-sockets.ini, /etc/php/8.1/fpm/conf.d/20-sysvmsg.ini, /etc/php/8.1/fpm/conf.d/20-sysvsem.ini, /etc/php/8.1/fpm/conf.d/20-sysvshm.ini, /etc/php/8.1/fpm/conf.d/20-tokenizer.ini
PHP API	20210902
PHP Extension	20210902
Zend Extension	420210902
Zend Extension Build	API420210902,NTS
PHP Extension Build	API20210902,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
DTrace Support	available, disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	zlib.*, string.rot13, string.toupper, string.tolower, convert.*, consumed, dechunk, convert.iconv.*

Рисунок 2.21 – Результат обробки коду info.php процесором PHP

2.3.4 Встановлення і конфігурування phpMyadmin

Щоб встановити пакету phpMyadmin виконується командою – apt install phpmyadmin. Під час встановлення пакету phpMyadmin, надається вибір який веб-сервер буде використовуватись (рис. 2.22), цей пункт було пропущено тому що в нашій роботі вже використовується Nginx.

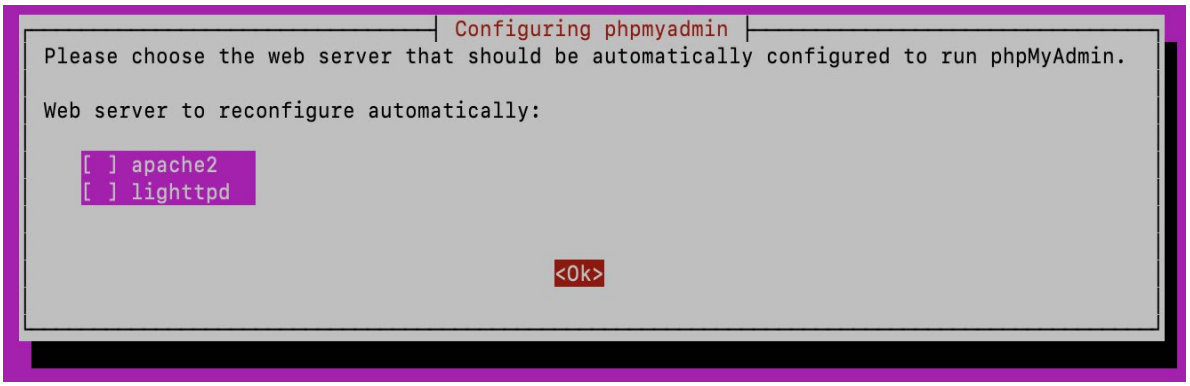


Рисунок 2.22 – Вибір веб-серверу під час установки phpMyadmin

Після установки пакету, phpMyadmin вважається встановленим, але ще немає до нього доступу через браузер. Щоб вирішити цю проблему, необхідно зв’язати папку phpMyadmin з кореневою директорією веб-платформи, це робиться за допомогою команди – ln -s /usr/share/phpmyadmin/ /var/www/euromebli/phpmyadmin, після цього у кореневій директорії нашого проєкту з’явиться папка phpmyadmin (рис. 2.23), і якщо перейти в браузері на публічну адресу веб-платформи і вказати “/phpmyadmin”, буде вітати сторінка авторизації у phpMyadmin (рис. 2.24).



Рисунок 2.23 – Папка phpMyadmin в кореневій папці проєкту

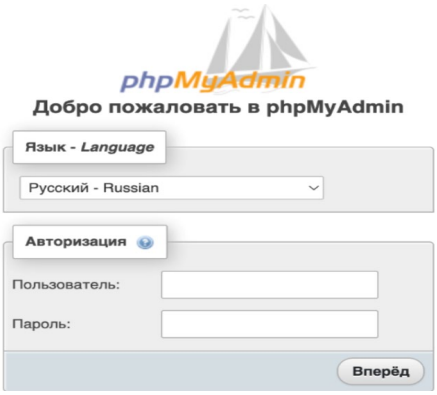


Рисунок 2.24 – Вітальна сторінка авторизації у phpMyadmin

2.3.5 Створення бази даних

Щоб створити базу даних для майбутнього веб-платформи, необхідно перейти за публічною адресою веб-платформи і вказати “/phpmyadmin”, там ми отримаємо вітальну сторінку авторизації у phpMyadmin (рис. 2.24), і ввести дані користувача який було створено до цього. Після успішної авторизації є головна сторінка phpMyadmin (рис. 2.25).

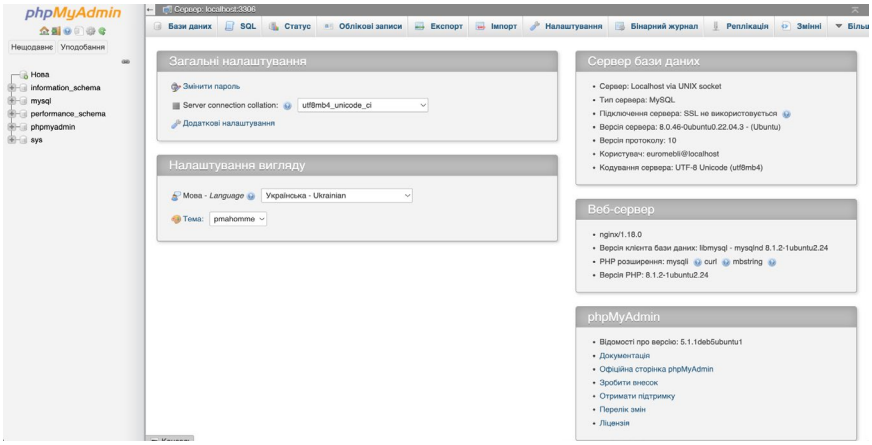


Рисунок 2.25 – Головна сторінка phpMyadmin

Далі після авторизації, необхідно в правій частині в колонках вибрати “Нова”, щоб створити базу даних. Процес створення бази даних зображено на рис. 2.26 – 2.27.

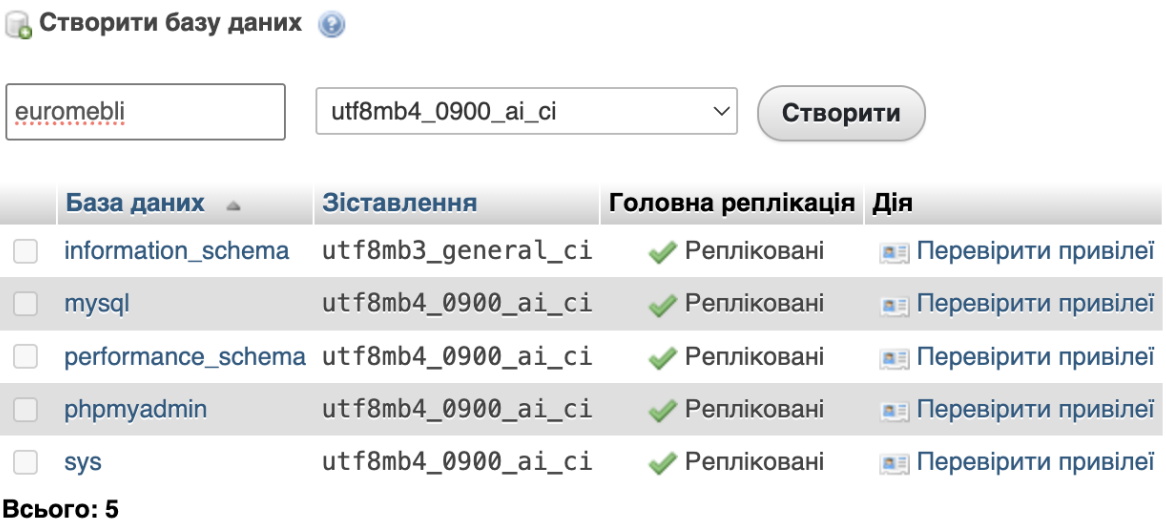


Рисунок 2.26 – Створення бази даних

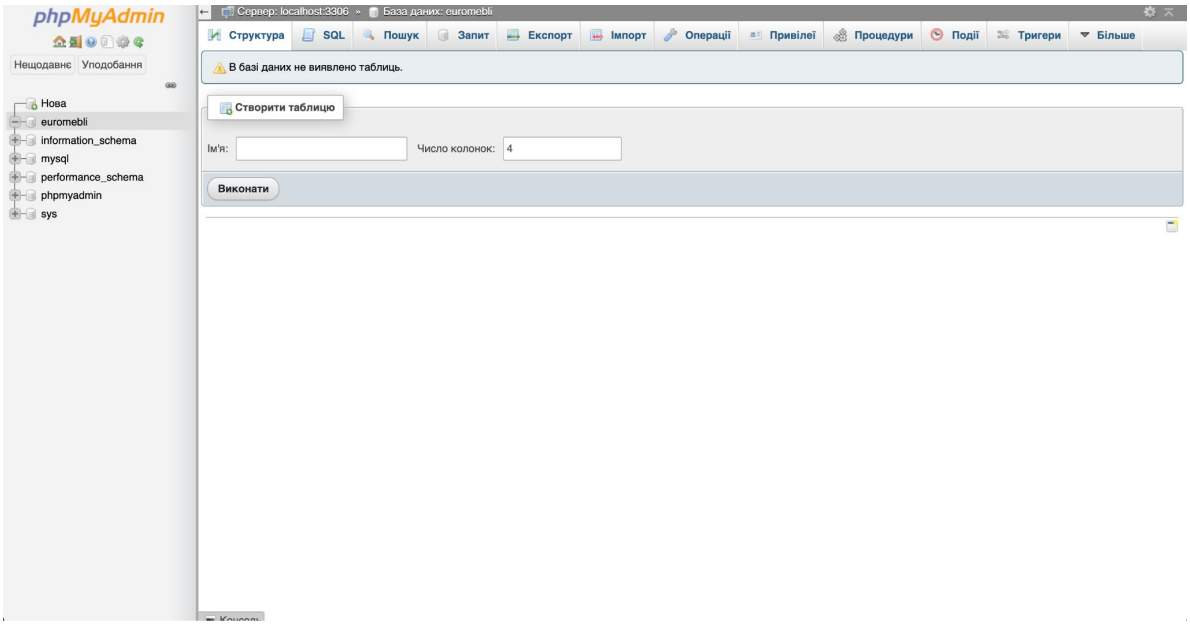


Рисунок 2.27 – Створення бази даних

Після створення необхідно створити необхідні таблиці, а саме:

– orders, яка буде зберігати усі створені замовлення користувачів, в якій буде 7 колонок з інформацією;

– news, яка буде зберігати новини які зображені на головній сторінці платформи, в якій буде 5 колонок з інформацією;

– furniture, яка буде зберігати усі можливі готові меблі, в якій буде 11 колонок з інформацією;

– contact_requests, яка буде зберігати всі хто відправив своє запитання у блоці зворотного зв’язку, в якій буде 4 колонки з інформацією;

Процес та результати створення цих таблиць зображено на рис. 2.28 – 2.31.

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
1	id	int			Ні	*Немає*			Змінити Знищити Більше
2	first_name	varchar(100)	utf8mb4_0900_ai_ci		Ні	*Немає*			Змінити Знищити Більше
3	last_name	varchar(100)	utf8mb4_0900_ai_ci		Ні	*Немає*			Змінити Знищити Більше
4	phone	varchar(30)	utf8mb4_0900_ai_ci		Ні	*Немає*			Змінити Знищити Більше
5	items_json	text	utf8mb4_0900_ai_ci		Ні	*Немає*			Змінити Знищити Більше
6	total	decimal(10,2)			Ні	*Немає*			Змінити Знищити Більше
7	created_at	timestamp			Так	CURRENT_TIMESTAMP		DEFAULT_GENERATED	Змінити Знищити Більше

Рисунок 2.28 – Структура таблиці “orders”

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
1	id	int			Ні	*Немає*			Змінити Знищити Більше
2	title	varchar(255)	utf8mb4_0900_ai_ci		Ні	*Немає*			Змінити Знищити Більше
3	body	text	utf8mb4_0900_ai_ci		Ні	*Немає*			Змінити Знищити Більше
4	image_url	varchar(300)	utf8mb4_0900_ai_ci		Так	*NULL*			Змінити Знищити Більше
5	created_at	timestamp			Так	CURRENT_TIMESTAMP		DEFAULT_GENERATED	Змінити Знищити Більше

Рисунок 2.29 – Структура таблиці “news”

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
1	id	int			Ні	*Немає*			Змінити Знищити Більше
2	name	varchar(150)	utf8mb4_0900_ai_ci		Ні	*Немає*			Змінити Знищити Більше
3	description	text	utf8mb4_0900_ai_ci		Так	*NULL*			Змінити Знищити Більше
4	price	decimal(10,2)			Ні	*Немає*			Змінити Знищити Більше
5	image_url	varchar(300)	utf8mb4_0900_ai_ci		Так	*NULL*			Змінити Знищити Більше
6	material	varchar(100)	utf8mb4_0900_ai_ci		Так	*NULL*			Змінити Знищити Більше
7	width_cm	int			Так	*NULL*			Змінити Знищити Більше
8	height_cm	int			Так	*NULL*			Змінити Знищити Більше
9	depth_cm	int			Так	*NULL*			Змінити Знищити Більше
10	weight_kg	decimal(6,2)			Так	*NULL*			Змінити Знищити Більше
11	manufacture	varchar(100)	utf8mb4_0900_ai_ci		Так	*NULL*			Змінити Знищити Більше
12	created_at	timestamp			Так	CURRENT_TIMESTAMP		DEFAULT_GENERATED	Змінити Знищити Більше

Рисунок 2.30 – Структура таблиці “furniture”

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
1	id	int			Ні	*Немає*			Змінити Знищити Більше
2	phone	varchar(30)	utf8mb4_0900_ai_ci		Ні	*Немає*			Змінити Знищити Більше
3	details	text	utf8mb4_0900_ai_ci		Так	*NULL*			Змінити Знищити Більше
4	created_at	timestamp			Так	CURRENT_TIMESTAMP		DEFAULT_GENERATED	Змінити Знищити Більше

Рисунок 2.31 – Структура таблиці “contact_requests”

2.4 Код веб-платформи

2.4.1 Архітектура робочої папки проєкту

Для реалізації коду веб-платформи необхідно в робочій папці створити необхідні файли і папки, а саме:

– папка admin, де створені файли admin_action.php, index.php,
– папка css, де створений файл style.css,
– index.php,
– header.php,
– footer.php,
– db.php,
– cart.php,

– cart_action.php,

– contact_action.php.

Таким чином, робоча папка виглядає наступним чином (рис. 2.32).

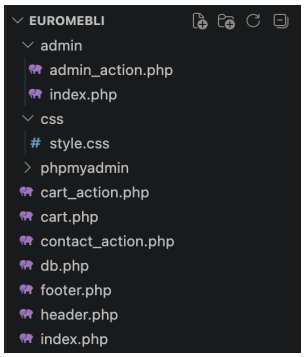


Рисунок 2.32 – Робоча папка проєкту

2.4.2 Код головної сторінки веб-платформи

Компонент “db.php” – один із найважливіших компонентів веб-платформи, який містить в собі підключення до реляційної бази даних mySQL з таким кодом (рис. 2.33). Підключення до бази даних виконується за допомогою PDO.

PDO – це розширення для підключення до бази даних, за допомогою якого можна зручно підключитись до бази, і цим самим використовуючи це розширення ми отримаємо захист від SQL-ін’єкцій який поставлений в задачі. Також є закриття сесії при неуспішному підключенні до бази даних, щоб гарантувати безпечну роботу веб-платформи. І функція getDB яка звертається до підключення к базі даних щоб в майбутньому в інших файлах веб-платформи можна було виконати безпечно SQL запрос який нам буде необхідний.



Рисунок 2.33 – Код компоненту “db.php”

Компонент “header.php” – містить в собі наступний код (рис. 2.34). Шапка веб-сайту (header) – це найвища частина веб-сторінки, яка розміщена перед основною частиною веб-сайту (тілом). В шапці зазвичай розміщують клікабельний логотип який переадресовує на головну сторінку сайту, меню навігації, основні контактні дані (адреса, клікабельний телефон та електронну пошту), та інші важливі елементи [19].

В випадку підприємства «ЄвроМеблі+», в шапці веб-платформи розміщено текстовий логотип з фірмовими кольорами, навігаційне меню веб-платформою, перемикач мови платформи між українською (UA) та англійською (EN), і динамічний значок кошику з покупками. Нижче у коді шапки є меню-гамбургер яке реалізовано для адаптивного веб-дизайну, оскільки метою проєкту є адаптація інтерфейсу під будь-який пристрій користувача.

header.php

```
<header>
    <nav>
        <a href="#home" class="nav-logo">Євро<span>Меблі</span>+</a>

        <ul class="nav-links" id="nav-links">
            <li><a href="#home">Головна</a></li>
            <li><a href="#furniture">Меблі</a></li>
            <li><a href="#news">Новини</a></li>
            <li><a href="#contact-section">Зв'язатись</a></li>
        </ul>

        <div class="lang-switcher">
            <a href="/" class="active">UA</a>
            <span>|</span>
            <a href="/en">EN</a>
        </div>

        <a href="cart.php" class="nav-cart">
            <svg xmlns="http://www.w3.org/2000/svg" width="22" height="22" viewBox="0 0 24 24"
                fill="none" stroke="currentColor" stroke-width="2" stroke-linecap="round" stroke-linejoin="round">
                <circle cx="9" cy="21" r="1"/><circle cx="20" cy="21" r="1"/>
                <path d="M1 1h4l2.68 13.39a2 2 0 0 0 2 1.61h9.72a2 2 0 0 0 2-1.61L23 6H6"/>
            </svg>

            <?php if ($cartCount > 0): ?>
                <span class="cart-badge" id="cart-badge"><?php echo $cartCount; ?></span>
            <?php else: ?>
                <span class="cart-badge cart-badge--hidden" id="cart-badge">0</span>
            <?php endif; ?>
        </a>

        <button class="nav-hamburger" id="hamburger">
            <span></span><span></span><span></span>
        </button>
    </nav>
</header>
```

Рисунок 2.34 – Код компоненту “header.php”

Код компоненту “footer.php” – має в собі код зображений на рис. 2.35. Футер (footer) – це фінальний контентний блок на сайті, який найчастіше містить

допоміжну інформацію про компанію: контакти, адресу, соц. мережі, корисні посилання тощо. Підвал сайту є протилежністю хедера і так само широко використовуваний для всіх сторінок. Здебільшого мета футера – продублювати важливу інформацію для користувача, цей блок не любить перевантаження, тому інформацію розміщену має бути корисною і коротко викладеною [20].

Так що, у нижньому колонтитулі (компонент “footer.php”), реалізовано відображення текстового логотипу з фірмовим кольором та слоганом підприємства, і авторськими правами.

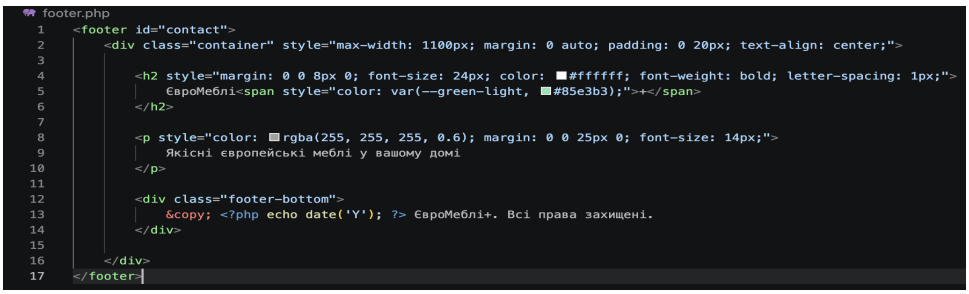


Рисунок 2.35 – Код компоненту “footer.php”

Так як базові компоненти для розробки головної сторінки готові, тепер настав час розробки головної сторінки. На головній сторінці яка є “index.php” (перша сторінка яка відкривається користувачу як він відвідує веб-платформу), реалізовано запуск сесій клієнтів, для збереження кількості товарів у кошику і оформленню замовлення (рис. 2.36).

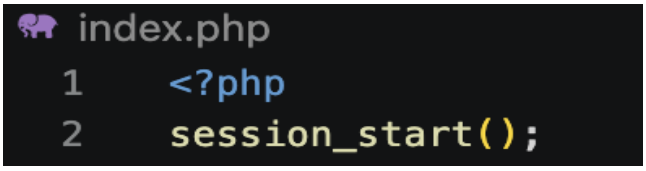


Рисунок 2.36 – Запуск сесії клієнта

Імпорт таких компонентів як “header.php”, “footer.php” в головну сторінку зроблено таким чином (рис. 2.37 – 2.38). Хедер було імпортовано перед тілом сторінки, футер перед кінцем основної частини веб-платформи.

```
48    <!-- Імпорт хеадеру -->
49    <?php include 'header.php'; ?>
```

Рисунок 2.37 – Імпорт компоненту “header.php” в головну сторінку платформи

```
215   <!-- Імпорт футеру -->
216   <?php include 'footer.php'; ?>
```

Рисунок 2.38 – Імпорт компоненту “footer.php” в головну сторінку платформи

Підключення головної сторінки до бази даних реалізовано імпортом компоненту “db.php” (рис. 2.39).

```
3     require_once 'db.php';
```

Рисунок 2.39 – Імпорт компоненту “db.php”

Далі реалізовано секційну структуру платформи (наприклад, секція домашньої сторінки, меблів, новин тощо).

Першу секцію, з домашньою сторінкою реалізовано наступним чином (рис. 2.40).

```
<section id="home">
    <div class="container">
        <div class="hero-content">
            <p class="hero-eyebrow">Європейська якість · Дизайн майбутнього</p>
            <h1>Меблі, які відчуваються<br><em> як вдома</em></h1>
            <p>
                Відкрийте для себе нашу ретельно відібрану колекцію виробів,
                виготовлених з високоякісних матеріалів перевірених часом.
            </p>
            <div class="hero-buttons">
                <a href="#furniture" class="btn btn-outline">Переглянути меблі</a>
            </div>
        </div>
    </div>
</section>
```

Рисунок 2.40 – Код першої секції веб-платформи

Другу секцію з готовими меблями реалізовано кодом, фрагмент якого зображено на рис. 2.41. У коді забезпечено використання даних, що були заздалегідь отримані з бази даних підприємства. Код із запитом на необхідну інформації за бази показано на рис. 2.42. Відображення отриманої інформації виконується із застосуванням функції htmlspecialchars(), яка протидіє XSS-атакам, блокуючи можливість підставляти значення шкідливих значень, що можуть бути використані для компрометації системи .

Окрім цього, після натискання кнопки «Інфо» здійснюється запит до бази даних та звернення до скрипта, зображеного на рис. 2.43. Цей код записує отримані параметри у змінні та викликає модальне вікно (попап) з детальною інформацією про обрану готову мебель. Код цього модального вікна представлено на рис. 2.44. Логіку роботи функції закриття попапу наведено на рис. 2.45. Алгоритм роботи функції закриття інтерфейсу як за натисканням на відповідну кнопку, так і в разі кліку на область фона поза межами вікна.

Після натискання кнопки «Купити», відбувається звернення до JavaScript-коду, проілюстрованого на рис. 2.46. Цей скрипт надсилає POST-запит до серверного компоненту “cart_action.php”, із параметром дії “add”. Фрагмент реалізацією додавання товару представлено на рис. 2.47. Для динамічного відображення кількості доданих позицій на іконці кошика в реальному часі використовується лічильник, для коректної роботи якого було реалізовано функцію, наведену на рис. 2.48.

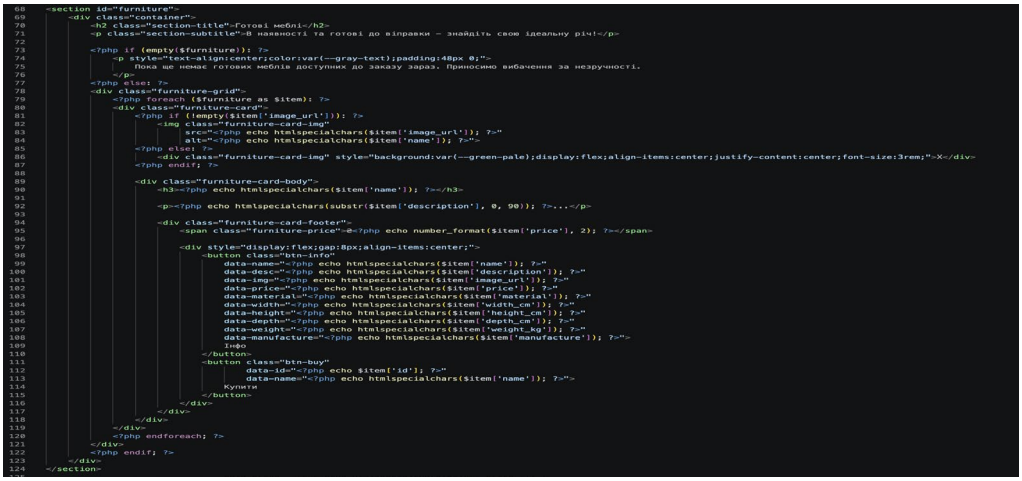


Рисунок 2.41 – Код другої секції веб-платформи

```php
$furniture = [];
try {
    $query_furniture = getDB()->query('SELECT * FROM furniture ORDER BY created_at DESC');
    $furniture = $query_furniture->fetchAll();
} catch (PDOException $e) {
    die("Помилка при отриманні меблів: " . $e->getMessage());
}
```

Рисунок 2.42 – Отримання інформації про меблі з бази даних

```javascript
// --- Попап з інформацією по меблям
var overlay = document.getElementById('popup-overlay');
var infoButtons = document.querySelectorAll('.btn-info');

for (var i = 0; i < infoButtons.length; i++) {
    infoButtons[i].addEventListener('click', function() {
        // отримання кнопки яку натиснули
        var btn = this;

        document.getElementById('popup-name').textContent = btn.getAttribute('data-name');
        document.getElementById('popup-desc').textContent = btn.getAttribute('data-desc');
        document.getElementById('popup-img').src = btn.getAttribute('data-img');
        document.getElementById('popup-material').textContent = btn.getAttribute('data-material');

        var price = btn.getAttribute('data-price');
        document.getElementById('popup-price').textContent = '₴' + price;

        document.getElementById('popup-width').textContent = btn.getAttribute('data-width') + ' cm';
        document.getElementById('popup-height').textContent = btn.getAttribute('data-height') + ' cm';
        document.getElementById('popup-depth').textContent = btn.getAttribute('data-depth') + ' cm';
        document.getElementById('popup-weight').textContent = btn.getAttribute('data-weight') + ' kg';
        document.getElementById('popup-manufacture').textContent = btn.getAttribute('data-manufacture');

        overlay.classList.add('active');
    });
}
```

Рисунок 2.43 – Записування отриманих даних з бази даних до змінних та виклик попапу з інформацією про меблі

```html
<div class="popup-overlay" id="popup-overlay">
    <div class="popup-box">
        <div class="popup-header">
            <h2 id="popup-name"></h2>
            <button class="popup-close" id="popup-close">&times;</button>
        </div>
        <div class="popup-body">
            <img id="popup-img" class="popup-img" src="" alt="">
            <div class="popup-details">
                <p id="popup-desc"></p>
                <div class="spec-grid">
                    <div class="spec-item">
                        <span class="spec-label">Ціна</span>
                        <span class="spec-value" id="popup-price"></span>
                    </div>
                    <div class="spec-item">
                        <span class="spec-label">Матеріал</span>
                        <span class="spec-value" id="popup-material"></span>
                    </div>
                    <div class="spec-item">
                        <span class="spec-label">Ширина</span>
                        <span class="spec-value" id="popup-width"></span>
                    </div>
                    <div class="spec-item">
                        <span class="spec-label">Висота</span>
                        <span class="spec-value" id="popup-height"></span>
                    </div>
                    <div class="spec-item">
                        <span class="spec-label">Глибина</span>
                        <span class="spec-value" id="popup-depth"></span>
                    </div>
                    <div class="spec-item">
                        <span class="spec-label">Вага</span>
                        <span class="spec-value" id="popup-weight"></span>
                    </div>
                    <div class="spec-item">
                        <span class="spec-label">Виробник</span>
                        <span class="spec-value" id="popup-manufacture"></span>
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>
```

Рисунок 2.44 – Код попапу з інформацією про мебель

```
    // Функція закриття попапу
    function closePopup() {
        overlay.classList.remove('active');
    }

    if (document.getElementById('popup-close')) {
        document.getElementById('popup-close').addEventListener('click', closePopup);
    }

    // Закриття по кліку на фон
    overlay.addEventListener('click', function(e) {
        if (e.target === overlay) {
            closePopup();
        }
    });
```

Рисунок 2.45 – Функції закриття попапу

```
    // --- Кнопка Придбати (Корзина) ---
    var badge = document.getElementById('cart-badge');
    var miniToast = document.getElementById('mini-toast');
    var buyButtons = document.querySelectorAll('.btn-buy');

    for (var i = 0; i < buyButtons.length; i++) {
        buyButtons[i].addEventListener('click', function() {
            var btn = this;
            var id = btn.getAttribute('data-id');
            var name = btn.getAttribute('data-name');

            fetch('cart_action.php', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/x-www-form-urlencoded'
                },
                body: 'action=add&id=' + id
            })
            .then(function(response) {
                return response.json();
            })
            .then(function(data) {
                if (data.ok == true) {
                    // Обнофлення цифри на корзині
                    badge.textContent = data.count;
                    badge.classList.remove('cart-badge--hidden');

                    // Показуємо повідомлення
                    miniToast.textContent = name + ' додано до кошика';
                    miniToast.classList.add('mini-toast--show');

                    // Функція прятання увідомлення 3 секунди
                    setTimeout(function() {
                        miniToast.classList.remove('mini-toast--show');
                    }, 3000);
                }
            });
        });
    }
```

Рисунок 2.46 – Код кнопки «Купити»

```php
case 'add':
    $id = isset($_POST['id']) ? (int)$_POST['id'] : 0;
    if ($id <= 0) {
        echo json_encode(['ok' => false]);
        exit;
    }

    // Якщо товару ще немає у корзині, достаємо його з бази
    if (!isset($_SESSION['cart'][$id])) {
        try {
            $stmt = getDB()->prepare('SELECT id, name, price, image_url FROM furniture WHERE id = ?');
            $stmt->execute([$id]);
            $item = $stmt->fetch();

            if (!$item) {
                echo json_encode(['ok' => false]);
                exit;
            }

            // Зберігаємо дані про товар у сессії
            $_SESSION['cart'][$id] = [
                'id' => $item['id'],
                'name' => $item['name'],
                'price' => $item['price'],
                'image_url' => $item['image_url'],
                'qty' => 0
            ];
        } catch (PDOException $e) {
            echo json_encode(['ok' => false]);
            exit;
        }
    }

    // Додаємо кількість
    $_SESSION['cart'][$id]['qty']++;
    break;
```

Рисунок 2.47 – Код реалізації функції “add”

```php
$cartCount = 0;
if (isset($_SESSION['cart']) && is_array($_SESSION['cart'])) {
    foreach ($_SESSION['cart'] as $item) {
        if (isset($item['qty'])) {
            $cartCount += $item['qty'];
        }
    }
}
```

Рисунок 2.48 – Функція для відображення кількості вибраних товарів в іконці кошика

Третю секцію з останніми новинами реалізовано за допомогою програмного коду, фрагмент якого наведено на рис. 2.49. У цьому блоці також забезпечено використання даних, що попередньо отримані з бази даних із застосуванням функції “htmlspecialchars()”. В секції з новинами так же

реалізовано використання даних отриманих заздалегідь, з функцією “htmlspecialchars()”. Код із запитом необхідних даних з таблиці зображено на рис. 2.50.

```
<section id="news">
    <div class="container">
        <h2 class="section-title">Останні новини</h2>
        <p class="section-subtitle">Залишайся в курсі усіх новин підприємства ЄвроМеблі+</p>

        <div class="news-grid">
            <?php if (empty($news)): ?>
                <div class="news-empty">
                    <p>Ще немає новин, якими ми могли би з вами поділитись.</p>
                </div>
            <?php else: ?>
                <?php foreach ($news as $post): ?>
                <div class="news-card">
                    <?php if (!empty($post['image_url'])): ?>
                        <img class="news-card-img"
                             src="<?php echo htmlspecialchars($post['image_url']); ?>"
                             alt="<?php echo htmlspecialchars($post['title']); ?>">
                    <?php else: ?>
                        <div class="news-card-img-placeholder">X</div>
                    <?php endif; ?>

                    <div class="news-card-body">
                        <span class="news-date"><?php echo date('d.m.Y', strtotime($post['created_at'])); ?></span>
                        <h3><?php echo htmlspecialchars($post['title']); ?></h3>

                        <p>
                            <?php
                                $text = $post['body'];
                                if (mb_strlen($text) > 120) {
                                    $text = mb_substr($text, 0, 120) . '...';
                                }
                                echo htmlspecialchars($text);
                            ?>
                        </p>
                    </div>
                </div>
                <?php endforeach; ?>
            <?php endif; ?>
        </div>
    </div>
</section>
```

Рисунок 2.49 – Код третьої секції веб-платформи

```
13  $news = [];
14  try {
15      $query_news = getDB()->query('SELECT * FROM news ORDER BY created_at DESC LIMIT 6');
16      $news = $query_news->fetchAll();
17  } catch (PDOException $e) {
18      die("Помилка при отриманні новин: " . $e->getMessage());
19  }
```

Рисунок 2.50 – Отримання інформації про останні новини з бази даних

Четверта секцію зі зворотнім зв’язком реалізовано за допомогою коду, фрагмент якого зображено на рис. 2.51. Після заповнення форми та натискання кнопки «Зв’язатись» система звертається до серверного компонента “contact_action.php” (рис. 2.52).

Код цього компоненту перевіряє коректність введеного номера телефону користувача. Якщо номер телефону вказано невірно, сервер повертає значення “error” для змінної “contact_status”. Через реалізовану на платформі систему прапорців (флагів) змінна “contactError” переходить у стан “true”, внаслідок чого скрипт виводить повідомлення про помилку, інформуючи клієнта про некоректне введення даних.

У разі правильного введення номеру телефону за аналогічною системою відбувається обробка запиту файлом “contact_action.php”. На основі отриманого статусу за допомогою прапорців змінна “contactOk” набуває значення “true”. Після цього на екрані користувача відображається повідомлення про успішне надсилання форми, а введена інформація зберігається в базі даних.

Окрім цього, незалежно від отриманого значення змінної “contact_status”, виконується код, наведений на рис. 2.53. Він автоматично прокручує сторінку до форми зворотнього зв’язку. Цей функціонал є необхідним, оскільки після надсилання даних з форми сторінка перезавантажується, і без автоматичного скролінгу користувач опинявся б у верхній частині домашньої сторінки.

```
<section id="contact-section" class="contact-section">
    <div class="container">
        <div class="contact-inner">

            <div class="contact-copy">
                <p class="contact-eyebrow">Ми тут, щоб допомогти</p>
                <h2>Хочете запитати <br>про інші меблі?</h2>
                <p class="contact-sub">
                    Залиште свій номер та коротке повідомлення нашій команді
                    і наша команда як найшвидше допоможе вам.
                </p>
            </div>

            <div class="contact-form-wrap">

                <?php if ($contactOk): ?>
                    <div class="contact-success-msg" id="contact-success">
                        <div class="contact-success-icon">✓</div>
                        <strong>Повідомлення отримано!</strong>
                        <span>Ми зв'яжемося з вами найближчим часом.</span>
                    </div>
                <?php elseif ($contactError): ?>
                    <div class="contact-error-msg">
                        Будь ласка, введіть дійсний номер телефону та спробуйте ще раз.
                    </div>
                <?php endif; ?>

                <form action="contact_action.php" method="post" class="contact-form" id="contact-form">
                    <div class="contact-form-group">
                        <label for="c-phone">Номер телефону *</label>
                        <input type="tel" id="c-phone" name="phone"
                               placeholder="+380 00 000 0000" required>
                    </div>
                    <div class="contact-form-group">
                        <label for="c-details">Деталі (не обов'язково)</label>
                        <textarea id="c-details" name="details"
                                  placeholder="Наприклад, я шукаю диван на замовлення синього кольору..."></textarea>
                    </div>
                    <button type="submit" class="btn-contact">Зв'язатись</button>
                </form>

            </div>
        </div>
    </div>
</section>
```

Рисунок 2.51 – Код четвертої секції веб-платформи

contact_action.php

```php
<?php
session_start();
require_once 'db.php';

if (isset($_POST['phone'])) {
    $phone = $_POST['phone'];
    $details = $_POST['details'];

    if ($phone == '') {
        header('Location: index.php?contact_status=error#contact-section');
        exit;
    }

    try {
        $stmt = getDB()->prepare('INSERT INTO contact_requests (phone, details) VALUES (?, ?)');
        $stmt->execute([$phone, $details]);

        header('Location: index.php?contact_status=ok#contact-section');
        exit;

    } catch (PDOException $e) {
        die("Помилка збереження заявки: " . $e->getMessage());
    }

} else {
    header('Location: index.php');
    exit;
}
?>
```

Рисунок 2.52 – Код компоненту “contact_action.php”

```
        if (window.location.search.indexOf('contact_status') !== -1) {
            var contactSec = document.getElementById('contact-section');
            if (contactSec) {
                contactSec.scrollIntoView();
            }
        }
    });
```

Рисунок 2.53 – Код прокручування сторінки до форми зворотнього зв’язку

2.4.3 Код сторінки з кошиком

Програмний код сторінки кошика “cart.php”, представлено на рис. 2.54 – 2.55. Наведений фрагмент коду описує структуру картки замовлення, яка

відповідає за виведення назв обраних товарів, їхньої кількості та підсумкової вартості.



Рисунок 2.54 – Код сторінки з кошиком “cart.php”



Рисунок 2.55 – Код сторінки з кошиком “cart.php”

Для забезпечення коректної роботи з іншими частинами веб-платформи та синхронізацією даних, на сторінці налаштовано підключення до бази даних (аналогічно до головної сторінки) та ініційовано старт користувацької сесії. Логіку передачі інформації про обрані товари з головної сторінки до кошика представлено на рис. 2.56.

Динамічне виведення кількості позицій у полі «Товари у кошику ()» забезпечує скрипт, фрагмент якого показано на рис. 2.57, а розрахунок і відображення загальної суми замовлення реалізовано за допомогою коду зображеному на рис. 2.58.

Керування вмістом кошика (збільшення і зменшення кількості конкретного товару, його видалення чи повного очищення списку) здійснюється за допомогою відповідних кнопок які передають серверній частині три параметри: “add”, “remove”, та “clear”. Логіку роботи та реалізацію параметру “add” (додавання товару) за допомогою коду, було детально розглянуто у підрозділі 2.4.2 «Код головної сторінки веб-платформи» під час опису другого сектору головної сторінки, тому на цій сторінці вона використовується повторно без змін.

Натомість параметри “remove”, та “clear” (зменшення кількості товарів, та повне очищення списку товарів) було за допомогою того ж самого компоненту “cart_action.php”, і фрагменти коду цих параметрів наведено на рис. 2.59.

Алгоритм дії “remove” спочатку зменшує значення кількості товару (qty) у сесії на одиницю. Якщо після цього кількість стає рівною чи меншою нулю, система автоматично викликає функцію яка повністю видаляє ідентифікатор цього товару з кошику.

Параметр “clear” працює лінійно – він миттєво обнуляє поточний кошик, видаляючи всі ідентифікатори товарів, що призводить до очищення кошику. Але перед тим як система звернеться до серверного компоненту, кнопка очищення викликає клієнтську функцію “clearCart()” (рис. 2.60). Після якої йде підтвердження дії, а вже далі очищення кошику.

```
5    $cart = isset($_SESSION['cart']) ? $_SESSION['cart'] : [];
```

Рисунок 2.56 – Функція передачі даних про вибрані продукти користувача з основної частини веб-сайту

```php
$totalQty = 0;
foreach ($cart as $item) {
    $totalQty += $item['qty'];
}
```

Рисунок 2.57 – Код для підрахунку кількості товарів

```php
$total = 0;
foreach ($cart as $item) {
    $total += $item['price'] * $item['qty'];
}
```

Рисунок 2.58 – Код для підрахунку суми замовлення

```php
    case 'remove':
        $id = isset($_POST['id']) ? (int)$_POST['id'] : 0;
        if (isset($_SESSION['cart'][$id])) {
            $_SESSION['cart'][$id]['qty']--;
            // Якщо кількість стала 0 чи менше, то чистимо з корзину
            if ($_SESSION['cart'][$id]['qty'] <= 0) {
                unset($_SESSION['cart'][$id]);
            }
        }
        break;

    case 'clear':
        // Повністю чистимо корзину
        $_SESSION['cart'] = [];
        break;

    default:
        echo json_encode(['ok' => false]);
        exit;
}
```

Рисунок 2.59 – Операції “remove” і “clear”

```javascript
function clearCart() {
    if (!confirm('Очистити кошик?')) return;
    fetch('cart_action.php', {
        method: 'POST',
        headers: { 'Content-Type': 'application/x-www-form-urlencoded' },
        body: 'action=clear'
    }).then(function() {
        window.location.reload();
    });
}
```

Рисунок 2.60 – Функція “clearCart()”

Завершенням інтерфейсу кошика є форма оформлення замовлення та його підтвердження. Форма містить в собі динамічне інформаційне вікно чека, яке за допомогою циклу перебирає масив сесії та виводить перелік обраних меблів у форматі “Назва x Кількість”, розраховуючи вартість для кожної позиції та всього замовлення загалом.

Поля для збору інформації (ім’я, прізвище та номер телефону) захищені функцією “htmlspecialchars()”, призначення якої було детально описано у попередньому розділі.

Після кліку на кнопку “Підтвердити замовлення”, виконується POST-запит, який звертається обробника на цій же сторінці. Код що відповідає за прийом та аналіз цього POST-запиту, зображено на рис. 2.61.

```php
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['place_order'])) {
    $firstName = trim($_POST['first_name']);
    $lastName  = trim($_POST['last_name']);
    $phone     = trim($_POST['phone']);
    $errors    = [];

    if ($firstName == '') { $errors[] = "Ім'я обов'язкове для заповнення."; }
    if ($lastName == '') { $errors[] = "Прізвище обов'язкове для заповнення."; }
    if ($phone == '') { $errors[] = "Номер телефону обов'язковий."; }
    if (empty($cart)) { $errors[] = "Ваш кошик порожній."; }

    if (empty($errors)) {
        $total = 0;
        foreach ($cart as $item) {
            $total += $item['price'] * $item['qty'];
        }

        try {
            $stmt = getDB()->prepare(
                'INSERT INTO orders (first_name, last_name, phone, items_json, total) VALUES (?, ?, ?, ?, ?)'
            );
            $stmt->execute([$firstName, $lastName, $phone, json_encode(array_values($cart)), $total]);

            $_SESSION['cart'] = [];
            header('Location: index.php?status=ordered');
            exit;
        } catch (PDOException $e) {
            $errors[] = 'Не вдалося зберегти замовлення: ' . $e->getMessage();
        }
    }
}
```

Рисунок 2.61 – Код обробки підтвердження замовлення

2.4.4 Код сторінки з адміністративною панеллю

Доступ до адміністративної сторінки веб-платформою здійснюється за маршрутом “/admin”. Роботу цієї сторінки забезпечують два функціональні файли: “index.php” та “admin_action.php”.

Під час переходу до розділу “/admin” базовим файлом сторінки є “index.php”. У ньому реалізована система користувацьких сесій та підключення до бази даних для подальшої взаємодії з її даними, робота і архітектура цих систем була описана у попередніх розділах.

Перед тим як користувач отримує доступ до панелі керування веб-платформою, система вимагає проходження системи аутентифікації адміністратора за паролем. Код реалізації функції аутентифікації адміністратора зображено на рис. 2.62. Форма авторизації має в собі поле з вводом паролю, і після натискання кнопки «Увійти» виконується POST-запит який надсилається на серверний обробник сторінки, логіку роботи якого представлено рис. 2.63.

Якщо пароль вказано правильно, обробник переводить зміну сесії “admin” у стан “true”, внаслідок чого внутрішний параметр “loggedIn” змінює стан з “false” на “true”, відкриваючи доступ до головного інтерфейсу адмін-панелі. У разі введення некоректного пароля, серверний обробник повертає повідомлення про помилку. Виведення повідомлення про помилку в обов’язковому порядку використовує функцію “htmlspecialchars()” для запобігання XSS-атакам.

```
181  <?php if (!$loggedIn): ?>
182  <div class="login-wrap">
183      <div class="login-card">
184          <h1>ЄвроМеблі+ Вхід</h1>
185          <p>Введіть пароль адміністратора для входу.</p>
186
187          <?php if ($error != ''): ?>
188              <div class="error-msg"><?php echo htmlspecialchars($error); ?></div>
189          <?php endif; ?>
190
191          <form method="post">
192              <div class="form-group">
193                  <label for="password">Пароль</label>
194                  <input type="password" id="password" name="password" required>
195              </div>
196              <button type="submit" class="btn btn-green">Увійти</button>
197          </form>
198
199          <p style="text-align:center; margin-top:20px; font-size:13px;">
200              <a href="../index.php" style="color:var(--green-main)"> На головну</a>
201          </p>
202      </div>
203  </div>
```

Рисунок 2.62 – Авторизація в адміністративній панелі

```php
if (isset($_POST['password'])) {
    if ($_POST['password'] === ADMIN_PASSWORD) {
        $_SESSION['admin'] = true;
    } else {
        $error = 'Неправильний пароль. Спробуйте ще раз.';
    }
}
```

Рисунок 2.63 – Обробник авторизації в адміністративну панель

Після того як значення параметра “loggedIn” змінюється на “true”, система відправляє запит на отримання даних з бази даних про наявність меблів, останніх новин, замовлень, і звернення з форми зворотнього зв’язку (рис. 2.64).

```php
$furniture = array();
$newsItems = array();
$orders = array();
$contactRequests = array();

if ($loggedIn) {
    try {
        $db = getDB();
        $furniture       = $db->query('SELECT * FROM furniture ORDER BY created_at DESC')->fetchAll();
        $newsItems       = $db->query('SELECT * FROM news ORDER BY created_at DESC')->fetchAll();
        $orders          = $db->query('SELECT * FROM orders ORDER BY created_at DESC')->fetchAll();
        $contactRequests = $db->query('SELECT * FROM contact_requests ORDER BY created_at DESC')->fetchAll();
    } catch (PDOException $e) {
        $flash = 'Помилка БД: ' . htmlspecialchars($e->getMessage());
    }
}
```

Рисунок 2.64 – Отримання інформації з бази даних

На головній сторінці панелі адміністратора реалізовано верхню панель (header), у якій передбачено можливість виході з адмін-панелі або повернення на основну сторінку платформи (рис. 2.65).

У центральній частині панелі (тілі) розміщено блок навігації між чотирма вкладками: «Меблі», «Новини», «Замовлення» та «Заявки». Код цього навігаційного меню наведено на рис. 2.66. На кнопках вкладок «Замовлення» та «Заявки» розроблено динамічне виведення кількості активних записів, сформованих на основі отриманої інформації за бази даних, за допомогою підрахунку елементів масиву.

Для забезпечення динамічного перемикання між вкладками без перезавантаження сторінки розроблено скрипт із використанням мови програмування JavaScript (рис. 2.67)

```
205    <?php else: ?>
206    <header class="admin-header">
207        <h1>Панель адміністратора — ЄвроМеблі+</h1>
208        <div style="display:flex; align-items:center; gap:20px;">
209            <a href="../index.php"> На сайт</a>
210            <form method="post" class="logout-form">
211                <button type="submit" name="logout" value="1">Вийти</button>
212            </form>
213        </div>
214    </header>
```

Рисунок 2.65 – Блок “Header” адміністративної панелі

```
222        <div class="tabs">
223            <button class="tab-btn active" data-tab="furniture">Меблі</button>
224            <button class="tab-btn" data-tab="news">Новини</button>
225            <button class="tab-btn" data-tab="orders">Замовлення (<?php echo count($orders); ?>)</button>
226            <button class="tab-btn" data-tab="contacts">Заявки (<?php echo count($contactRequests); ?>)</button>
227        </div>
```

Рисунок 2.66 – Навігація між сторінками на адміністративній панелі

```
475    var buttons = document.querySelectorAll('.tab-btn');
476    buttons.forEach(function(btn) {
477        btn.addEventListener('click', function() {
478            document.querySelectorAll('.tab-btn').forEach(function(b) { b.classList.remove('active'); });
479            document.querySelectorAll('.tab-panel').forEach(function(p) { p.classList.remove('active'); });
480
481            btn.classList.add('active');
482            var currentTab = btn.getAttribute('data-tab');
483            document.getElementById('tab-' + currentTab).classList.add('active');
484        });
485    });
```

Рисунок 2.67 – Скрипт динамічного переключення вкладок

Першою розділом панелі керування є вкладка «Меблі», яка забезпечує адміністрування каталогом товарів. Структура коду цього розділу поділена на два блоки: форму створення нової одиниці продукції, і таблицю вже наявних товарів. Фрагмент коду блоку додавання нових меблів наведено на рис. 2.68, а блок з таблицею наявних товарів зображено на рис. 2.69.

Додавання нової одиниці товару (меблів), реалізовано з використанням полів введення. Адміністратор веб-платформи заповнює такі параметри товару, як: назва, опис, ціна, посилання на зображення, матеріал, ширина, висота, глибина, вага та виробник. Після заповнення всіх необхідних параметрів та натискання кнопки «Додати товар» система відправляє запит до компоненту

“admin_action.php” із передачею параметра дії “add_furniture”. Фрагмент коду, що відповідає за обробку функції на сервері, проілюстровано на рис. 2.70. У разі успішного створення нової одиниці товару, функція створює новий запис у базі даних платформи та автоматично відображається на основній сторінці в секторі готової меблевої продукції.

Відображення вже наявних у каталозі товарів реалізована за допомогою імпорту інформації про меблі з відповідної таблиці бази даних та їхнього подальшого структурування в в блоці наявних товарів. Якщо в базі даних відсутні записи про наявні меблі, система виводить замість списку меблів текст що товарів немає. Для управління і актуалізування наявності товару передбачено функціонал з видаленням товару. У кожному рядку таблиці з наявним товаром міститься кнопка «Видалити», яка звертається до того ж самого компоненту “admin_action.php” із параметром дії “delete_furniture”. Код реалізації цієї функції наведено на рис. 2.71.

```
<div class="tab-panel active" id="tab-furniture">
    <div class="two-col">

        <div class="card">
            <div class="card-header"><h2>Додати новий товар</h2></div>
            <div class="card-body">
                <form method="post" action="admin_action.php">
                    <input type="hidden" name="action" value="add_furniture">
                    <div class="form-group">
                        <label>Назва *</label>
                        <input type="text" name="name" required>
                    </div>
                    <div class="form-group">
                        <label>Опис</label>
                        <textarea name="description"></textarea>
                    </div>
                    <div class="form-group">
                        <label>Ціна (₴) *</label>
                        <input type="number" name="price" step="0.01" min="0" required>
                    </div>
                    <div class="form-group">
                        <label>Посилання на фото (URL)</label>
                        <input type="url" name="image_url" placeholder="https://...">
                    </div>
                    <div class="form-group">
                        <label>Матеріал</label>
                        <input type="text" name="material" placeholder="Наприклад: Дуб / Тканина">
                    </div>
                    <div style="display:grid; grid-template-columns: 1fr 1fr 1fr; gap:10px;">
                        <div class="form-group">
                            <label>Ширина (см)</label>
                            <input type="number" name="width_cm" min="0">
                        </div>
                        <div class="form-group">
                            <label>Висота (см)</label>
                            <input type="number" name="height_cm" min="0">
                        </div>
                        <div class="form-group">
                            <label>Глибина (см)</label>
                            <input type="number" name="depth_cm" min="0">
                        </div>
                    </div>
                    <div class="form-group">
                        <label>Вага (кг)</label>
                        <input type="number" name="weight_kg" step="0.01" min="0">
                    </div>
                    <div class="form-group">
                        <label>Виробник</label>
                        <textarea name="manufacture"></textarea>
                    </div>
                    <button type="submit" class="btn btn-green">Додати товар</button>
                </form>
            </div>
        </div>
```

Рисунок 2.68 – Блок створення нової меблі

```php
            <div class="card">
                <div class="card-header">
                    <h2>Товари на сайті (<?php echo count($furniture); ?>)</h2>
                </div>
                <div style="overflow-x:auto;">
                    <table>
                        <thead>
                            <tr>
                                <th>Фото</th>
                                <th>Назва</th>
                                <th>Ціна</th>
                                <th>Дія</th>
                            </tr>
                        </thead>
                        <tbody>
                        <?php if (empty($furniture)): ?>
                            <tr><td colspan="4" style="text-align:center; padding:20px;">Товарів немає.</td></tr>
                        <?php else: ?>
                            <?php foreach ($furniture as $item): ?>
                            <tr>
                                <td>
                                    <?php if ($item['image_url']): ?>
                                        <img class="table-img" src="<?php echo htmlspecialchars($item['image_url']); ?>" alt="">
                                    <?php else: ?>
                                        <span>🛋️</span>
                                    <?php endif; ?>
                                </td>
                                <td><?php echo htmlspecialchars($item['name']); ?></td>
                                <td>₴<?php echo number_format((float)$item['price'], 2); ?></td>
                                <td>
                                    <form method="post" action="admin_action.php" onsubmit="return confirm('Видалити цей товар?')">
                                        <input type="hidden" name="action" value="delete_furniture">
                                        <input type="hidden" name="id" value="<?php echo $item['id']; ?>">
                                        <button type="submit" class="btn btn-sm btn-delete">Видалити</button>
                                    </form>
                                </td>
                            </tr>
                            <?php endforeach; ?>
                        <?php endif; ?>
                        </tbody>
                    </table>
                </div>
            </div>

        </div>
    </div>
```

Рисунок 2.69 – Блок з меблями які є в наявності

```php
    case 'add_furniture':
        $name      = $_POST['name'];
        $desc      = $_POST['description'];
        $price     = (float)$_POST['price'];
        $image_url = $_POST['image_url'];
        $material  = $_POST['material'];
        $width     = $_POST['width_cm'];
        $height    = $_POST['height_cm'];
        $depth     = $_POST['depth_cm'];
        $weight    = $_POST['weight_kg'];
        $manufacture = $_POST['manufacture'];

        if ($name == '' || $price <= 0) {
            $_SESSION['flash'] = 'Помилка: Назва та позитивна ціна є обов\'язковими.';
            break;
        }

        $stmt = $db->prepare(
            'INSERT INTO furniture (name, description, price, image_url, material, width_cm, height_cm, depth_cm, weight_kg, manufacture)
             VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)'
        );
        $stmt->execute([$name, $desc, $price, $image_url, $material, $width, $height, $depth, $weight, $manufacture]);

        $_SESSION['flash'] = "Товар " . htmlspecialchars($name) . " успішно додано.";
        break;

    // --- Видалення товару ---
    case 'delete_furniture':
        $id = isset($_POST['id']) ? (int)$_POST['id'] : 0;
        if ($id > 0) {
            $db->prepare('DELETE FROM furniture WHERE id = ?')->execute([$id]);
            $_SESSION['flash'] = 'Товар видалено.';
        }
        break;
```

Рисунок 2.70 – Фрагмент коду функції “add_furniture”

```
52          case 'delete_furniture':
53              $id = isset($_POST['id']) ? (int)$_POST['id'] : 0;
54              if ($id > 0) {
55                  $db->prepare('DELETE FROM furniture WHERE id = ?')->execute([$id]);
56                  $_SESSION['flash'] = 'Товар видалено.';
57              }
58              break;
```

Рисунок 2.71 – Фрагмент коду функції “delete_furniture”

Розділ керування останніми новинами представлено у вкладці «Новини», архітектура якої теж складається з двох основних блоків: форми створення нової новини та списку вже опублікованих новин. Загальну структуру коду цієї вкладки наведено на рис. 2.72.

Блок, який відповідає за створення нової новини, містить в собі три поля для введення даних: заголовок, текст публікації та посилання на зображення. Після заповнення цих параметрів та натискання кнопки «Опублікувати», форма надсилає запит до компоненту “admin_action.php” з параметром функцією “add_news”. Код реалізації функції наведено на рис. 2.73.

У блоці відображення вже опублікованих новин використовується інформація, що була попередньо отримана від бази даних платформи, та імпортована у відповідні поля інтерфейсу. Якщо база повернула пусту відповідь з інформацією про наявність опублікованих новин, система виводить замість новин текст про те що новин ще немає. Для адміністрування опублікованих новин передбачено функціонал видалення конкретної публікації, біля кожної новини є кнопка «Видалити», після натискання якої відбувається звернення до головного компоненту “admin_action.php” з параметром функцією “delete_news”, код якого зображено на рис. 2.74.



```
<div class="tab-panel" id="tab-news">
    <div class="two-col">

        <div class="card">
            <div class="card-header"><h2>Створити новину</h2></div>
            <div class="card-body">
                <form method="post" action="admin_action.php">
                    <input type="hidden" name="action" value="add_news">
                    <div class="form-group">
                        <label>Заголовок *</label>
                        <input type="text" name="title" required>
                    </div>
                    <div class="form-group">
                        <label>Текст новини *</label>
                        <textarea name="body" style="min-height:140px;" required></textarea>
                    </div>
                    <div class="form-group">
                        <label>Посилання на зображення</label>
                        <input type="url" name="image_url" placeholder="https://...">
                    </div>
                    <button type="submit" class="btn btn-green">Опублікувати</button>
                </form>
            </div>
        </div>

        <div class="card">
            <div class="card-header">
                <h2>Опубліковані новини (<?php echo count($newsItems); ?>)</h2>
            </div>
            <div style="overflow-x:auto;">
                <table>
                    <thead>
                        <tr>
                            <th>Дата</th>
                            <th>Заголовок</th>
                            <th>Дія</th>
                        </tr>
                    </thead>
                    <tbody>
                    <?php if (empty($newsItems)): ?>
                        <tr><td colspan="3" style="text-align:center; padding:20px;">Новин немає.</td></tr>
                    <?php else: ?>
                        <?php foreach ($newsItems as $post): ?>
                        <tr>
                            <td style="white-space:nowrap;"><?php echo date('d.m.Y', strtotime($post['created_at'])); ?></td>
                            <td><?php echo htmlspecialchars($post['title']); ?></td>
                            <td>
                                <form method="post" action="admin_action.php" onsubmit="return confirm('Видалити цю новину?')">
                                    <input type="hidden" name="action" value="delete_news">
                                    <input type="hidden" name="id" value="<?php echo $post['id']; ?>">
                                    <button type="submit" class="btn btn-sm btn-delete">Видалити</button>
                                </form>
                            </td>
                        </tr>
                        <?php endforeach; ?>
                    <?php endif; ?>
                    </tbody>
                </table>
            </div>
        </div>

    </div>
</div>
```

Рисунок 2.72 – Код вкладки новин

```
// --- Додавання новини ---
case 'add_news':
    $title     = $_POST['title'];
    $body      = $_POST['body'];
    $image_url = $_POST['image_url'];

    if ($title == '' || $body == '') {
        $_SESSION['flash'] = 'Помилка: Заголовок і текст новини обов\'язкові.';
        break;
    }

    $stmt = $db->prepare('INSERT INTO news (title, body, image_url) VALUES (?, ?, ?)');
    $stmt->execute([$title, $body, $image_url]);

    $_SESSION['flash'] = "Новину \"" . htmlspecialchars($title) . "\" опубліковано.";
    break;
```

Рисунок 2.73 – Код функції “add_news”

```
case 'delete_news':
    $id = isset($_POST['id']) ? (int)$_POST['id'] : 0;
    if ($id > 0) {
        $db->prepare('DELETE FROM news WHERE id = ?')->execute([$id]);
        $_SESSION['flash'] = 'Новину видалено.';
    }
    break;
```

Рисунок 2.74 – Код функції “delete_news”

Розділ з розміщеними замовленнями представлено у вкладці «Замовлення», код якої наведено на рис. 2.75. Динамічне виведення кількості замовлень поруч із назвою вкладки реалізованою за аналогічною схемою підрахунку елементів масиву як і у навігації панелі.

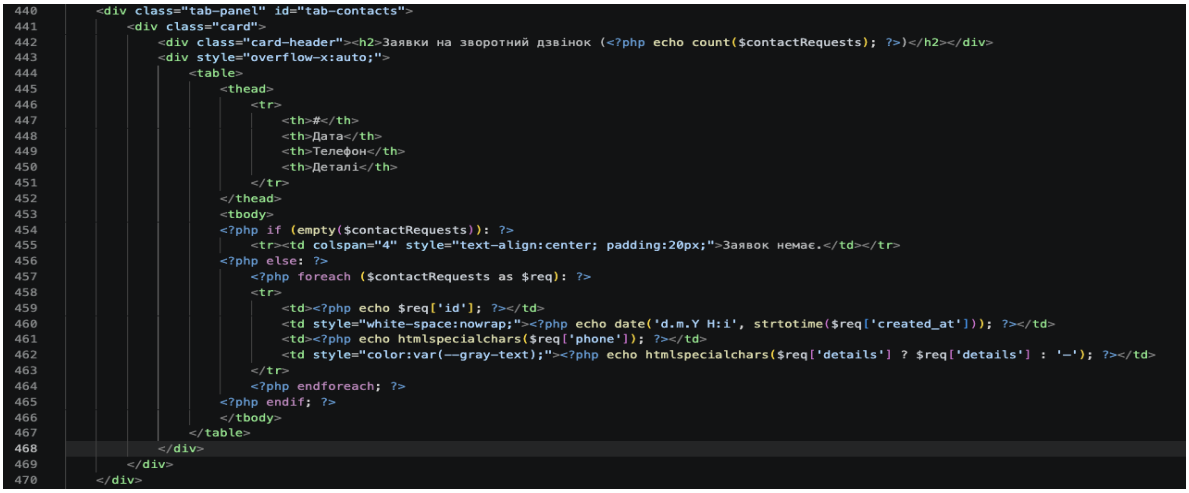
В інтерфейсі цього розділі відображаються детальні дані стосовно оформлених покупок, які попередньо отримано за допомогою бази даних, аналогічно і до інших вкладок, якщо у базі даних немає інформації про звернення буде виведено повідомлення що звернень немає.

```
<div class="tab-panel" id="tab-orders">
    <div class="card">
        <div class="card-header"><h2>Замовлення клієнтів (<?php echo count($orders); ?>)</h2></div>
        <div style="overflow-x:auto;">
            <table>
                <thead>
                    <tr>
                        <th>#</th>
                        <th>Дата</th>
                        <th>Клієнт</th>
                        <th>Телефон</th>
                        <th>Товари</th>
                        <th>Сума</th>
                    </tr>
                </thead>
                <tbody>
                <?php if (empty($orders)): ?>
                    <tr><td colspan="6" style="text-align:center; padding:20px;">Замовлень немає.</td></tr>
                <?php else: ?>
                    <?php foreach ($orders as $order):
                        $items = json_decode($order['items_json'], true);
                        if (!is_array($items)) { $items = array(); }
                    ?>
                    <tr>
                        <td><?php echo $order['id']; ?></td>
                        <td style="white-space:nowrap;"><?php echo date('d.m.Y H:i', strtotime($order['created_at'])); ?></td>
                        <td><?php echo htmlspecialchars($order['first_name'] . ' ' . $order['last_name']); ?></td>
                        <td><?php echo htmlspecialchars($order['phone']); ?></td>
                        <td>
                            <?php foreach ($items as $it): ?>
                                <span class="badge badge-green">
                                    <?php echo htmlspecialchars($it['name']); ?> ×<?php echo $it['qty']; ?>
                                </span>
                            <?php endforeach; ?>
                        </td>
                        <td><strong>₴<?php echo number_format((float)$order['total'], 2); ?></strong></td>
                    </tr>
                    <?php endforeach; ?>
                <?php endif; ?>
                </tbody>
            </table>
        </div>
    </div>
</div>
```

Рисунок 2.75 – Код вкладки з замовленням клієнтів

Розділ з залишеними користувачами заявками на зворотній зв'язок відображено у вкладці «Заявки», яка працює за аналогічним принципом, що й вкладка «Замовлення». Фрагмент коду цієї вкладки зображено на рис. 2.76.

Для відображення залишених заявок, інформація таким же чином імпортується з представлених базою даних, якщо даних не надано в таблиці відображається що залишених заявок немає.



```
<div class="tab-panel" id="tab-contacts">
    <div class="card">
        <div class="card-header"><h2>Заявки на зворотний дзвінок (<?php echo count($contactRequests); ?>)</h2></div>
        <div style="overflow-x:auto;">
            <table>
                <thead>
                    <tr>
                        <th>#</th>
                        <th>Дата</th>
                        <th>Телефон</th>
                        <th>Деталі</th>
                    </tr>
                </thead>
                <tbody>
                <?php if (empty($contactRequests)): ?>
                    <tr><td colspan="4" style="text-align:center; padding:20px;">Заявок немає.</td></tr>
                <?php else: ?>
                    <?php foreach ($contactRequests as $req): ?>
                    <tr>
                        <td><?php echo $req['id']; ?></td>
                        <td style="white-space:nowrap;"><?php echo date('d.m.Y H:i', strtotime($req['created_at'])); ?></td>
                        <td><?php echo htmlspecialchars($req['phone']); ?></td>
                        <td style="color:var(--gray-text);"><?php echo htmlspecialchars($req['details'] ? $req['details'] : '—'); ?></td>
                    </tr>
                    <?php endforeach; ?>
                <?php endif; ?>
                </tbody>
            </table>
        </div>
    </div>
</div>
```

Рисунок 2.76 – Код вкладки заявок

На даний момент написання коду завершено, веб-платформа для підприємства «ЄвроМеблі+» має:

– блоки: header , footer,

– сектор домашньої сторінки яка є гарною візиткою підприємства;

– сектор вибору готових меблів які коригуються в онлайн-часі через адміністративну панель веб-платформи;

– сектор останніх новин, які також можна додавати і видаляти в режимі онлайн, за допомогою адмін-панелі;

– сектор зворотнього зв’язку який зберігає і записує запити клієнтів для майбутнього зв’язку з менеджерами платформи;

– сторінку з кошиком, яка є динамічною і гнучкою що надає клієнту можливість легко створювати замовлення

– адміністративну панель, яка дозволяє в реальному часі дивитись на створенні заявки на зворотній зв’язок, замовлення, так і має функціонал додавання/видалення готових меблів, новини на веб-платформу.

– дизайн який має мобільну реакцію, є трендовим і сучасним;

На даний момент розробку сайту завершено.

ВИСНОВКИ

В даній кваліфікаційні роботі мною було описано процеси розробки веб-платфороми для мебельного підприємства «ЄвроМеблі+», та організування захисту від SQL-ін’єкцій і XSS-атак

Визначено та опрацьовано основні етапи для виконання даної задачі:

– виконано аналіз технології та обґрунтований вибір використовуваних технології,

– прописано технічне завдання щодо розробки макету веб-платформи,

– прописано технічне завдання щодо написання коду та верстки для веб-платформи,

– продемонстровано фінальний макет дизайну веб-платформи,

– прописано хід створення коду веб-сайту,

– надано фрагменти коду веб-сторінки,

– описані форми захисту від SQL-ін’єкцій і XSS-атак,

Виконана робота є змістовним базисом для підприємства, а саме для відділу продажів та інновацій даної компанії.

Впровадження інструментів, описаних у даній роботі, створюють додаткове джерело доходу та кардинального розвитку підприємства.

Загалом, розробка якісної веб-платформи, дозволить підприємству «ЄвроМеблі+» мати представлення в мережі, реалізовувати і просувати як і свої готові меблі так і меблі інших брендів, розміщати останні новини свого підприємства, отримувати інтернет-замовлення.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Що таке віртуальна машина та як вона працює [Електронний ресурс] / Аліна Мариненко // gigacloud. – 2025. – Режим доступу до ресурсу: https://gigacloud.ua/articles/shho-take-virtualna-mashyna-ta-yak-vona-praczyuye/
2. Apache vs Nginx: Practical Considerations [Електронний ресурс] / Alex Garnett / Justin Ellingwood // digitalocean. – 2022. – Режим доступу до ресурсу: https://www.digitalocean.com/community/tutorials/apache-vs-nginx-practical-considerations
3. Usage Statistics and Market Share of Web Servers, July 2026 [Електронний ресурс] // w3techs. – 2026. – Режим доступу до ресурсі: https://w3techs.com/technologies/overview/web_server
4. The History of HTML | From the HTML 1.0 spec to XHTML 1.0... [Електронний ресурс] / Ross Shannon // youthtmlsource. – 2025. – Режим доступу до ресурсу: https://www.yourhtmlsource.com/starthere/historyofhtml.html
5. HTML5 Introduction [Електронний ресурс] / kartikgoel1999 // geeksforgeeks. – 2025. – Режим доступу до ресурсу: https://www.geeksforgeeks.org/html/html5-introduction/
6. A History of CSS Through Fifteen Years of 24 ways [Електронний ресурс] / Rachel Andrew // 24ways. – 2019. – Режим доступу до ресурсу: https://24ways.org/2019/a-history-of-css-through-15-years-of-24-ways/
7. CSS Grid vs Flexbox. CSS Grid vs Flexbox: When to Use Each… [Електронний ресурс] // Lukman Faiz / Medium. – 2025. – Режим доступу до ресурсу: https://medium.com/@lukmanfaiz117/css-grid-vs-flexbox-caa33bce0695
8. What is JavaScript, and why is it important? [Електронний ресурс] / Apu sutra dhar // Medium. – 2023. – Режим доступу до ресурсу: https://medium.com/@apusutradhar77/what-is-javascript-and-why-is-it-important-6e1c63a4a8d1
9. What is AJAX?. AJAX (Asynchronous JavaScript and XML)… | by Jefferson Soares Costa Junior [Електронний ресурс] / Jefferson Soares Costa Junior // Medium. – 2024. – Режим доступу до ресурсу: https://medium.com/@jeffersonscjunior/what-is-ajax-b81c639a1f66

10. PHP Introduction [Електронний ресурс] / harsh.agarwal0 // Geeksforgeeks. – 2025. – Режим доступу до ресурсу: https://www.geeksforgeeks.org/php/php-introduction/

11. Usage Statistics and Market Share of PHP for Websites, July 2026 [Електронний ресурс] // w3techs. – 2026. – Режим доступу до ресурсу: https://w3techs.com/technologies/details/pl-php

12. NOSQL vs SQL. Key differences and when to choose each [Електронний ресурс] // version-2.com. – 2024. – Режим доступу до ресурсу: https://version-2.com/en/2024/02/nosql-vs-sql-key-differences-and-when-to-choose-each/

13. PostgreSQL vs MySQL: The Critical Differences [Електронний ресурс] / Anthony Schultes // integrate.io. – 2024. – Режим доступу до ресурсу: https://www.integrate.io/blog/postgresql-vs-mysql-which-one-is-better-for-your-use-case/

14. What is phpMyAdmin? [Електронний ресурс] // Yash Agarwal / scaler.com. – 2024. – Режим доступу до ресурсу: https://www.scaler.com/topics/what-is-php-myadmin/

15. Що таке макет сайту та навіщо він потрібен [Електронний ресурс] / hosting.pat@gmail.com // patprofi. – 2022. – Режим доступу до ресурсу: https://patprofi.com/shho-take-maket-sajtu-ta-navishho-vin-potriben/

16. Wireframe Vs. Mockup – What's The Difference? [Електронний ресурс] // figma. – 2024. – Режим доступу до ресурсу: https://www.figma.com/resource-library/wireframe-vs-mockup/

17. Що таке Figma та для кого вона потрібна [Електронний ресурс] / Kukurudza // cases.media. – 2024. – Режим доступу до ресурсу: https://cases.media/article/sho-take-figma-ta-dlya-kogo-vona-potribna

18. Верстка це? Що таке верстка сайту? [Електронний ресурс] / Promodex // it-rating.ua. – 2026. – Режим доступу до ресурсу: https://it-rating.ua/scho-take-verstka-saytu

19. Header що це таке? [Електронний ресурс] Вікторія Драч // project-seo.net. – 2019. – Режим доступу до ресурсу: https://project-seo.net/terms-uk/header-2/

20. Що таке футер: визначення, приклади та поради [Електронний ресурс] / Наталя Устименко // esputnik. – 2024. – Режим доступу до ресурсу: https://esputnik.com/uk/slovnyk-email-marketologa/futer