ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ

Навчально-науковий інститут денної освіти

Форма навчання денна

Кафедра комп’ютерних наук та інформаційних технологій

Допускається до захисту

Завідувач кафедри

_____ Олена ОЛЬХОВСЬКА

(підпис)

«___»_____2024р.

КВАЛІФІКАЦІЙНА РОБОТА

на тему

РОЗРОБКА САЙТУ ІНТЕРНЕТ-МАГАЗИНУ «ГАЛЕРЕЯ ШТОР»

зі спеціальності 122 Комп’ютерні науки

освітня програма «Комп’ютерні науки»

ступеня бакалавра

Виконавець роботи Дудник Костянтин Олександрович

_____«____»_____ 2024 р.

(підпис)

Науковий керівник к. ф.-м. н., доцент, Парфьонова Тетяна Олександрівна

_____«____»_____ 2024 р.

(підпис)

Рецензент _____

ПОЛТАВА 2024

РЕФЕРАТ

Записка: 79 с., 34 рис., 3 таблиць, джерела 19.

Об’єкт розробки – інтернет-магазин «Галерея штор»

Мета роботи – розробка інтернет-магазину "Галерея штор" з метою можливості швидкого та зручного придбання товарів магазину.

Методи реалізації – програмна реалізація веб-сайту була виконана з використанням HTML для структури сторінок, CSS для оформлення, JavaScript для взаємодії з користувачем, PHP для обробки серверної логіки, SQL для роботи з базою даних і XAMPP для створення та запуску локального веб-сервера.

У результаті тестування було виявлено деякі помилки в адаптивності макету, які були швидко виправлені. Інтернет-магазин відповідає поставленій задачі. Створення даного магазину було важливим та актуальним завданням у сучасному світі, оскільки інтернет-магазини стали необхідним інструментом для онлайн покупок для користувачів, а для бізнесу можливість збільшити прибуток та потік нових клієнтів.

ЗМІСТ

ЗМІСТ 3
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ 4
ВСТУП 5
1. ПОСТАНОВКА ЗАДАЧІ 7
2. ІНФОРМАЦІЙНА ЧАСТИНА 8
2.1. Огляд типів веб-сайтів та переваг інтернет-магазинів. 8
2.2. Основні етапи створення інтернет-магазину 10
2.3. Огляд інструментів і технологій, які будуть використовуватися під час розробки 11
2.3.1 Переваги MySQL 12
2.3.2 Переваги використання PHP 13
3. ТЕОРЕТИЧНА ЧАСТИНА 15
3.1. Проектування бази даних для інтернет-магазину 15
3.2 Розробка блок-схеми інтернет-магазину, яка потребує програмування 18
4. ПРАКТИЧНА ЧАСТИНА 20
4.1. Опис процесу розробки інтернет-магазину «Галерея штор» 20
4.2 Перевірка та тестування магазину на можливі помилки 51
ВИСНОВКИ 52
СПИСОК ІНФОРМАЦІЙНИХ ДЖЕРЕЛ 54
ДОДАТОК А. КОД ПРОГРАМИ 56

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

Умовні позначення, символи, скорочення, терміни	Пояснення умовних позначень, скорочень, символів
CSS	Cascading Style Sheets — Каскадні таблиці стилів
Figma	Figma — векторний онлайн-сервіс розробки інтерфейсів та прототипування.
HTML	HyperText Markup Language — мова розмітки гіпертексту
JavaScript	JavaScript (JS) — динамічна, об’єктно-орієнтовна прототипна мова програмування.
MySQL	MySQL — вільна система керування реляційними базами даних.
PHP	Hypertext PreProcessor — Скриптова мова програмування, для виконання задач на стороні сервера.
SQL	Structured query language — мова структурованих запитів до бази даних.
XAMPP	XAMPP — Платформа для розробки та тестування веб-додатків, яка включає в себе сервер Apache, базу даних MySQL, мову програмування PHP та інші інструменти.

ВСТУП

Предметом дизайну є створення і реалізація інтернет-магазину "Галерея штор", який включає в себе функції магазину. Основна увага в розробці приділяється наданню користувачам зручних і швидких інструментів для покупки товарів. Основне завдання створення інтернет-магазину виникає як з реальних потреб і потреб сучасного світу, так і з стрімкого темпу життя. Тепер ці магазини стали невід'ємною частиною нашого життя, можливістю для людей зручно і швидко купувати необхідні товари, не виходячи з дому.

Веб-сайт інтернет-магазину містить список товарів, які користувачі можуть замовити в Інтернеті. Основна відмінність між таким магазином і звичайним магазином полягає у відсутності фізичних інструментів. Якщо для звичайного магазину потрібно торгова площадка, магазин, цінник, продавець, консультант, то в програмі реалізована вся інфраструктура інтернет-магазину. Інтернет-магазини стають все більш популярними завдяки зручностям, які вони пропонують своїм користувачам. Основні проблеми виникають через те, що сучасному суспільству потрібно купувати зручно і швидко. Від дизайну до функціональності всі аспекти створення магазину спрямовані на максимальну зручність для покупців і спрощення процесу вибору і покупки товарів. Основна мета створення інтернет-магазину- забезпечити максимальний рівень зручності та доступності для покупців.

Основна мета – створити зручний, надійний і в той же час простий інструмент для покупки товарів в Інтернеті. Це дозволяє збільшити аудиторію та залучити більше потенційних клієнтів.

Крім того, для інтернет-магазинів це зручна презентація всіх продуктів в асортименті та можливість пропонувати додаткові послуги та продукти.

При проектуванні магазину фіранок ми використовуємо передові технології і програмні рішення, які дозволяють створити зручний і ефективний магазин. Ключовими міркуваннями є безпека, конфіденційність даних та зручність їх

використання. Функціональність магазину також покращується для задоволення зростаючих потреб користувачів.

При розробці веб-магазину "Галерея штор" будуть використані сучасні технології, включаючи мови програмування та бази даних.

Метою кваліфікаційної роботи є створення зручного інтернет-магазину для придбання під замовлення штор та тюль, користувачі зможуть швидко замовити потрібний їм розмір пошиття, кольору та одразу зручно сплатити. Також якщо клієнту буде потрібна допомога, на сайті є зручна форма для зв'язку із магазином.

Подальший зміст роботи буде представлений в таких розділах:

- Вступ.
- Постановка задачі.
- Інформаційна частина.
- Теоретична частина.
- Практична частина.
- Висновки.

Кожен з цих розділів міститиме важливу інформацію та огляд, спрямований на досягнення поставлених цілей та завдань.

Виконанням дипломного проекту є створений та готовий до використання інтернет-магазин "Галерея штор".

Обсяг роботи складає 78 сторінок, з яких 55 основного тексту, 23 сторінки – додаток.

1. ПОСТАНОВКА ЗАДАЧІ

Основною задачею роботи було створити зручний інтернет-магазин, щоб клієнти змогли швидко замовити потрібний їм товар.

Основні завдання, які виникли при реалізації проекту наступні:

1. **Дослідження та вивчення існуючих типів сайтів та інтернет-магазинів:** Дослідити які типи сайтів існують, чим вони відрізняються по функціональності роботи та які технології створення використовуються.
2. **Розробка інтерфейсу сайту:** Розробити інтуїтивно зрозумілий та зручний інтерфейс сайту, який буде включати головну сторінку, каталог, підкаталог, сторінку товару, кошик, та оформлення замовлення.
3. **Створення бази даних:** Зробити базу даних, де буде зберігатися весь товар.
4. **Розробка сайту із використанням сучасних технологій:** Використовуючи сучасні технології мов програмувань створити зручний та адаптивний сайт.
5. **Тестування:** Провести тестування сайту на можливі помилки та виправити їх.

Завдання та цілі цього проекту будуть докладно розглянуті та реалізовані в подальших розділах даного проекту.

2. ІНФОРМАЦІЙНА ЧАСТИНА

2.1. Огляд типів веб-сайтів та переваг інтернет-магазинів.

Сайт - це веб-сторінка або група веб-сторінок, які знаходяться на одному домені інтернету. Вони доступні користувачам за допомогою веб-браузера.

Найпоширеніші за типом сайти:

- **Сайт-візитка:** Ці сайти не мають багато сторінок. Зазвичай подібні сайти використовують різні компанії для короткого опису, знайомства користувачами із продукцію та послугами а також можливість зв'язку.
- **Лендіг (Landing page):** Подібні сайти зазвичай односторінкові, за винятком можуть мати декілька додаткових сторінок. Найчастіше використовуються для просування одного товару або певну групу товарі. Містить тільки важливу інформацію. Сайт фокусує та спонукає користувача придбати товар й не відпускати до кінця.
- **Інтернет-магазини (e-commerce):** Призначені для продажу товарів чи послуг онлайн. Завдяки веб-сайту e-commerce користувачі можуть переглядати і купувати товари та послуги онлайн у зручний для них час, вони роблять покупки простішими та без стресу. Після того, як товари додані до кошика, користувач зможе оформити замовлення за допомогою банківської картки або оплатити його через платіжні онлайн-платформи, такі як "Google Pay".
- **Корпоративні сайти:** Ці сайти найбільш підходять для середнього і великого бізнесу та для великих підприємств. Сайт має багато сторінок є велика кількість розділів яка містить інформацію про компанію або бізнес та їхню продукцію або послуги.
- **Блоги:** Створені для того щоб люди могли розміщувати письмову та візуальну інформацію на будь-які теми, які їм подобаються. Наприклад це може бути особистий блог про "Кулінарію" або "Подорожі".

- **Новинні портали:** Необхідні, щоб надавати людям новини на різні теми безплатно та своєчасно, але на деяких порталах потрібно придбати підписку. Вони є чудовою альтернативою традиційним засобам масової інформації, таким як газети, радіо і телебачення.
- **Навчальні сайти:** Використовуються навчальними закладами переважно для інформування всіх бажаючих про свій навчальний заклад або для надання різної інформації студентам. Також є освітні веб-сайти які пропонують онлайн-курси.
- **Соціальні мережі:** Ці сайти сьогодні є найпопулярнішими та найчастіше використовуваними серед усіх інших типів веб-сайтів. Майже кожен має власні акаунти у Instagram або іншій мережі, де вони використовують їх для спілкування, обміну інформацією та створення різного контенту.
- **Розважальні сайти:** Одними з найпопулярніших сайті для перегляду фільмів або відео є "YouTube" і "Netflix".

  Так ще багато інших .

Основні плюси інтернет-магазинів:

- **Глобальне охоплення:** Інтернет-магазин надає можливість подолати географічні бар'єри, щоб охопити глобальну аудиторію. Це розширить клієнтську базу і дозволить вийти на ринки, які раніше були недоступні. Це дозволяє пропонувати продукти і послуги широкому колу споживачів, збільшуючи потенціал продажів, клієнтів та зростання бізнесу.
- **Доступність 24/7:** На відміну від звичайних магазинів, які працюють за фіксованим графіком, інтернет-магазини можуть працювати цілодобово. Це зручно для клієнтів, які можуть переглядати та робити покупки в будь-який час, незалежно від місця розташування чи часового поясу. Це, в свою чергу, підвищує задоволеність клієнтів і сприяє збільшенню продажів, особливо компаніям, що обслуговують міжнародну аудиторію.
- **Можливість аналітики:** Інструменти аналізу дозволяють компаніям отримувати цінну інформацію про поведінку, уподобання та покупки клієнтів.

Цей підхід дозволяє розробляти цільові маркетингові стратегії, вдосконалювати сайт, а також адаптувати та оптимізувати пропозиції своїх продуктів на основі зворотного зв'язку в режимі реального часу. Результатом є краще прийняття рішень та покращення бізнес-результатів.

- **Масштабованість:** Завдяки електронній комерції яка дозволяє безперешкодно розширюватися без обмежень, пов'язаних з фізичним простором. Завдяки можливості додавати нові продукти, послуги та функції.

2.2. Основні етапи створення інтернет-магазину

Перший етап - стратегія: Ця стадія відноситься до абстрактної, адже вона побудована на бізнес цілях і щоб користувачі користувалися нашим продуктом потрібно дізнатися їхні цілі через (дослідження користувачів).

Другий етап - сфера застосування: На цьому етапі вирішується що продукт має робити, як буде працювати, які додаткові сервіси потрібні, об'єм роботи, ціна та терміни.

Третій етап - структура: Тут робимо схему взаємодії з нашим продуктом, усі можливі випадки помилок зі сторони користувача, наприклад, що буде якщо користувач натисне на ту чи іншу кнопку куди його перемістить. Також на цьому етапі можна підготувати текст який буде на майбутньому сайті.

Четвертий етап - скелет(прототип): Це вже стосується дизайну, як буде розташований кожен елемент сайту (фото, текст, кнопка та інше), щоб користувачу було зручно користуватися сайтом.

П'ятий етап - Зовнішність: Тут вже дизайн. Це те, що користувач буде бачити і чим користуватися.

Шостий етап - Програмування: На цьому етапі створюємо базу даних, розробляємо функціональність сайту та код для взаємодії з користувачем

використовуючи HTML, CSS та JavaScript для створення структури, стилізації та функціональності веб-сторінок.

Сьомий етап - Перевірка справності: Проводимо тестування всіх аспектів сайту, щоб переконатися, що він працює належним чином.

2.3. Огляд інструментів і технологій, які будуть використовуватися під час розробки

Під час розробки інтернет магазину «Галерея штор» будуть використовуватися такі інструменти та технології:

Інструмент	Опис
CSS	Мова стилів для веб-сторінок, яка визначає вигляд і форматування елементів, створених за допомогою HTML.
Figma	Це векторний онлайн-сервіс розробки інтерфейсів та прототипування.
HTML	Потрібно для розмітки для веб-сторінок.
JavaScript	Це мова програмування, яка додає інтерактивність до веб-сторінок, дозволяючи взаємодіяти з елементами на сторінці, змінювати їх вміст та реагувати на події користувача.
MySQL	Вільна система керування реляційними базами даних.
PHP	Це мова програмування для створення динамічних веб-сайтів та веб-додатків на стороні сервера.
SQL	Мова запитів, яка використовується для доступу до даних у базах даних.
XAMPP	Платформа для розробки та тестування веб-додатків, яка включає в себе сервер Apache, базу даних MySQL, мову програмування PHP та інші інструменти.

Вибір інструментів та технологій для розробки інтернет-магазину «Галерея штор» був проведений на основі таких критеріїв:

- **Доступність**. Всі інструменти та технології мають бути легкодоступними, легкими у використанні та оновлені до сучасних стандартів.

- **Надійність**. Інструменти і технології повинні бути надійними і запобігати виникненню системних збоїв.
- **Масштабованість**. Технології повинні бути гнучкими та здатними зростати разом із збільшенням обсягів трафіку та продажів.

На основі цих критеріїв було обрано наступні інструменти та технології:

- **Figma** – це безкоштовний векторний онлайн-сервіс розробки інтерфейсів та прототипування.
- **HTML, CSS та JavaScript** - це класичні технології які використовуються для розробки веб-сайтів і додатків, які добре себе зарекомендували та є доступними для більшості розробників.
- **PHP** - це відома мова програмування, яка добре підходить для створення веб-застосунків та є доступною і простою.
- **SQL** - Доволі популярна, проста та доступна мова запитів, яку використовують для отримання даних в БД.
- **MySQL** - Вільна система керування реляційними базами даних, що забезпечує широкі можливості для зберігання та обробки інформації.
- **XAMPP** - Популярна програма для запуску локального серверу Apache та бази даних MySQL й скриптів для мови програмування PHP. XAMPP є безкоштовною та доступною.

Вибір інструментів і технологій для інтернет-магазину "Галерея штор" - важливий етап, що визначає подальший розвиток магазину. Оптимально підібрані інструменти і технології в подальшому забезпечать створення надійної, масштабованої і ефективної системи.

2.3.1 Переваги MySQL

MySQL - це широко використовувана реляційна система управління базами даних з відкритим вихідним кодом. Вона відома своєю швидкістю та масштабованістю, що робить її чудовим вибором для електронної комерції з великою

кількістю продуктів та клієнтів. Багато провідних платформ електронної комерції, таких як WooCommerce, Magento та OpenCart, використовують MySQL.

- **Продуктивність:** База даних гарантує, що інтернет-магазин оброблятиме дані швидко та ефективно, що дозволяє пришвидшити завантаження сторінок та обробку транзакцій. Це також допомагає зменшити незручностей для клієнтів та допомагає підвищити продажі і конверсію.
- **Масштабованість:** У міру зростання магазину збільшується і обсяг даних. Базу даних легко масштабувати, і вона може адаптуватися до цього зростання, щоб у міру зростання бізнесу вона могла продовжувати надавати високоякісний користувальницький досвід.
- **Надійність:** Ефективна база даних забезпечує цілісність даних і захист від втрати даних, що має вирішальне значення для підтримки довіри клієнтів до інтернет-магазину і забезпечення безперебійної роботи.

2.3.2 Переваги використання PHP

PHP - це серверна мова сценаріїв, яка широко використовується для веб-розробки. Відомий своєю простотою та універсальністю.

- **Перша перевага -** відкритий вихідний код: Це важлива перевага для розвитку електронної комерції. Це означає, що розробники мають доступ до великої спільноти учасників та ресурсів. Це не тільки заохочує інновації та забезпечує постійний потік оновлень і вдосконалень, що робить PHP динамічною мовою, яка розвивається відповідно до потреб електронної комерції яка постійно вдосконалюється.
- **Друга перевага -** економічно ефективний розвиток: Для бізнесу економічна ефективність є вирішальним фактором. PHP, будучи мовою з відкритим вихідним кодом, усуває потребу у великих ліцензійних зборах. Вартість розробки значно знижується, що робить його привабливим варіантом для стартапів і малих і середніх підприємств.

- **Третя перевага -** універсальність і сумісність: PHP легко інтегрується з різними базами даних, що робить його універсальним вибором для розробки електронної комерції. Незалежно від того, яка буде обрана база даних, PHP може працювати з усіма. Така сумісність гарантує, що розробники можуть вибрати базу даних, яка найкраще відповідає конкретним вимогам проекту електронної комерції.
- **Четверта перевага -** масштабованість для зростання: Платформи електронної комерції – це динамічні платформи, які часто швидко розвиваються. PHP зі своєю масштабованою архітектурою дозволяє компаніям безперешкодно розширювати свою присутність в Інтернеті. Незалежно від того, чи йдеться про роботу зі збільшенням кількості користувачів, транзакцій чи продуктів, PHP забезпечує масштабованість, необхідну для зростання без шкоди для продуктивності.
- **П'ята перевага -** підтримка спільноти та документація: Спільнота PHP велика й активна. Це означає велику кількість документації, навчальних посібників та форумів, де розробники можуть шукати поради та вирішення різних проблем.

Вибір мови програмування є ключовим у формуванні успіху інтернет-магазину. PHP з відкритим вихідним кодом, економічною ефективністю, універсальністю, масштабованістю та надійними функціями безпеки стає найкращим варіантом.

3. ТЕОРЕТИЧНА ЧАСТИНА

3.1. Проектування бази даних для інтернет-магазину

В цьому розділі буде розглянуто проектування структури бази даних для інтернет-магазину.

Список необхідних таблиць:

Таблиця 3.1 – catalog містить інформацію про основні категорії товарів.

Поле	Тип	Опис
id	INT	Унікальний ідентифікатор каталогу
name	VARCHAR(30)	Назва групи товарів в каталозі

Таблиця 3.2 – subcatalog містить інформацію про товари.

Поле	Тип	Опис
id	INT	Унікальний ідентифікатор підкаталогу
name	VARCHAR(30)	Назва товару
image	VARCHAR(50)	Фото товару
catalog_id	INT	Індефікатор каталогу

Таблиця 3.3 – products містить інформацію про товар.

Поле	Тип	Опис
id	INT	Унікальний ідентифікатор продукту
name	VARCHAR(30)	Назва товару
image	VARCHAR(50)	Головне фото товару
image_s1	VARCHAR(50)	Додаткове фото 1
image_s2	VARCHAR(50)	Додаткове фото 2
image_s3	VARCHAR(50)	Додаткове фото 3
price	DECIMAL(10,2)	Ціна товару
composition	VARCHAR(40)	Склад тканини
figure	VARCHAR(40)	Малюнок
manufacturer	VARCHAR(40)	Виробник
density	VARCHAR(40)	Щільність тканини
protection	VARCHAR(40)	Захист від сонця
care	VARCHAR(40)	Догляд
colors	TEXT	Кольори
is_exist	TINYINT(1)	В наявності/відсутній
subcatalog_id	INT	Ідентифікатор subcatalog

База даних подана у вигляді блок-схеми (рис.3.1).

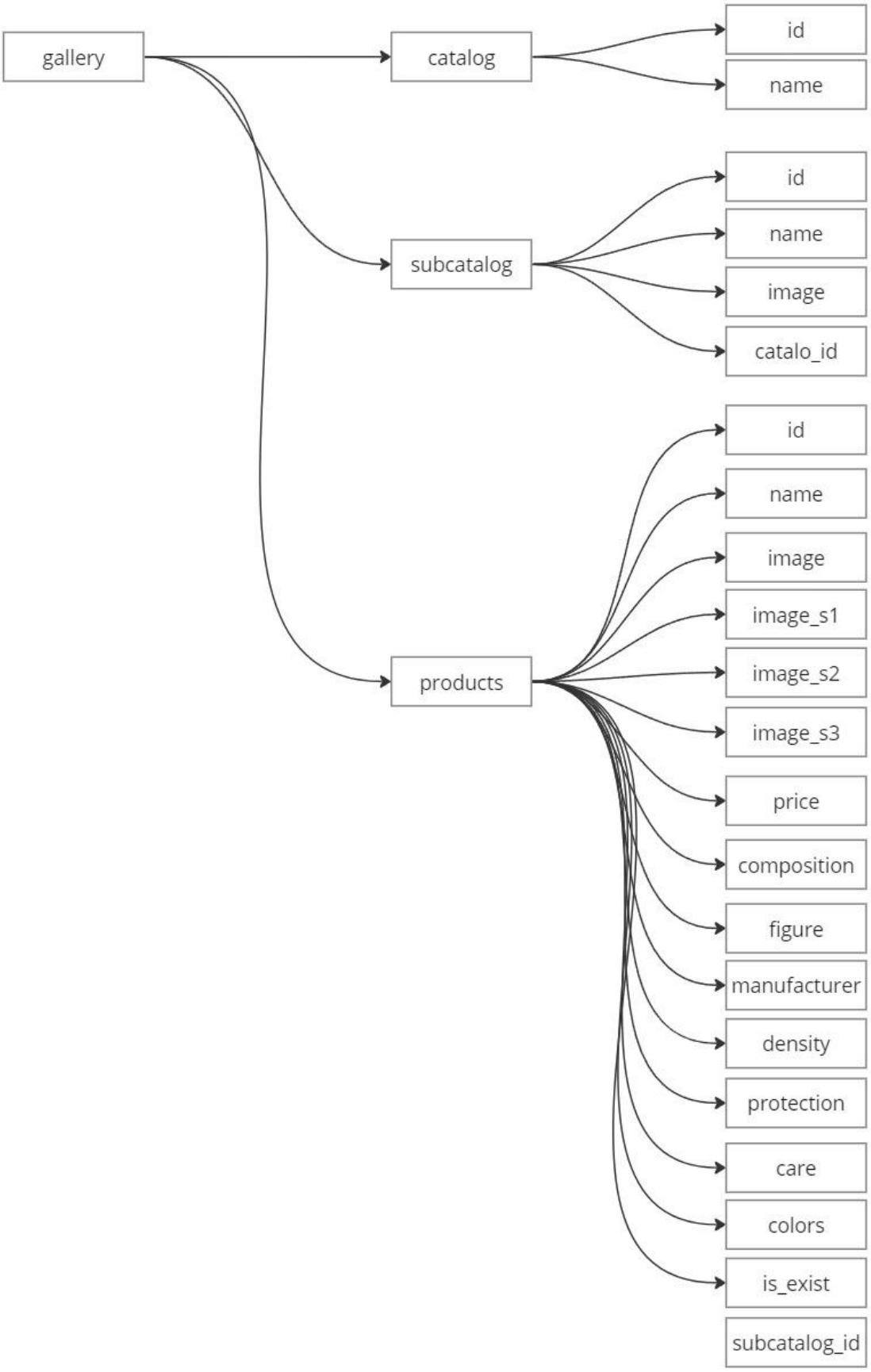


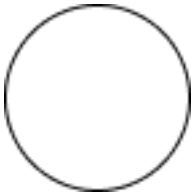
Рисунок 3.1 – Схема Баз даних

Зв'язок між таблицями:

- Таблиця **gallery** має посилання на всі інші таблиці (**catalog**, **subcatalog**, **products**).
- Таблиця **catalog** пов'язана з таблицею **subcatalog** за допомогою **catalog_id**.
- Таблиця **subcatalog** пов'язана з таблицею **products** за допомогою **subcatalog_id**.

3.2 Розробка блок-схеми інтернет-магазину, яка потребує програмування

Таблиця 3.4 – містить символи блок-схеми.

Інструмент	Опис
	Показує головні сторінки сайту.
	Лінії з'єднують різні елементи блок-схеми, показуючи послідовність переходів між сторінками та діями.
	Показує секції на сторінці. Наприклад: слайдер або відгуки.
	Дія. Виникає, коли користувач взаємодіє з чимось, наприклад, з кнопкою на сайті.

Блок схема сайту відображено на рис. 3.2.

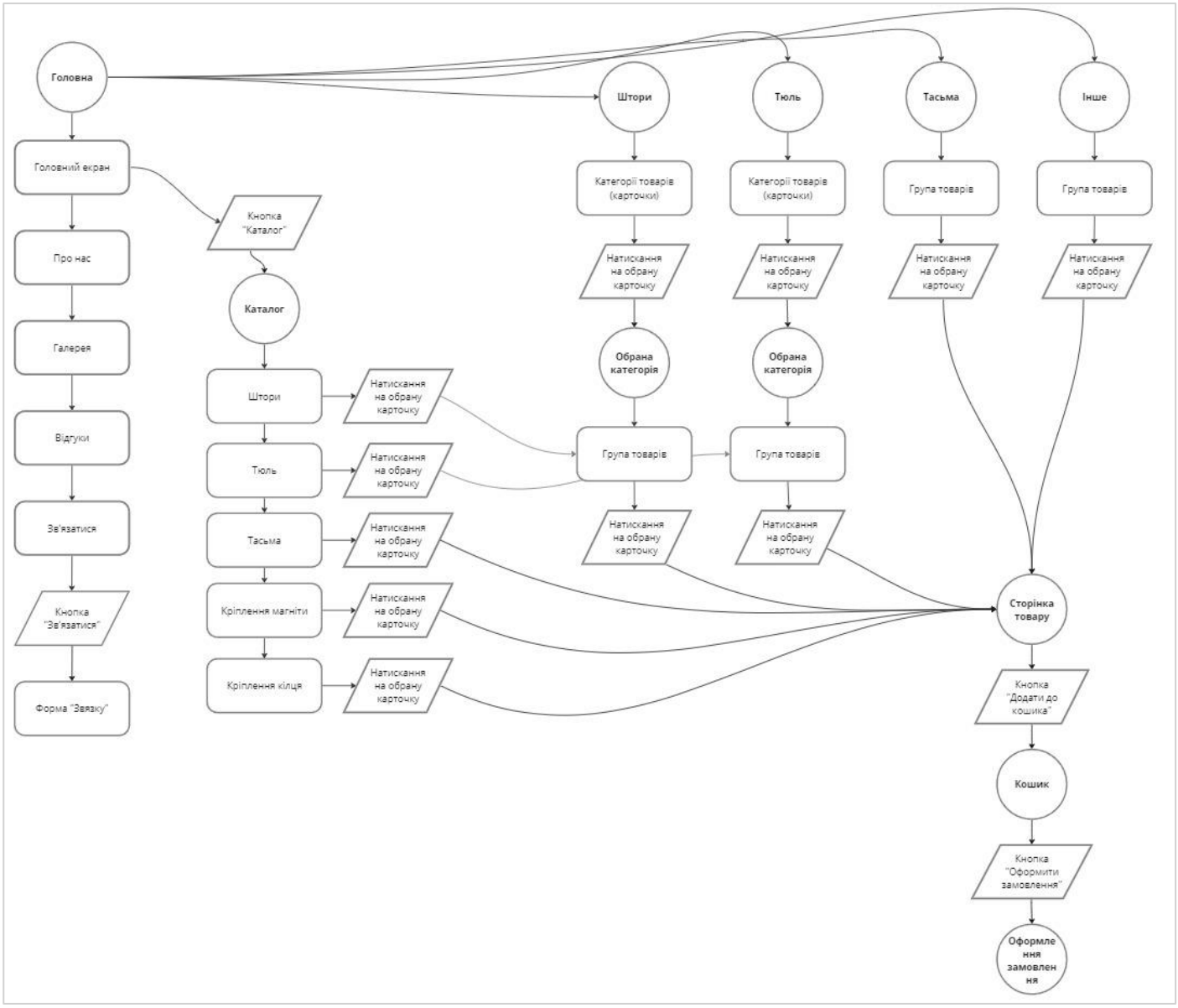


Рисунок 3.2 – Схема Баз магазину

4. ПРАКТИЧНА ЧАСТИНА

4.1. Опис процесу розробки інтернет-магазину «Галерея штор»

Спершу проектуємо інтерфейс магазину в програмі Figma

Перший екран головної сторінки (рис. 4.1) – шапка сайту, який привертатиме увагу користувачів. На цьому екрані розміщена головна кнопка дії, яка перенаправляє на сторінку каталогу (рис. 4.7).

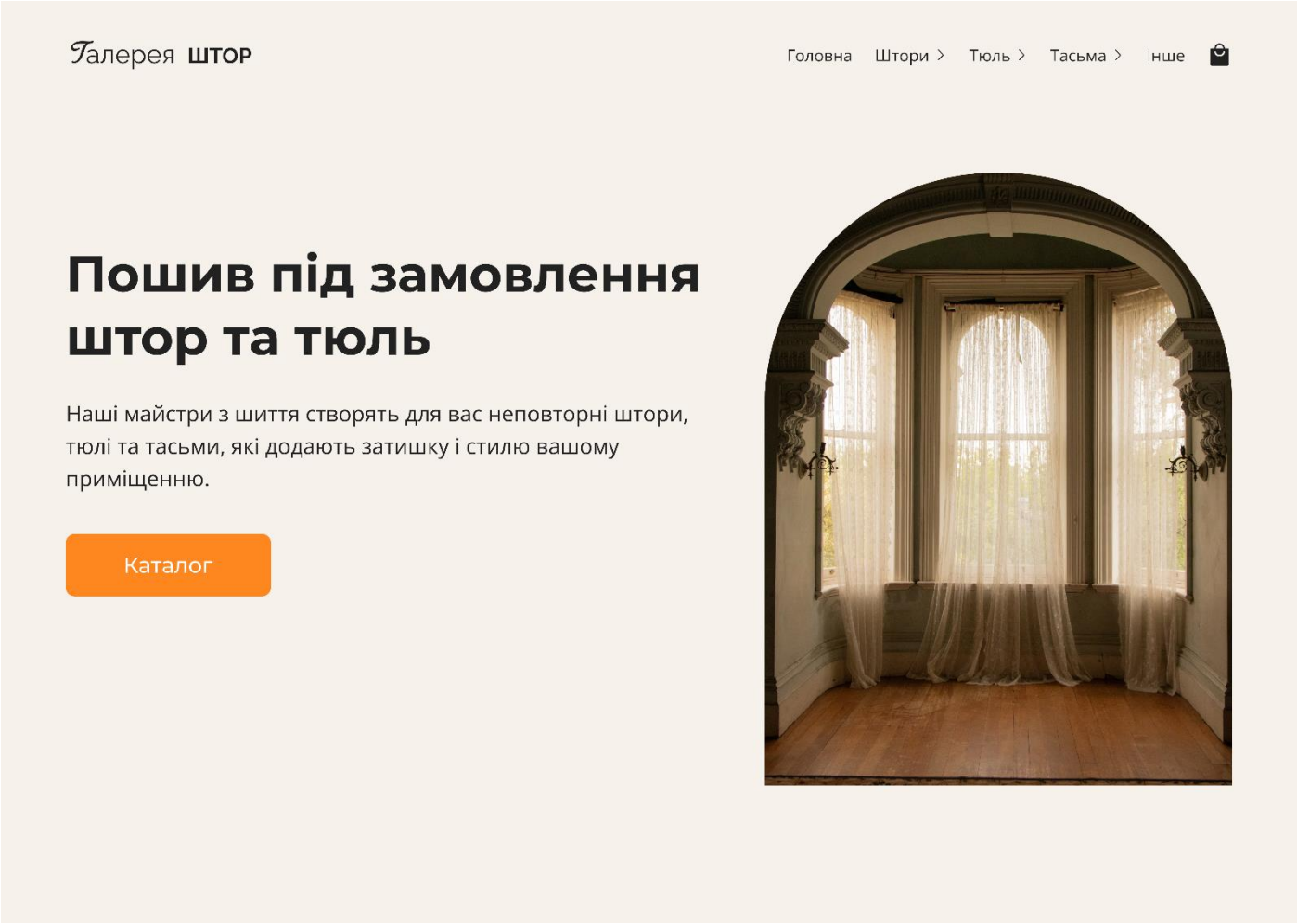


Рисунок 4.1 – Шапка сайту

Другий екран (рис. 4.2) містить інформацію про наш магазин. Тут користувачі можуть дізнатися про магазин.

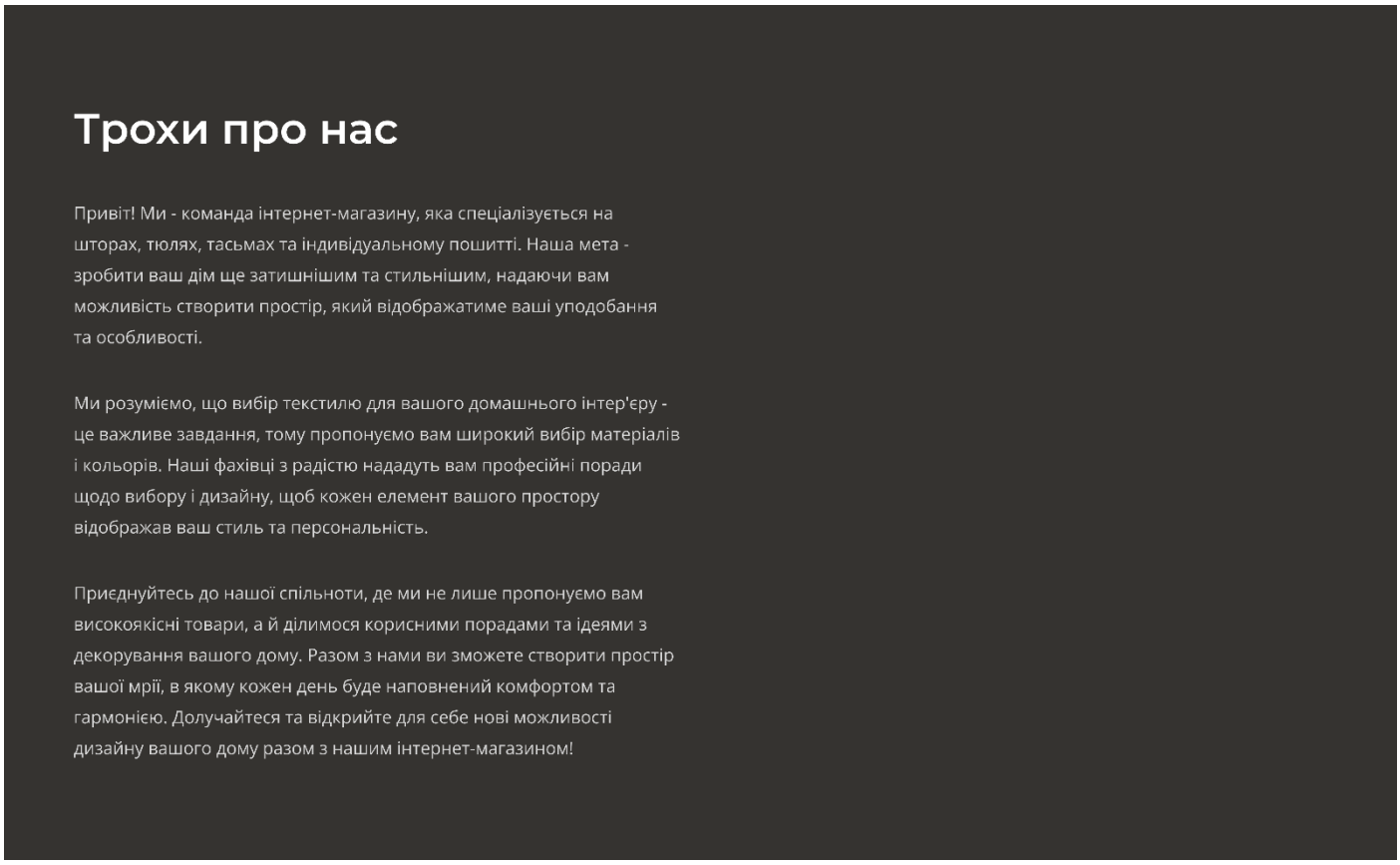


Рисунок 4.2 – Інформація про магазин

На третьому екрані (рис. 4.3) є слайдер з фотографіями покупців, там зображено пошиті під замовлення штор та тюль з магазину. Цей слайдер дає можливість побачити вироби в справжньому інтер'єрі, підкреслюючи якість і естетичну привабливість.

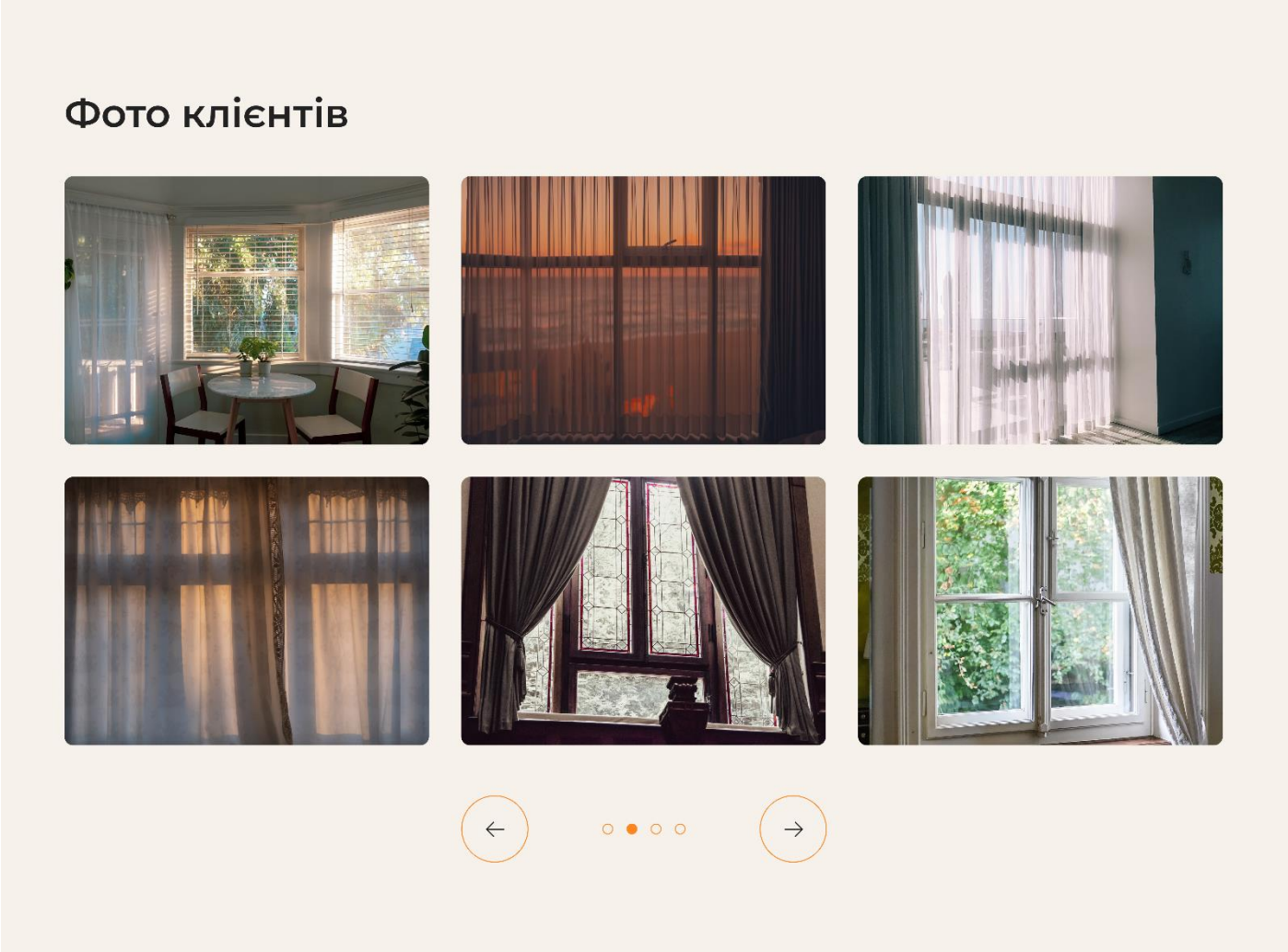


Рисунок 4.3 – Слайдер

Четвертий екран (рис. 4.4) містить відгуки клієнтів та посилання, яке перенаправляє на сторінку з відгуками в Google. В цьому екрані користувачі можуть ознайомитися з реальними враженнями інших покупців.

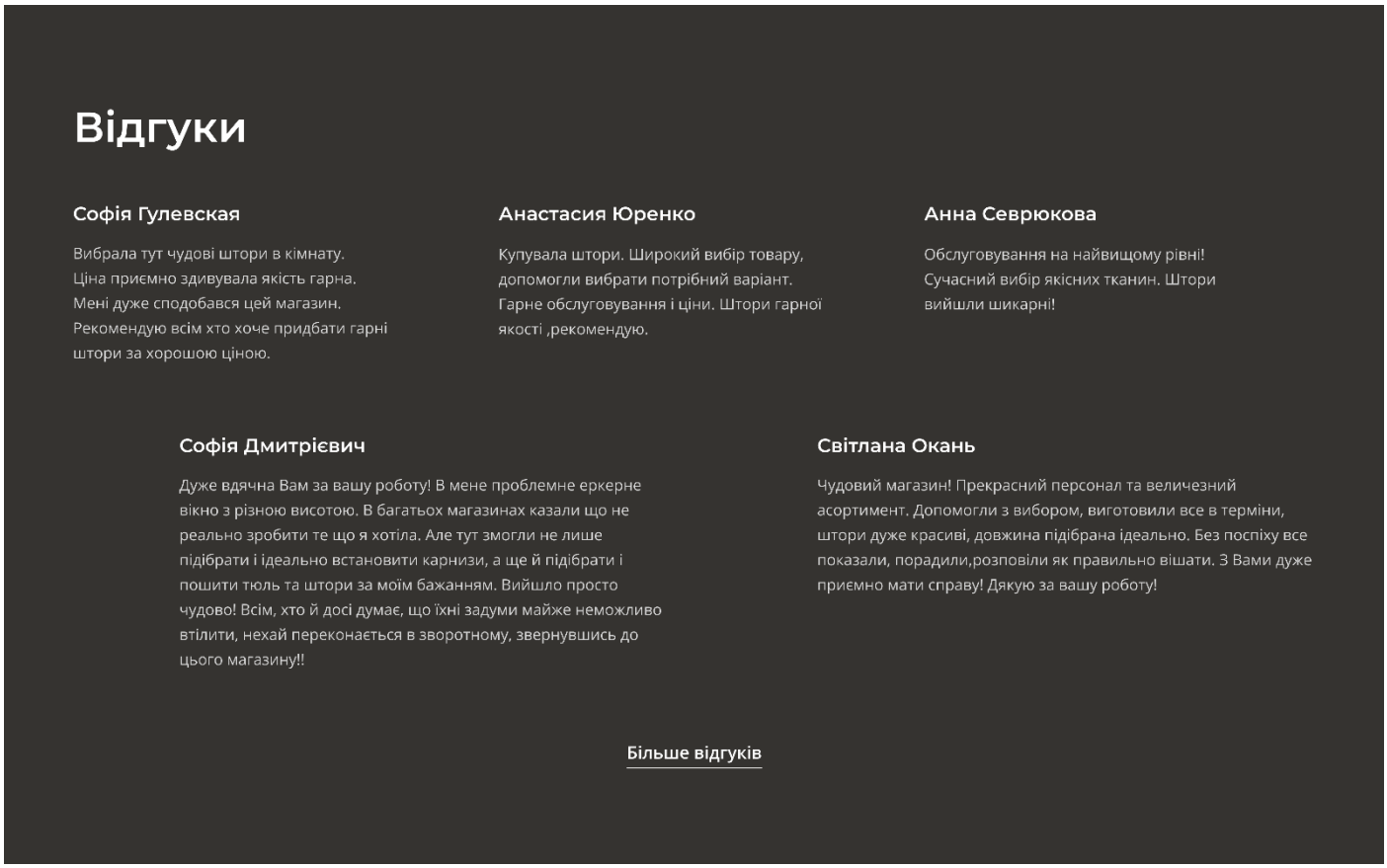


Рисунок 4.4 – Відгуки

П'ятий екран (рис. 4.5) містить інформацію про магазин: години роботи, адреса, номер телефону, електрона пошта та вбудована карта для зручної навігації. Після натискання кнопки відкриває форму для зв'язку з магазином (рис. 4.6) і користувачі можуть швидко та легко зв'язатися якщо виникнуть будь які запитання.

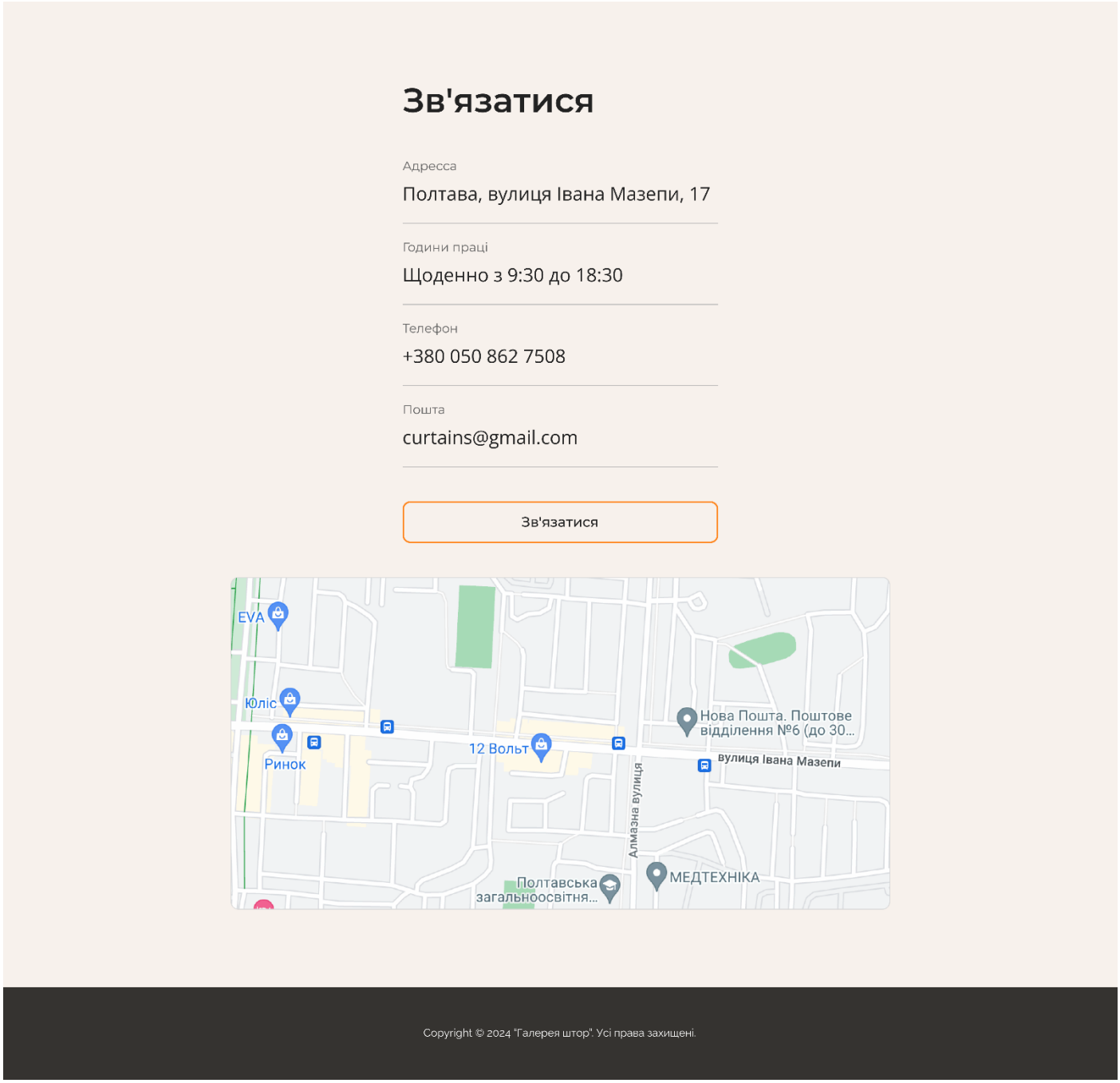


Рисунок 4.5 – Розділ для зв’язку з магазином

Після натискання на кнопку "Зв'язатися" відкриється форма для зв'язку із магазином.

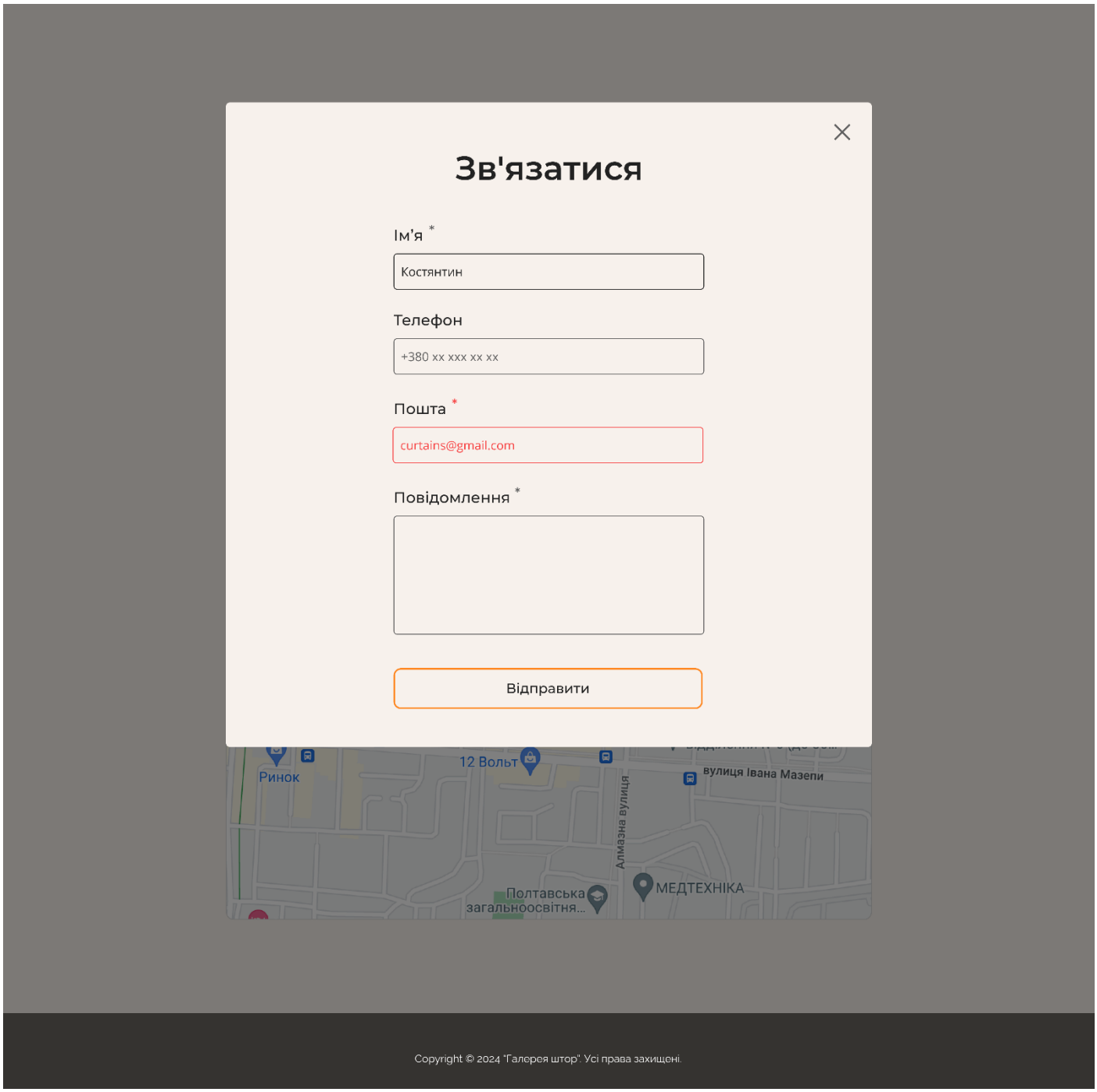


Рисунок 4.6 – Форма зв’язку

Проектуємо наступну сторінку каталог (рис. 4.7). На цій сторінці продемонстровані всі товари магазину які поділені на групи.

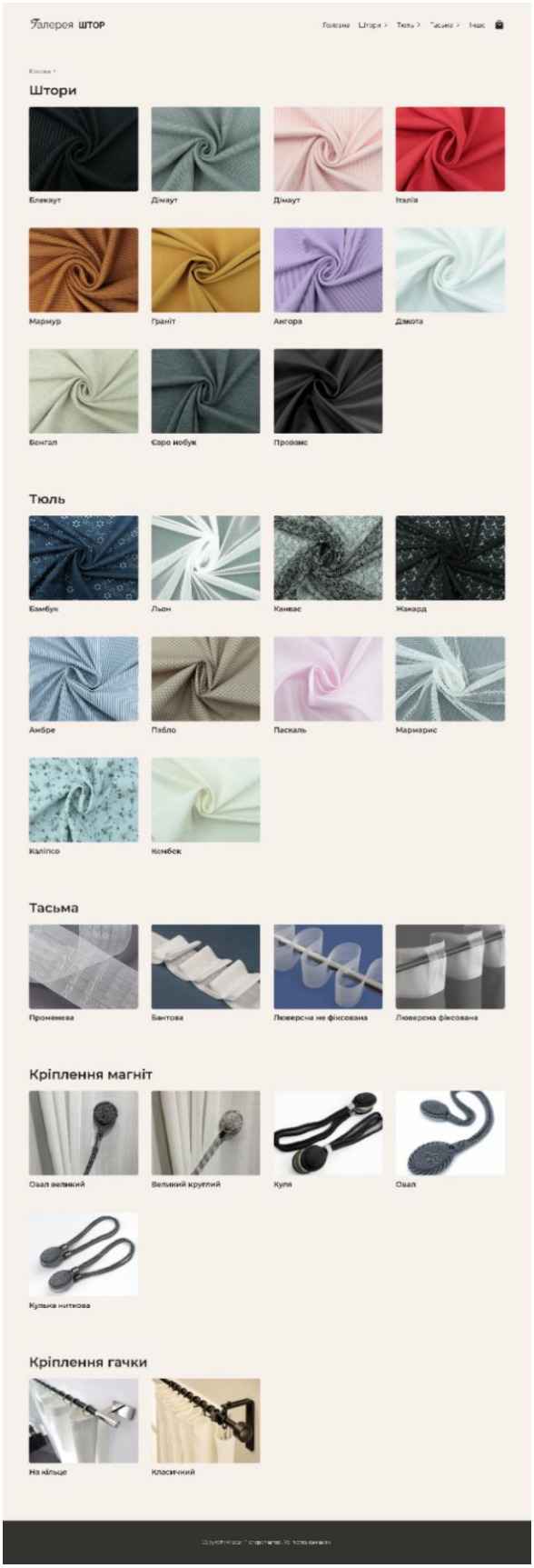


Рисунок 4.7 – Каталог

Проектуємо сторінку підкаталогу (рис. 4.8). Після того, як користувач вибере потрібний продукт з категорії і визначить наприклад тип штор, він перейде на сторінку в підкаталог, що містить відповідний продукт. Тут він зможе більш детально ознайомитися з асортиментом.

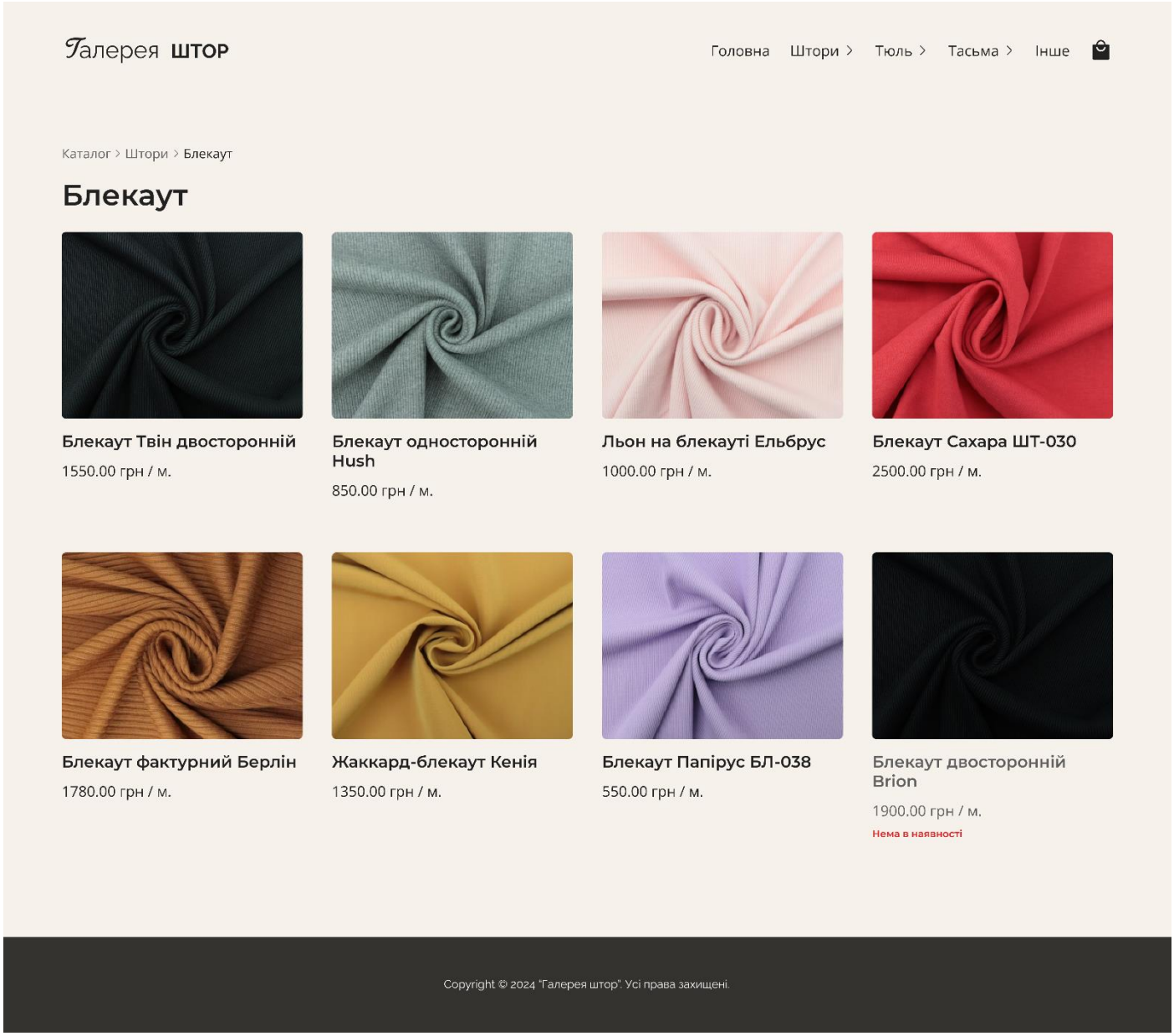


Рисунок 4.8 – Підкаталог

Проектуємо сторінку товару (рис. 4.8). На цій сторінці користувачі можуть знайти фотографії товару, ціну, доступні кольори, кнопку "Додати в кошик" та характеристики товару. Це дозволяє легко оцінити та вибрати відповідний продукт перед покупкою. Після натискання "Додати в кошик" з'явиться вікно "Додано в кошик" (рис. 4.9).

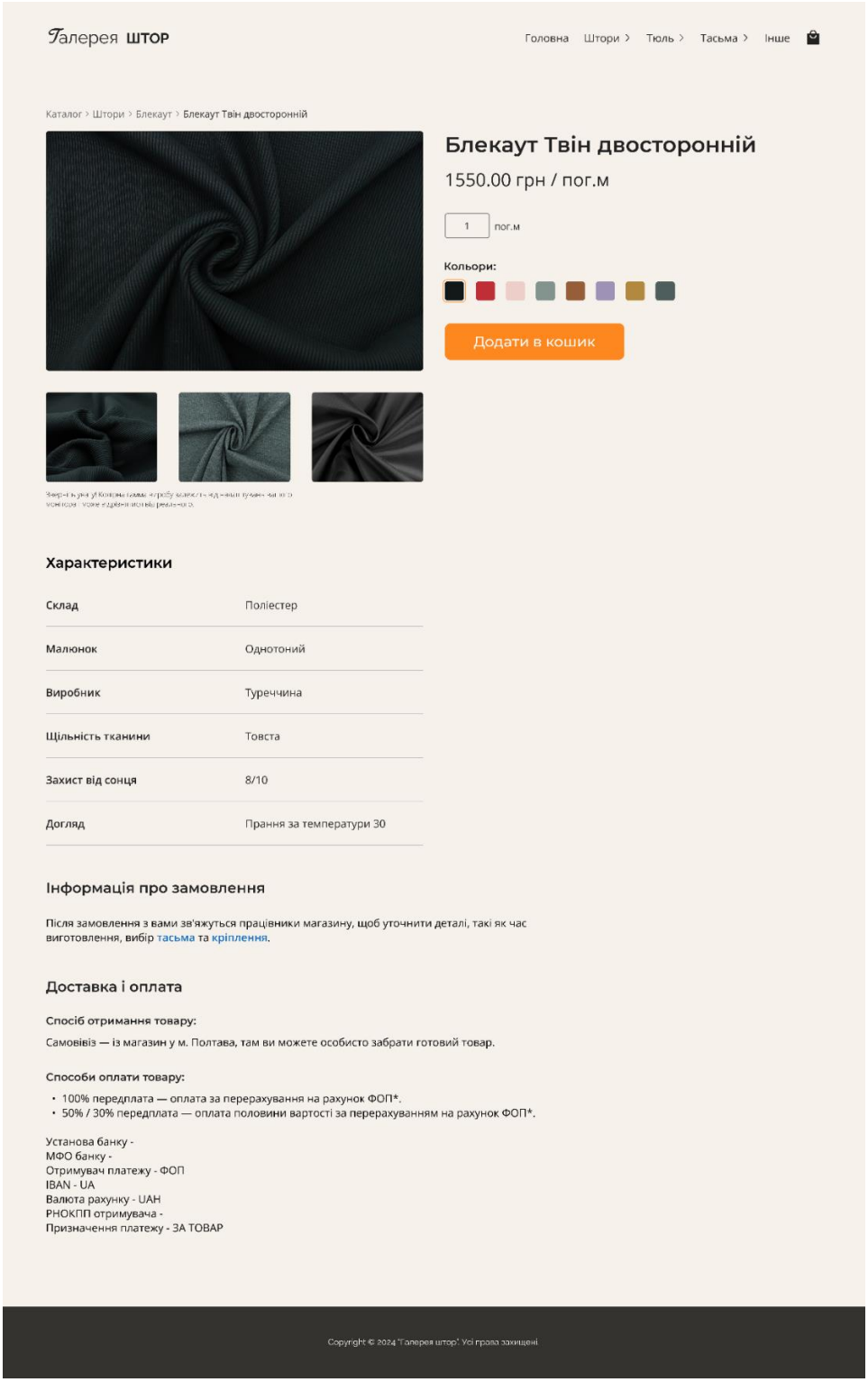


Рисунок 4.8 – Сторінка товару

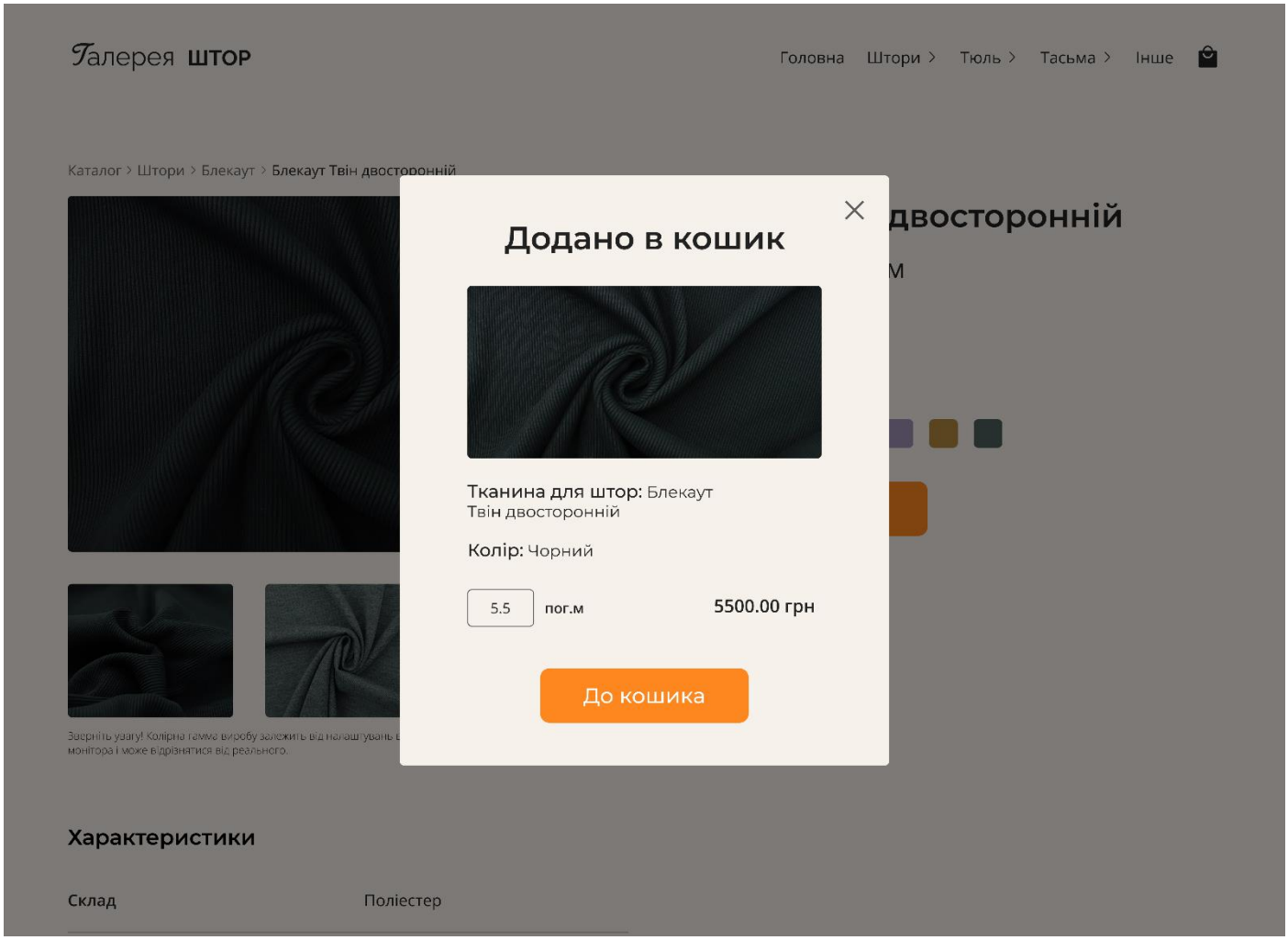


Рисунок 4.9 – Додано в кошик

Проектуємо сторінку кошик (рис. 4.10). Тут фото товару, назва, обраний колір, кількість погонних метрів було обрано користувачем, ціна та можливість видалити товар з кошика.

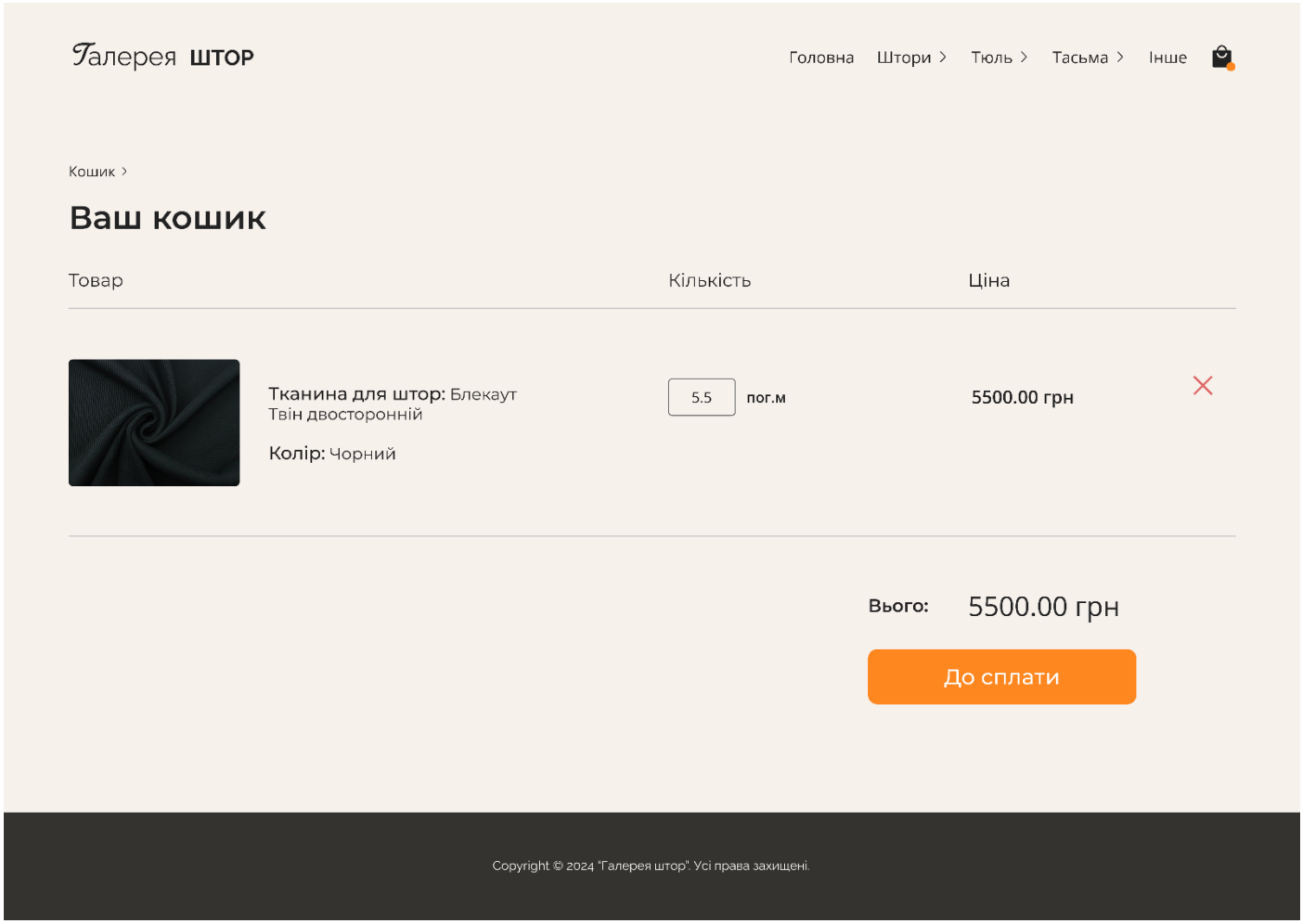


Рисунок 4.10 – Кошик

Проектуємо сторінку оформити замовлення (рис. 4.11). Тут в нас інформація про користувача, вибір передплати, доставка та інтерактивна мапа, те що в кошику та підсумок замовлення.

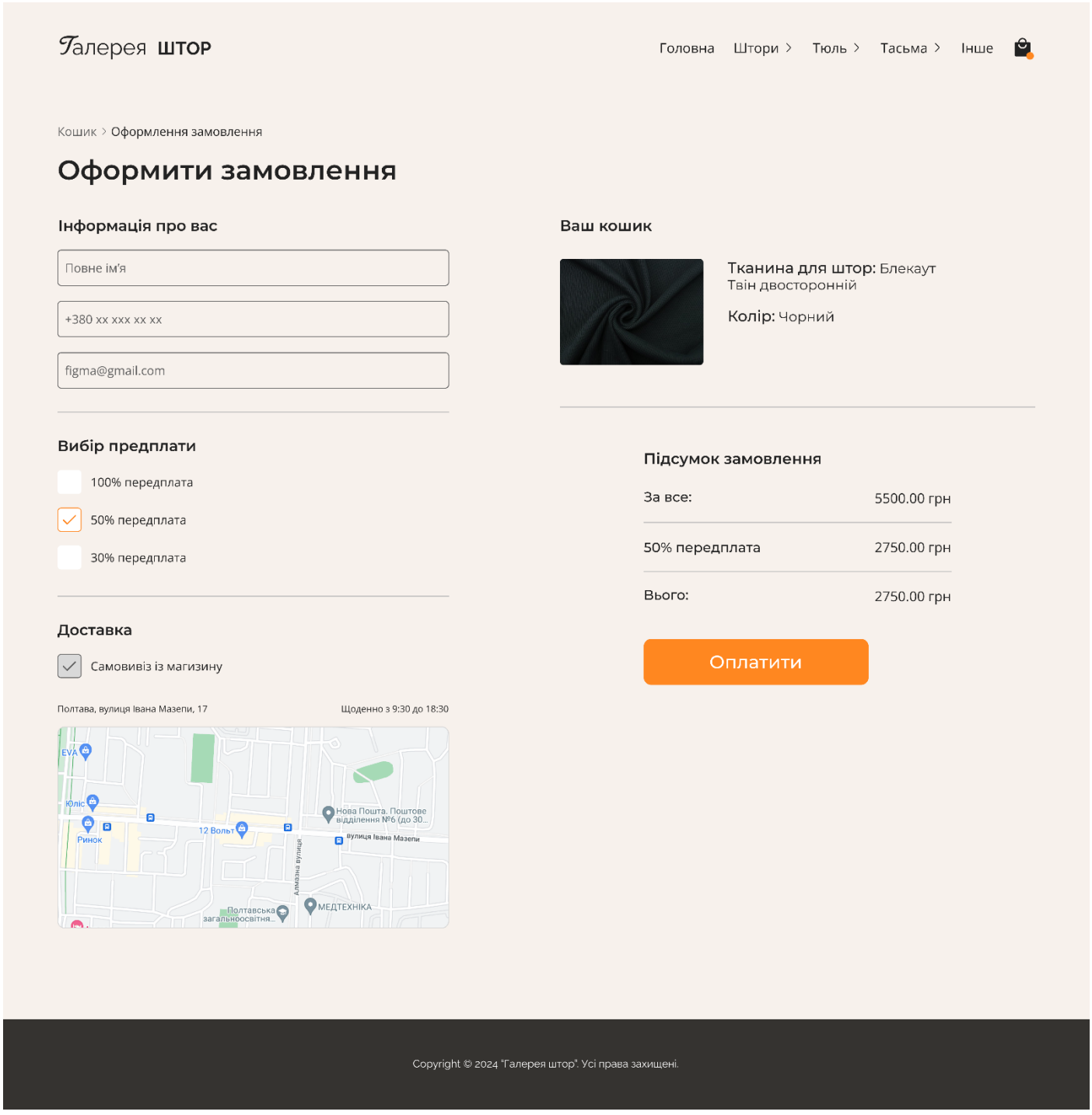


Рисунок 4.11 – Оформлення замовлення

Зараз сайтам важливо мати адаптивний дизайн, щоб користувачи змогли зайти на сайт з любого гаджета. Проектуємо це для всіх сторінок сайту, щоб вони виглядали добре на різних екранах: 1440px, 1000px, 760px (рис. 4.12), 480px (рис. 4.13) та 350px.

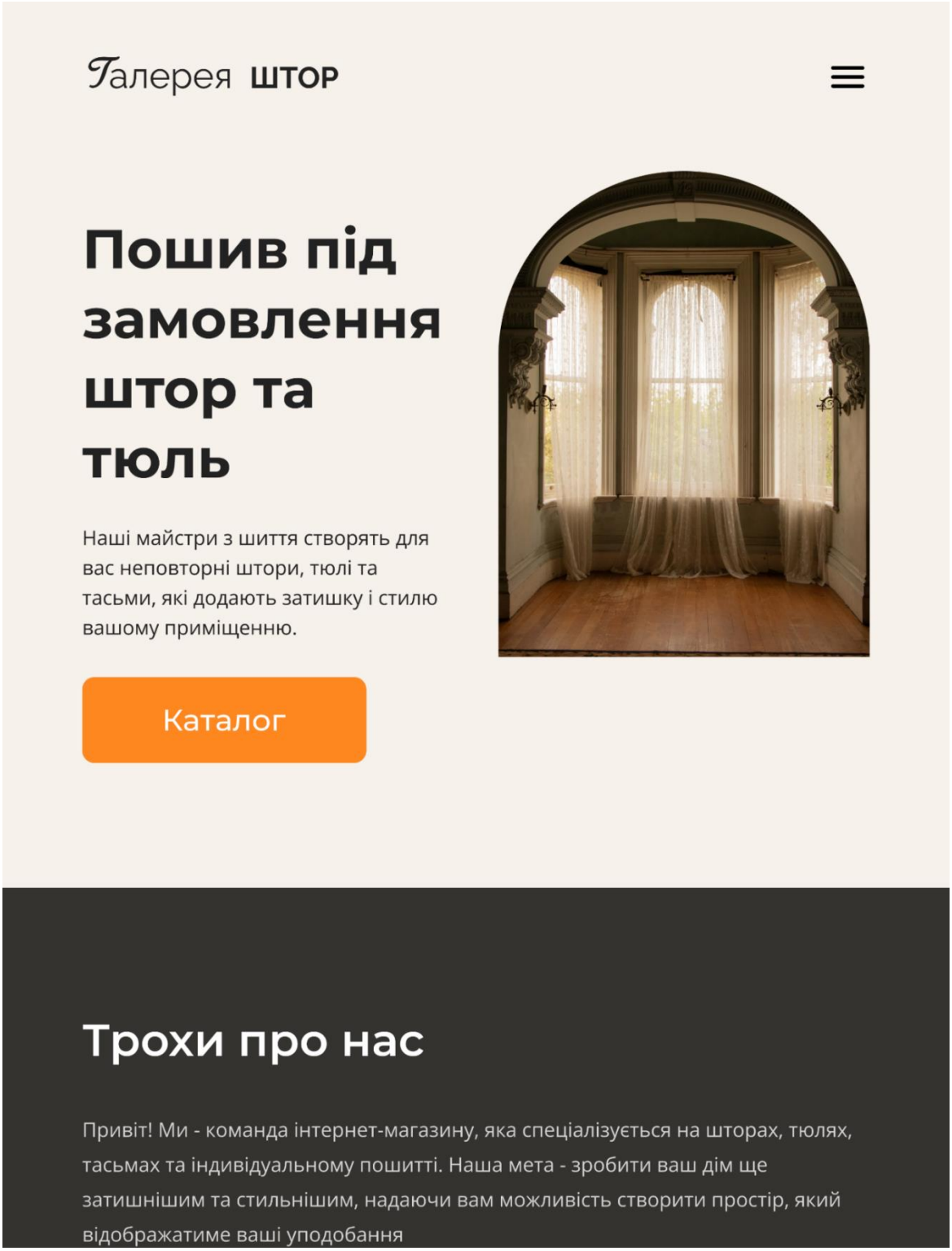


Рисунок 4.12 – Адаптив для екрану в 760px

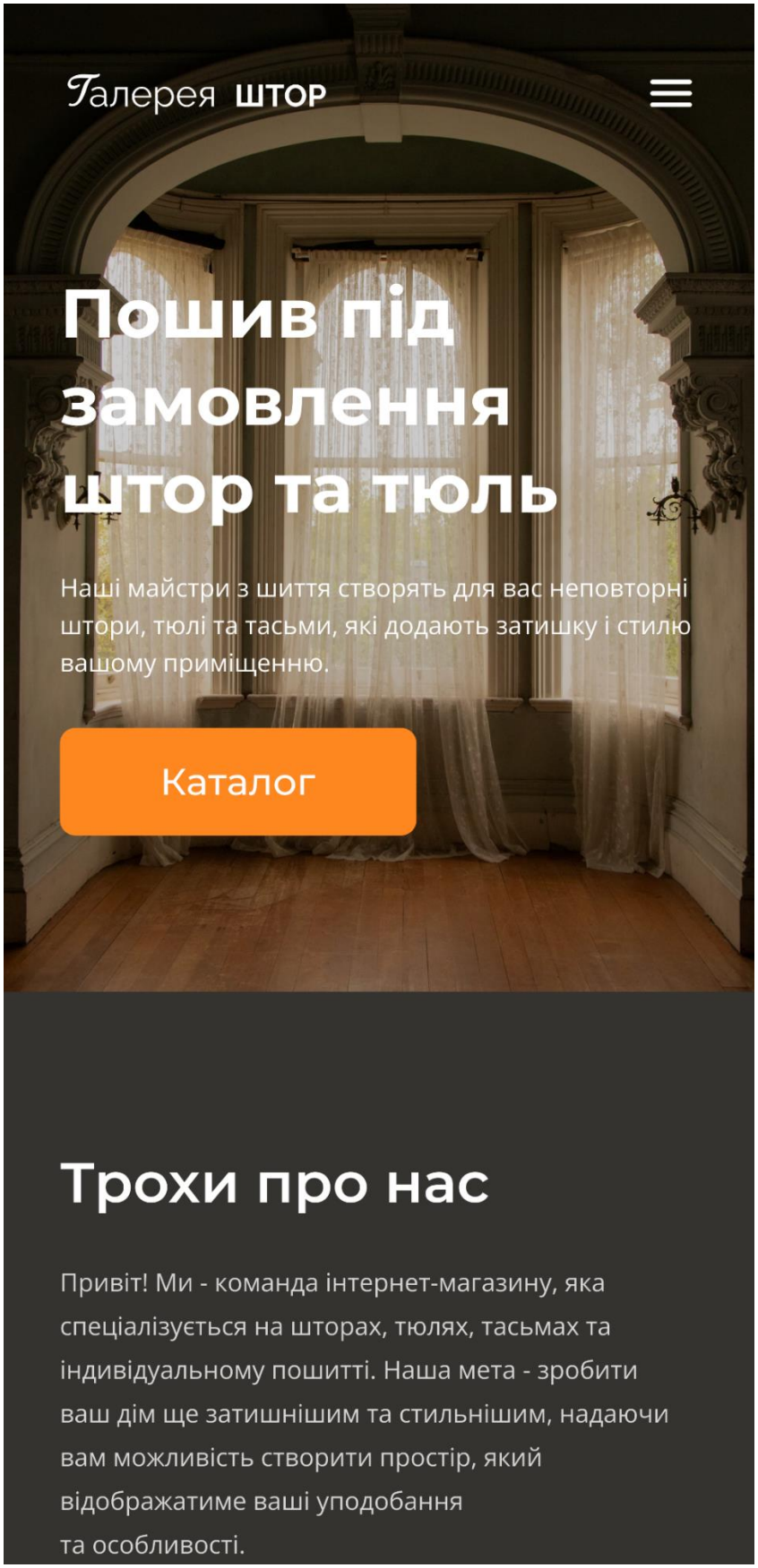


Рисунок 4.13 – Адаптив для екрану в 480px

Далі створимо базу даних.

Щоб працювати із базою даних та мовою PHP нам потрібен локальний сервер. Для цього потрібно встановити програму XAMPP. Запускаємо і біля Apache та MySQL натискаємо Start (рис. 4.14). Проект потрібно зберігати в папці: C:\xampp\htdocs\Curtain-Gallery-main.

Сам сайт буде доступний за адресою: http://localhost/Curtain-Gallery-main

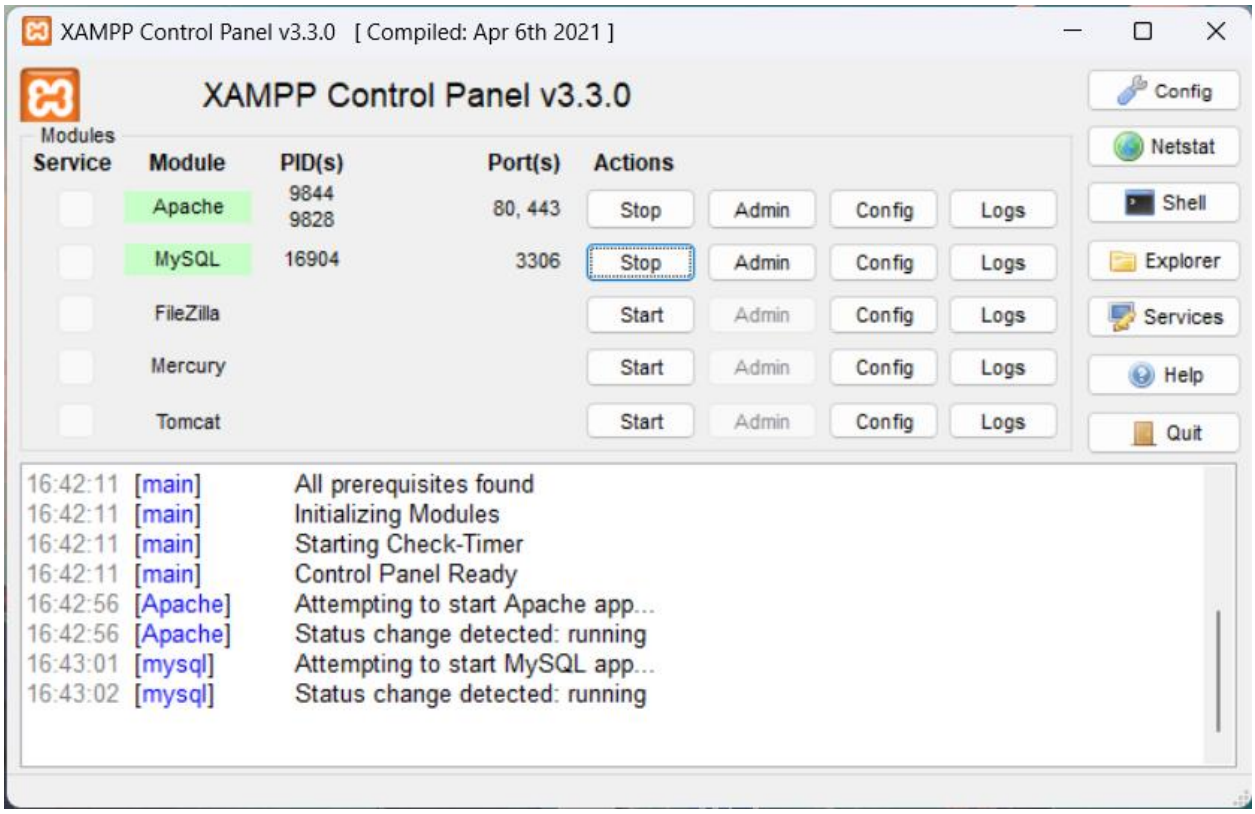


Рисунок 4.14 – Програма XAMPP

Потім потрібно зайти в phpMyAdmin. В XAMPP біля MySQL натиснути кнопку Admin.

Це андмін-панель база даних SQL (рис. 4.15).

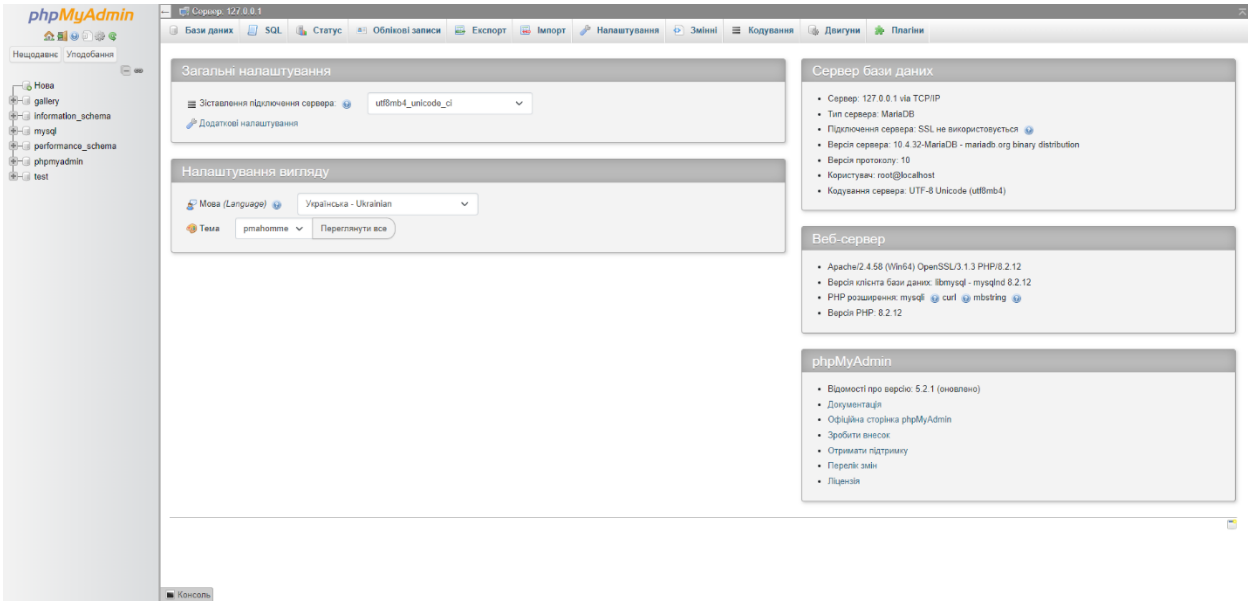


Рисунок 4.15 – Адмін панель БД SQL

Тепер створюємо базу даних. Її можно зробити двома способами.

- Вручну через панель phpMyAdmin.
- Або через SQL запити.

Будем робити через SQL запити.

Створюємо базу даних:

```
CREATE DATABASE IF NOT EXISTS gallery;
```

Тепер створюємо таблиці для зберігання даних: каталог, підкаталог і товар.

```
CREATE TABLE IF NOT EXISTS catalog (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(30) NOT NULL
);
CREATE TABLE IF NOT EXISTS subcatalog (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(30) NOT NULL,
    image VARCHAR(50) NOT NULL UNIQUE,
    catalog_id INT NOT NULL,
    FOREIGN KEY (catalog_id) REFERENCES catalog(id)
);
```

```
CREATE TABLE IF NOT EXISTS products (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(30) NOT NULL,
    image VARCHAR(50) NOT NULL UNIQUE,
    image_s1 VARCHAR(50) NOT NULL UNIQUE,
    image_s2 VARCHAR(50) NOT NULL UNIQUE,
    image_s3 VARCHAR(50) NOT NULL UNIQUE,
    price DECIMAL(10, 2),
    composition VARCHAR(40),
    figure VARCHAR(40),
    manufacturer VARCHAR(40),
    density VARCHAR(40),
    protection VARCHAR(40),
    care VARCHAR(40),
    colors TEXT,
    is_exist BOOLEAN,
    subcatalog_id INT NOT NULL,
    FOREIGN KEY (subcatalog_id) REFERENCES subcatalog(id)
);
```

- **CREATE TABLE** — це ключове слово, яке використовується для створення таблиці.
- **Catalog, subcatalog, products** — це імена таблиць
- **INT** — це тип даних, який буде використовуватися для зберігання ідентифікаторів
- **VARCHAR(50)** — Це тип даних для зберігання рядків обраною (50) довжиною.
- **DECIMAL(10.2)** — Це тип даних для зберігання чисел з фіксованою точністю.
- **TINYINT(1)** — Значення **1** означає, що продукт є в наявності, а значення **0** означає, що продукт відсутній.

Далі можна починати створювати сам сайт.

Перше що потрібно зробити це створити файл (db_manager) для підключення до бази даних (рис. 4. 16):

```
private $servername = "localhost";
private $username = "root";
private $password = "";
private $dbname = "gallery";
private $port = 3306;
private $conn;
```

Рисунок 4.16 – Підключення до БД

У кожному PHP коді буде строка для підключення до БД: require_once("../db/db_manager.php");

Далі створюємо головну сторінку.

Створюємо html сторінку, потім підключаємо стилі css, скрипти js та шрифтові пари. В кожні html сторінці використовуємо базові теги такі як: <html>, <head>, <body> та інші.

- Підключення стилів: <link rel="stylesheet" href="./css/index.css" />
- Підключення скриптів: <script src="./js/cart.js"></script>

Створюємо головну html сторінку index.html (рис. 4.17)

```
<!DOCTYPE html>
<html lang="ua">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Головна сторінка</title>
    <link rel="stylesheet" href="./css/index.css" />
    <link rel="stylesheet" href="./css/global.css" />
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
    <link
      href="https://fonts.googleapis.com/css2?family=Raleway:ital,wght@0,100..900;1,100..900&display=swap"
      rel="stylesheet"
    />
    <link
      href="https://fonts.googleapis.com/css2?family=Open+Sans:ital,wght@0,300..800;1,300..800&family=Raleway:ital,wght@0,100..900;1,100..900&display=swap"
      rel="stylesheet"
    />
    <link
      href="https://fonts.googleapis.com/css2?family=Montserrat:ital,wght@0,100..900;1,100..900&display=swap"
      rel="stylesheet"
    />
  </head>
```

Рисунок 4.17 – Фрагмент html коду головної сторінки

Перший екран сайту (рис 4.18): Матиме вигляд (рис 4.1).

```
<div class="lv1 container">
  <div>
    <h1>Пошив під замовлення штор та тюль</h1>
    <p>
      Наші майстри з шиття створять для вас неповторні штори, тюлі та
      тасьми, які додають затишку і стилю вашому приміщенню.
    </p>
    <a href="/catalog"><button class="button-bg">Каталог</button></a>
  </div>
  <img
    src="./assets/images/home/top.png"
    alt="top-img"
    width="519"
    height="680"
  />
</div>
```

Рисунок 4.18 – Фрагмент html коду головної сторінки першого блоку

Також робимо **навігацію** (меню) сайту (рис. 4.19) і одразу бургер меню (рис. 4.20) для адаптиву:

```
<nav class="h_nav align_center">
  <a href="/">Головна</a>
  <div class="h_select">
    <div class="align_center">
      <span>Штори</span>
      <img
        src="./assets/images/header/arrow.svg" alt="arrow" height="12" width="6" id="arrow_s1" class="arrow_ss"/>
    </div>
    <div id="catalog_s1" class="burger_th"></div>
  </div>
  <div class="h_select">
    <div class="align_center">
      <span>Тюль</span>
      <img
        src="./assets/images/header/arrow.svg" alt="arrow" height="12" width="6" id="arrow_s2" class="arrow_ss"/>
    </div>
    <div id="catalog_s2" class="burger_th"></div>
  </div>
  <div class="h_select">
    <div class="align_center">
      <span>Тасьма</span>
      <img
        src="./assets/images/header/arrow.svg" alt="arrow" height="12" width="6" id="arrow_s3" class="arrow_ss"/>
    </div>
    <div id="catalog_s3" class="burger_th"></div>
  </div>
  <a href="/other">Інше</a>
  <a href="/cart" style="display: flex; position: relative" id="cart-btn">
    <img
      src="./assets/images/header/cart.svg" alt="cart" height="21" width="24"/>
  </a>
</nav>
```

Рисунок 4.19 – Фрагмент html коду меню

```
<div class="h_burger">
  <img
    src="./assets/images/header/burger.svg" alt="burger" width="26" height="17" onclick="showBurger()"/>
  <div class="mob_burg">
    <img
      src="./assets/images/home/close.svg" alt="close" width="21" height="21" onclick="closeBurger()"/>
    <div>
      <a href="/" style="margin-bottom: 10px">Головна</a>
      <div class="hm_nav">
        <div class="align_center" onclick="showMobCatalog(1)">
          <div class="title-hm">
            <span>Штори</span>
            <img
              src="./assets/images/header/arrow.svg" alt="arrow" height="12" width="6" id="arrow_s1" class="arrow_ss"/>
          </div>
        </div>
        <div id="catalog_sm1"></div>
      </div>
      <div class="hm_nav">
        <div class="align_center" onclick="showMobCatalog(2)">
          <div class="title-hm">
            <span>Тюль</span>
            <img
              src="./assets/images/header/arrow.svg" alt="arrow" height="12" width="6" id="arrow_s1" class="arrow_ss"/>
          </div>
        </div>
        <div id="catalog_sm2"></div>
      </div>
      <div class="hm_nav">
        <div class="align_center" onclick="showMobCatalog(3)">
          <div class="title-hm">
            <span>Тасьма</span>
            <img
              src="./assets/images/header/arrow.svg" alt="arrow" height="12" width="6" id="arrow_s1" class="arrow_ss"/>
          </div>
        </div>
        <div id="catalog_sm3"></div>
      </div>
      <a href="/catalog">Інше</a>
    </div>
  </div>
</div>
```

Рисунок 4.20 – Фрагмент html коду бургер меню

На головному сайті є слайдер із фотографіями клієнтів, зробимо його одразу в файлі index.html за допомогою JS:

```
<script>
  let countImage = 1;
  let index = 1;
  const screenWidth = window.innerWidth;
  let listImage = [];
```

Рисунок 4.21 – Фрагмент JS скрипта ініціалізація змінних

- **countImage** - кількість зображень у слайдері.
- **index** - індекс поточного зображення.
- **screenWidth** - ширина вікна браузера.

- **listImage** - масив імен зображень.

Функція для визначення кількості зображень на сторінці (Рис. 4.22). В залежності від ширини екрана визначається кількість зображень, що показуються на сторінці.

```
function calculateImagesToShow() {
  if (screenWidth > 1000) {
    return 6;
  } else if (screenWidth > 760) {
    return 4;
  } else {
    return 1;
  }
}
```

Рисунок 4.22 – Фрагмент JS скрипта

Завантаження зображень з сервера (рис. 4.23):

```
fetch("./server/slider/slider.php", {
  method: "GET",
  headers: {
    "Content-Type": "application/json",
  },
})
  .then((response) => response.json())
  .then((images) => {
    listImage = images;
    countImage = images.length;
    const imageGallery = document.getElementById("imagesGallery");
    const listPag = document.getElementsByClassName("list");

    const count = calculateImagesToShow();
    let pageCount = Math.floor(countImage / count);

    if (pageCount <= 1) {
      document.getElementById("arrow-left").disabled = true;
      document.getElementById("arrow-right").disabled = true;
    }

    if (count === 1 && pageCount > 7) pageCount = 7;

    for (let i = 1; i <= pageCount; i++) {
      const div = document.createElement("div");
      div.id = `page-${i}`;
      if (i === 1) div.className = "active";
      listPag[0].appendChild(div);
    }

    images.forEach((image, i) => {
      if (i < count) {
        const imageName = image.split(".")[0];
        const img = document.createElement("img");
        img.src = `./assets/images/home/list/${image}`;
        img.alt = imageName;
        img.id = `img-${imageName}`;
        imageGallery.appendChild(img);
      }
    });
  })
  .catch((error) => console.error("Помилка завантаження зображень:", error));
```

Рисунок 4.23 – Фрагмент JS скрипта

- Завантажуються імена зображень з сервера.
- Визначається кількість зображень для показу на сторінці.
- Генеруються елементи пагінації.
- Додаються перші зображення до галереї.

Функція для зміни слайдів (рис. 4.24)

```
function changeSlide(direction) {
  const imageGallery = document.getElementById("imagesGallery");
  const count = calculateImagesToShow();
  const pageCount = Math.ceil(countImage / count);

  let newIndex;

  if (direction === "prev") {
    newIndex = index - count;
    if (newIndex < 1) {
      newIndex += countImage;
    }
  } else if (direction === "next") {
    newIndex = index + count;
    if (newIndex > countImage) {
      newIndex -= countImage;
    }
  }

  index = newIndex;

  const paginationBlocks = document.querySelectorAll(".list > div");
  paginationBlocks.forEach((block) => block.classList.remove("active"));

  const currentPage = Math.ceil(index / count);
  const activePage = document.getElementById(`page-${currentPage}`);
  if (activePage) {
    activePage.classList.add("active");
  }

  imageGallery.innerHTML = "";

  for (let i = index; i < index + count; i++) {
    const imgIndex = i > countImage ? i - countImage : i;
    const imageName = listImage[imgIndex - 1].split(".")[0];
    const img = document.createElement("img");
    img.src = `./assets/images/home/list/${listImage[imgIndex - 1]}`;
    img.alt = imageName;
    img.id = `img-${imageName}`;
    imageGallery.appendChild(img);
  }
}
```

Рисунок 4.24 – Фрагмент JS скрипта

- В залежності від напрямку (**prev** або **next**), змінюється індекс поточного зображення.
- Оновлюється активна сторінка пагінації.
- Очищується галерея і додаються нові зображення.

Функції для переходу на попередній/наступний слайд (рис. 4.25)

```
// Prev Slider Page
function prevSlide() {
  changeSlide("prev");
}

// Next Slider Page
function nextSlide() {
  changeSlide("next");
}
</script>
```

Рисунок 4.25 – Фрагмент JS скрипта

Для слайдера також необхідно створити php код який буде отримувати список доступних зображень з серверу. (рис. 4.26).

```
<?php
if ($_SERVER["REQUEST_METHOD"] == "GET") {
    echo json_encode(array_map('basename', glob("../../assets/images/home/list/*")));
}
```

Рисунок 4.26 – Фрагмент PHP скрипта

Ось так чином виглядають стилі для всього нашого сайту, буде декілька файлів css: index.css, cart.css, catalog.css, global.css (рис. 4.27), order.css, product.css, subcatalog.css

```css
.lv1 {
  margin-top: 115px;
  display: flex;
  justify-content: space-between;
  align-items: flex-start;
  gap: 30px;
  padding-bottom: 155px;
}

.lv1 h1 {
  font-family: Montserrat;
  font-size: 56px;
  font-weight: 700;
  line-height: 70px;
}

.lv1 p {
  font-family: Open Sans;
  font-size: 24px;
  font-weight: 400;
  line-height: 36px;
  margin-bottom: 44px;
}

.bg-top {
  display: none;
  background-image: url("../assets/images/home/top2.png");
  position: absolute;
  z-index: -1;
  top: 0;
  left: 0;
  width: 100%;
  height: 752px;
  background-size: cover;
  background-position: center;
  background-repeat: no-repeat;
}

.lv2,
.lv4 {
  padding-block: 160px;
  background: #353330;
}

.lv2 h2,
.lv4 h2 {
  color: #ffffff;
  font-family: Montserrat;
  font-size: 44px;
```

Рисунок 4.27 – Фрагмент CSS коду

Таким чином робимо всі інші сторінки css: каталог, підкаталог, сторінка товару, кошик, оформити замовлення та адаптив.

Далі створюємо php файл для каталогу: get_subcatalog.php (рис. 4.28)

Цей код обробляє запит, отримує дані про підкаталоги з бази даних і повертає їх у форматі JSON. Якщо даних немає або виникла помилка, він повідомляє про це.

```php
<?php
// Підключення необхідних файлів
require_once("../db/db_manager.php");

// Перевірка типу HTTP-запиту
if ($_SERVER["REQUEST_METHOD"] == "GET") {
    // Створення підключення до бази даних
    $databaseManager = new DatabaseManager();

    // Виконання запиту до бази даних
    $result = $databaseManager->sendQuery("SELECT sc.id, sc.name, sc.image, c.name AS catalog_name
                                           FROM subcatalog AS sc
                                           JOIN catalog AS c ON sc.catalog_id = c.id");

    // Закриття підключення до бази даних
    unset($databaseManager);

    // Обробка результату запиту
    if ($result) {
        if ($result->num_rows > 0) {
            $rows = $result->fetch_all(MYSQLI_ASSOC);
            http_response_code(200);
            echo json_encode($rows);
        } else {
            http_response_code(204);
            echo json_encode(["message" => "No records found"]);
        }
    } else {
        http_response_code(500);
        echo json_encode(["message" => "Database error"]);
    }
}
```

Рисунок 4.28 – PHP скрипт

- Перевіряється чи визначає, чи це HTTP GET-запит.
- Підключається до бази даних: Створює з'єднання з базою даних.
- Виконує SQL-запит**:** Відправляє запит до бази даних для отримання інформації про підкаталоги та відповідні каталоги.
- Обробляє результат запиту: Перевіряє, чи отримані дані, і повертає їх у форматі JSON, якщо дані знайдено. Якщо дані відсутні, повертає повідомлення про це.

Якщо виникла помилка під час виконання запиту, повертає повідомлення про помилку.

- Закриває з'єднання з базою даних: Закриває з'єднання після завершення операцій з базою даних.

Наступний php файл для підкаталогу: get_products.php (рис. 4.29).

Цей PHP код обробляє GET запит, отримує параметр **id**, виконує два SQL запити до бази даних для отримання інформації про продукти та їх підкаталоги. Потім він повертає результати у форматі JSON. Якщо дані не знайдено, повертається повідомлення з кодом 204.

- Перевіряє метод HTTP запиту: Перевіряється, чи метод запиту є HTTP GET.
- Отримує параметр id з GET запиту: Витягує значення параметра id з URL-адреси.
- Створює підключення до бази даних: Встановлює з'єднання з базою даних.
- Виконує SQL запит для отримання продуктів: Виконує запит до бази даних для отримання інформації про продукти, які належать до підкаталогу з вказаним id.
- Виконує SQL запит для отримання інформації про підкаталог: Виконує другий запит до бази даних для отримання інформації про сам підкаталог.
- Обробляє результати запитів**:** Перевіряє, чи є результати обох запитів та формує відповідь у форматі JSON з отриманими даними. Якщо жоден із запитів не повернув результатів, повертається відповідне повідомлення.
- Закриває підключення до бази даних: Закриває з'єднання з базою даних для ефективного використання ресурсів.

```php
<?php
// Підключення необхідних файлів
require_once("../db/db_manager.php");

// Перевірка методу запиту
if ($_SERVER["REQUEST_METHOD"] == "GET") {
    // Отримання параметра id з GET запиту
    $subcategory = intval($_GET['id']);

    // Створення підключення до бази даних
    $databaseManager = new DatabaseManager();

    // Виконання SQL запиту для отримання продуктів за sub-catalogs
    $result = $databaseManager->sendQuery("SELECT p.id, p.name, p.image, p.price, p.is_exist
                                           FROM products AS p
                                           WHERE p.subcatalog_id = $subcategory");

    // Виконання SQL запиту для отримання інформації про підкаталог та каталог
    $result_i = $databaseManager->sendQuery("SELECT sc.name AS subcatalog_name, c.name AS catalog_name
                                           FROM subcatalog AS sc
                                           JOIN catalog AS c ON sc.catalog_id = c.id
                                           WHERE sc.id = $subcategory
                                           LIMIT 1");

    // Закриття підключення до бази даних
    unset($databaseManager);

    // Обробка результатів запитів
    if ($result->num_rows > 0 && $result_i->num_rows > 0) {
        $rows = $result->fetch_all(MYSQLI_ASSOC);
        $rows_i = $result_i->fetch_all(MYSQLI_ASSOC)[0];
        http_response_code(200);
        echo json_encode(["rows" => $rows, "rows_i" => $rows_i]);
    } elseif ($result->num_rows > 0) {
        $rows = $result->fetch_all(MYSQLI_ASSOC);
        http_response_code(200);
        echo json_encode(["rows" => $rows]);
    } elseif ($result_i->num_rows > 0) {
        $rows_i = $result_i->fetch_all(MYSQLI_ASSOC)[0];
        http_response_code(200);
        echo json_encode(["rows_i" => $rows_i]);
    } else {
        http_response_code(204);
        echo json_encode(["message" => "No records found"]);
    }
}
```

Рисунок 4.29 – PHP скрипт

Наступний php файл для сторінки товару: get_product.php (Рис. 4.30)

Цей код обробляє GET-запит, отримує інформацію про продукт з бази даних за його ідентифікатором і повертає цю інформацію у форматі JSON.

```php
<?php
// Підключення db_manager
require_once("../db/db_manager.php");

// Перевірка методу запиту
if ($_SERVER["REQUEST_METHOD"] == "GET") {
    // Отримання id продукту
    $product = $_GET['id'];

    // Створення з'єднання з базою даних
    $databaseManager = new DatabaseManager();

    // Виконання SQL-запиту для отримання інформації про продукт
    $result = $databaseManager->sendQuery("SELECT p.*
                                           FROM products AS p
                                           WHERE p.id = $product
                                           LIMIT 1");

    // Виконання SQL-запиту для отримання шляху до сторінок
    $result_i = $databaseManager->sendQuery("SELECT sc.name AS subcatalog_name, c.name AS catalog_name, sc.id AS sc_id
                                           FROM products AS p
                                           JOIN subcatalog AS sc ON p.subcatalog_id = sc.id
                                           JOIN catalog AS c ON sc.catalog_id = c.id
                                           WHERE p.id = $product
                                           LIMIT 1");

    // Від'єднатися з базою даних
    unset($databaseManager);

    // Обробка результатів запиту
    if ($result && $result_i) {
        if ($result->num_rows > 0 && $result_i->num_rows > 0) {
            $rows = $result->fetch_all(MYSQLI_ASSOC)[0];
            $rows_i = $result_i->fetch_all(MYSQLI_ASSOC)[0];
            http_response_code(200);
            echo json_encode(array("rows_i" => $rows_i, "rows" => $rows));
        } else {
            http_response_code(204);
            echo json_encode(["message" => "No records found"]);
        }
    } else {
        http_response_code(500);
        echo json_encode(["message" => "Database error"]);
    }
}
```

Рисунок 4.30 – PHPскрипт

- Перевіряється, чи обидва запити виконані успішно.
- Якщо обидва запити повертають результати, дані з результатах запитів конвертуються у асоціативні масиви та надсилаються у відповідь як JSON використовуючи код стану 200 як відповідь.
- Якщо дані не були повернуті в жодному із запитів, буде надіслано відповідь із кодом 204 та повідомленням "запис не знайдено"
- Якщо під час виконання запиту виникає помилка, надсилається відповідь із кодом 500 та повідомленням "помилка бази даних".

Далі робимо можливість користувачу додати товар у кошик.

Створюємо JS файл: cart.js (рис. 4.31).

```
// Додати в кошик
updateCartData = (newData) => {
  const existingData = getCartData();
  const updatedData = [...existingData, newData];
  localStorage.setItem("cart", JSON.stringify(updatedData));

  let cart_ex = document.getElementById("cart-btn");
  if (!cart_ex.classList.contains("exist")) cart_ex.classList.add("exist");
};

// Видалити товар із кошика
removeCartItem = (itemId) => {
  const existingData = getCartData();
  const updatedData = existingData.filter((item) => item.id !== itemId);
  localStorage.setItem("cart", JSON.stringify(updatedData));

  if (updatedData.length === 0) {
    let cart_ex = document.getElementById("cart-btn");
    cart_ex.classList.remove("exist");
  }
};

// Перевірка наявності товару в кошику
isCartEmpty = () => {
  const cartData = getCartData();
  return cartData.length === 0;
};

// Отримує всі елементи з кошика
getCartData = () => {
  const cartData = localStorage.getItem("cart");
  return cartData ? JSON.parse(cartData) : [];
};

// Перевірка наявності елементів у кошику при завантаженні сторінки
if (!isCartEmpty()) document.getElementById("cart-btn").classList.add("exist");
```

Рисунок 4.31 – JS скрипт

Цей код дозволяє додавати та видаляти елементи до кошика, зберігати та отримувати дані з локального сховища браузера, а також переглядати стан кошика (порожній він чи ні).

Ще потрібен код, який буде завантажувати підкаталоги з сервера, групувати їх за основними каталогами і відображати на веб-сторінці у відповідних розділах.

Створюємо файл: header.js (рис. 4.32).

```
// Отримання підкаталогу з сервера
fetch("./server/catalog/get_subcatalog.php", {
  method: "GET",
  headers: {
    "Content-Type": "application/json",
  },})
  .then((response) => response.json())
  .then((res) => {
    // Групування підкаталогів за каталогами
    const groupedData = res.reduce((acc, curr) => {
      const key = curr.catalog_name;
      if (!acc[key]) {
        acc[key] = [];
      }
      acc[key].push(curr);
      return acc;
    }, {});
    // Отримати дані html-документа
    let catalog_s1 = document.getElementById("catalog_s1");
    let catalog_s2 = document.getElementById("catalog_s2");
    let catalog_s3 = document.getElementById("catalog_s3");
    let catalog_sm1 = document.getElementById("catalog_sm1");
    let catalog_sm2 = document.getElementById("catalog_sm2");
    let catalog_sm3 = document.getElementById("catalog_sm3");
    // Відобразити всі підкаталоги
    // Отримати назву каталогу та всі підкаталоги, прив'язані до нього:
    Object.entries(groupedData).forEach(([catalogName, items]) => {
      // Отримання підкаталогів і рендеринг в документ каталогу:
      items.forEach((element) => {
        let link = document.createElement("a");
        link.href = `/subcatalog?id=${element.id}`;
        link.textContent = element.name;
        switch (catalogName) {
          case "Штори":
            catalog_s1.appendChild(link);
            catalog_sm1.appendChild(link.cloneNode(true));
            break;
          case "Тюль":
            catalog_s2.appendChild(link);
            catalog_sm2.appendChild(link.cloneNode(true));
            break;
          case "Тасьма":
            catalog_s3.appendChild(link);
            catalog_sm3.appendChild(link.cloneNode(true));
            break;}
      });
    });
  }).catch((error) => console.error("Помилка загрузки каталога:", error));
```

Рисунок 4.32 – JS скрипт

Далі робимо адаптив. Потрібно щоб на розмірі екрану який менше 760px, ховалося меню у бургер.

Створюємо файл: global.js (рис. 4.33).

```
// Початок показати/сховати бургер меню
function eventBurgerClose(e) {
  if (!e.target.closest(".mob_burg")) closeBurger();
}
function showBurger() {
  document.getElementsByClassName("mob_burg")[0].classList.add("active");

  setTimeout(() => document.addEventListener("click", eventBurgerClose), 500);
}
function closeBurger() {
  document.getElementsByClassName("mob_burg")[0].classList.remove("active");

  document.removeEventListener("click", eventBurgerClose);
}
// Кінець показати/сховати бургер меню

// Увімкнути/вимкнути прокрутку сторінок
function disScroll() {
  document.body.style.overflow = "hidden";
}
function allScroll() {
  document.body.style.overflow = "unset";
}

// Показати/приховати каталог бургер меню
function showMobCatalog(num) {
  let nav = document.getElementsByClassName("hm_nav")[num - 1];
  if (nav.classList.contains("active")) nav.classList.remove("active");
  else nav.classList.add("active");
}
```

Рисунок 4.33 – JS скрипт

Більш детально як він працює:

1. **Показати/сховати бургер меню**
    - **showBurger()**: Додає клас "active" до першого елементу з класом "mob_burg". Це зазвичай відображає або приховує бургер-меню на мобільному пристрої. Після цього встановлюється таймер на 500 мілісекунд для додавання події "click" на весь документ, яка викликає функцію **eventBurgerClose**.

- **eventBurgerClose(e)**: Якщо елемент, на який натиснули, не є бургер-меню або не знаходиться в його дочірніх елементах, то бургер-меню закривається, викликаючи **closeBurger()**.
- **closeBurger()**: Видаляє клас "active" у першого елементу з класом "mob_burg" і видаляє обробник подій, який був доданий раніше.

2. **Увімкнути/вимкнути прокрутку сторінок**
   - **disScroll()**: Встановлює стиль першого елементу **<body>** таким чином, щоб заборонити прокрутку сторінки.
   - **allScroll()**: Скидає стиль першого елементу **<body>**, щоб дозволити прокрутку сторінки.
3. **Показати/приховати каталог бургер меню**
   - **showMobCatalog(num)**: Дозволяє відображати або приховувати каталог, відповідно до параметру **num**. Цей параметр вказує на позицію каталогу у масиві елементів з класом "hm_nav". Якщо каталог вже має клас "active", то він видаляється, інакше - додається.

4.2 Перевірка та тестування магазину на можливі помилки

Під час тестування веб-сайту та бази даних магазину всі основні функції працювали чудово. Однак, коли перевірявся адаптивний макет, було виявлено деякі помилки. Зокрема, на екрані шириною 760 пікселів було порушено макет, що вплинуло на відображення сторінки та загальну доступність сайту. Ця проблема була вирішена швидко, тепер сайт відображається правильно на всіх пристроях, незалежно від розміру екрану.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи, був створений інтернет-магазин "Галерея штор". Розробка даного магазину є важливим та актуальним завданням у сучасному світі, оскільки інтернет-магазини стали необхідним інструментом для онлайн покупок для користувачів, а для бізнесу можливість збільшити прибуток та потік нових клієнтів.

Під час створення магазину "Галерея штор" було виконано ряд завдань та досягнуто основні цілі:

1. Аналіз і вивчення існуючих сайтів: Був проведений детальний аналіз функціональності та особливостей інших сайтів, щоб краще зрозуміти їх переваги та недоліки.
2. Проектування структури бази даних та розробка блок схеми сайту: Була розроблена структура бази даних, яка дозволяє зберігати всі товари в магазині та розробка схеми сайту.
3. Проектування інтерфейсу користувача: Був створений інтуїтивний та зручний інтерфейс для взаємодії користувачів з магазином.
4. Використання сучасних технологій: Для розробки сайту використовувалися сучасні технології програмування та інструменти, які забезпечили надійність та ефективність роботи інтернет-магазину.
5. Тестування: Було проведено тестування, щоб переконатися, що сайт відповідає вимогам та очікуванням користувачів. Тести дозволили виявити і усунути різні помилки і недоліки.

Для успішної реалізації проекту магазину "Галерея штор" було використано програму для запуску локального серверу XAMPP, що включає веб-сервер Apache, базу даних SQL та мову програмування PHP. Проектування інтерфейсу було в програмі Figma, створення інтерфейсу користувача виконано за допомогою веб-технологій, таких як HTML для структури, CSS для стилізації та JavaScript для реалізації динамічних елементів. Використання цих технологій дозволило досягти

високого рівня взаємодії з користувачем, а також отримати сучасний і зручний дизайн. Також було використано мову запитів SQL для оптимізації доступу та управління базою даних для підвищення продуктивності та ефективності інтернет магазину "Галерея штор".

Загалом, розробка інтернет-магазину "Галерея штор" є успішною та відповідає поставленим цілям та завданням роботи.

СПИСОК ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Джон Дакетт. "HTML та CSS: Проектування та створення веб-сайтів". / Джон Дакетт - Wiley, 2011 – 490 с.
2. Девід Фланаган. "JavaScript: Вичерпний посібник". / Девід Фланаган - O'Reilly Media, 2021 – 746 с.
3. Ларрі Уллман. "PHP та MySQL для динамічних веб-сайтів". / Ларрі Уллман - Peachpit Press, 2018 – 704 с.
4. Ерік Мейєр та Естель Вейл. "CSS: Вичерпний посібник". / Ерік Мейєр, Естель Вейл - O'Reilly Media, 2021 – 1088 с.
5. Дженніфер Нідерст Роббінс. "Вивчаємо веб-дизайн: Посібник для початківців з HTML, CSS та веб-графіки". / Дженніфер Нідерст Роббінс - O'Reilly Media, 2018 – 810 с.
6. Робін Ніксон. "Вивчаємо PHP, MySQL та JavaScript". / Робін Ніксон - O'Reilly Media, 2018 – 806 с.
7. Стів Круг. "Не змушуй мене думати, переглянуте: Підхід до юзабіліті на основі здорового глузду". / Стів Круг - New Riders, 2014 – 216 с.
8. Ітан Браун. "Веб-розробка за допомогою HTML, CSS та JavaScript". / Ітан Браун - O'Reilly Media, 2020 – 408 с.
9. Джон Олсопп. "Окрема книга: Адаптивний веб-дизайн". / Джон Олсопп - A Book Apart, 2011 – 183 с.
10. Ніколя Бевакква. "Розробка додатків на JavaScript: A Build-First Approach". / Ніколя Бевакква - O'Reilly Media, 2015 – 182 с.
11. Бред Фрост. "Атомарний дизайн". / Бред Фрост - BradFrost, 2016 – 256 с.
12. Шей Хоу. "Вчимося програмувати HTML та CSS: Розробка та стилізація веб-сайтів". / Шей Хоу - Pearson, 2014 – 256 с.
13. Дженніфер Нідерст Роббінс. "Веб-дизайн у двох словах". / Дженніфер Нідерст Роббінс - O'Reilly Media, 2006 – 812 с.

14. Джон Дакетт. "JavaScript та JQuery: Інтерактивна інтерфейсна веб-розробка". / Джон Дакетт - Wiley, 2014 – 640 с.

15. Джессі Джеймс Гарретт. "Елементи користувацького досвіду: Орієнтований на користувача дизайн для Інтернету і не тільки". / Джессі Джеймс Гарретт - New Riders, 2010 – 192 с.

16. Девід Сойєр МакФарланд. "PHP та MySQL: Відсутній посібник". / Девід Сойєр МакФарланд - O'Reilly Media, 2012 – 526 с.

17. Адам Фріман. "Pro HTML5 з CSS, JavaScript та мультимедіа". / Адам Фріман - Apress, 2017 – 849 с.

18. Естель Вейл. "CSS: Відсутній посібник". / Естель Вейл - O'Reilly Media, 2020 – 898 с.

19. Ольховська Олена Володимирівна, Черненко Оксана Олексіївна. "Методичні рекомендації щодо оформлення пояснювальних записок до дипломної роботи". / Ольховська Олена Володимирівна, Черненко Оксана Олексіївна - Вищий навчальний заклад Укоопспілки «Полтавський університет економіки і торгівлі», 2023 – 67 с.

ДОДАТОК А. КОД ПРОГРАМИ

```
<!DOCTYPE html>

<html lang="ua">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Головна сторінка</title>

    <link rel="stylesheet" href="./css/index.css" />

    <link rel="stylesheet" href="./css/global.css" />

    <link rel="preconnect" href="https://fonts.googleapis.com" />

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />

    <link

      href="https://fonts.googleapis.com/css2?family=Raleway:ital,wght@0,100..900;1,100..900&display=swap"
rel="stylesheet" />

    <link

      href="https://fonts.googleapis.com/css2?family=Open+Sans:ital,wght@0,300..800;1,300..800&family=Raleway:ital,wght
@0,100..900;1,100..900&display=swap"

      rel="stylesheet" />

    <link

      href="https://fonts.googleapis.com/css2?family=Montserrat:ital,wght@0,100..900;1,100..900&display=swap"

      rel="stylesheet"/>

  </head>

  <body>

    <div class="bg-top"></div>
```

```
<header class="container">
  <a href="/">
    <img
      src="./assets/images/logo.svg"
      alt="logo"
      width="200"
      height="30"
    />
  </a>
  <nav class="h_nav align_center">
    <a href="/">Головна</a>
    <div class="h_select">
      <div class="align_center">
        <span>Штори</span>
        <img
          src="./assets/images/header/arrow.svg" alt="arrow" height="12" width="6" id="arrow_s1" class="arrow_ss"/>
      </div>
      <div id="catalog_s1" class="burger_th"></div>
    </div>
    <div class="h_select">
      <div class="align_center">
        <span>Тюль</span>
        <img
```

```
        src="./assets/images/header/arrow.svg" alt="arrow" height="12" width="6" id="arrow_s2" class="arrow_ss"/>
      </div>
      <div id="catalog_s2" class="burger_th"></div>
    </div>
    <div class="h_select">
      <div class="align_center">
        <span>Тасьма</span>
        <img
          src="./assets/images/header/arrow.svg" alt="arrow" height="12" width="6" id="arrow_s3" class="arrow_ss"/>
      </div>
      <div id="catalog_s3" class="burger_th"></div>
    </div>
    <a href="/other">Інше</a>
    <a href="/cart" style="display: flex; position: relative" id="cart-btn">
      <img
        src="./assets/images/header/cart.svg" alt="cart" height="21" width="24"/>
    </a>
  </nav>
  <div class="h_burger">
    <img
      src="./assets/images/header/burger.svg" alt="burger" width="26" height="17" onclick="showBurger()"/>
    <div class="mob_burg">
      <img
```

```
 src="./assets/images/home/close.svg" alt="close" width="21" height="21" onclick="closeBurger()"/>
<div>
 <a href="/" style="margin-bottom: 10px">Головна</a>
 <div class="hm_nav">
  <div class="align_center" onclick="showMobCatalog(1)">
   <div class="title-hm">
    <span>Штори</span>
    <img
     src="./assets/images/header/arrow.svg" alt="arrow" height="12" width="6" id="arrow_s1" class="arrow_ss"/>
   </div>
  </div>
  <div id="catalog_sm1"></div>
 </div>
 <div class="hm_nav">
  <div class="align_center" onclick="showMobCatalog(2)">
   <div class="title-hm">
    <span>Тюль</span>
    <img
     src="./assets/images/header/arrow.svg" alt="arrow" height="12" width="6" id="arrow_s1" class="arrow_ss"/>
   </div>
  </div>
  <div id="catalog_sm2"></div>
 </div>
```

```
      <div class="hm_nav">
        <div class="align_center" onclick="showMobCatalog(3)">
          <div class="title-hm">
            <span>Тасьма</span>
            <img
              src="./assets/images/header/arrow.svg" alt="arrow" height="12" width="6" id="arrow_s1" class="arrow_ss"/>
          </div>
        </div>
        <div id="catalog_sm3"></div>
      </div>
      <a href="/catalog">Інше</a>
    </div>
  </div>
</header>
<!-- Main -->
<main>
  <!-- Contact Modal (Send Mail) -->
  <div id="contact-modal">
    <div id="contact-modal-child">
      <img
        src="./assets/images/home/close.svg"
        alt="close"
```

```
        width="21"
        height="21"
        onclick="contactClose()"/>
      <h2>Зв'язатися</h2>
      <form onsubmit="contactSubmit(event)">
        <fieldset>
          <label for="name" class="req">Ім’я</label>
          <input type="text" name="name" placeholder="Валентин(а)" class="input-ds" id="name-check"
oninput="onChangeName(event)" onblur="onBlurName(event)"/>
        </fieldset>
        <fieldset>
          <label for="phone">Телефон</label>
          <input type="tel" name="phone" placeholder="+380 xx xxx xx xx" class="input-ds"/>
        </fieldset>
        <fieldset>
          <label for="email" class="req">Пошта</label>
          <input
            type="email"
            name="email"
            placeholder="example@gmail.com"
            class="input-ds"
            id="email-check"
            oninput="onChangeEmail(event)"
            onblur="onBlurEmail(event)"/>
```

```
      </fieldset>
      <fieldset>
        <label for="message" class="req">Повідомлення</label>
        <textarea
          name="message"
          class="input-ds"
          id="mess-check"
          oninput="onChangeMess(event)"
          onblur="onBlurMess(event)"></textarea>
      </fieldset>
      <button type="submit" class="button-tr" id="contact-btn-d">
        Відправити
      </button>
    </form>
  </div>
</div>
<div class="lv1 container">
  <div>
    <h1>Пошив під замовлення штор та тюль</h1>
    <p>
      Наші майстри з шиття створять для вас неповторні штори, тюлі та
      тасьми, які додають затишку і стилю вашому приміщенню.
    </p>
```

```
    <a href="/catalog"><button class="button-bg">Каталог</button></a>
  </div>
  <img
    src="./assets/images/home/top.png"
    alt="top-img"
    width="519"
    height="680"/>
</div>
<div class="lv2">
  <div class="container">
    <h2>Трохи про нас</h2>
    <p>
      Привіт! Ми - команда інтернет-магазину, яка спеціалізується на
      шторах, тюлях, тасьмах та індивідуальному пошитті. Наша мета -
      зробити ваш дім ще затишнішим та стильнішим, надаючи вам можливість
      створити простір, який відображатиме ваші уподобання та особливості.
    </p>
    <p>
      Ми розуміємо, що вибір текстилю для вашого домашнього інтер'єру - це
      важливе завдання, тому пропонуємо вам широкий вибір матеріалів і
      кольорів. Наші фахівці з радістю нададуть вам професійні поради щодо
      вибору і дизайну, щоб кожен елемент вашого простору відображав ваш
      стиль та персональність.
```

```
    </p>
    <p>
      Приєднуйтесь до нашої спільноти, де ми не лише пропонуємо вам
      високоякісні товари, а й ділимося корисними порадами та ідеями з
      декорування вашого дому. Разом з нами ви зможете створити простір
      вашої мрії, в якому кожен день буде наповнений комфортом та
      гармонією. Долучайтеся та відкрийте для себе нові можливості дизайну
      вашого дому разом з нашим інтернет-магазином!
    </p>
  </div>
</div>
<!-- Company Info Images -->
<div class="lv3 container">
  <h2>Фото клієнтів</h2>
  <div id="imagesGallery"></div>
  <div class="pagination">
    <button class="arrow" onclick="prevSlide()" id="arrow-left">
      <img src="./assets/images/arrow-left.svg" alt="arrow-left" width="17" height="20"/>
    </button>
    <div class="list"></div>
    <button class="arrow" onclick="nextSlide()" id="arrow-right">
      <img src="./assets/images/arrow-right.svg" alt="arrow-right" width="17" height="20"/>
    </button>
```

```
  </div>
</div>
<div class="lv4">
  <div class="container">
    <h2>Відгуки</h2>
    <div id="slide-b">
      <div>
        <div>
          <h3>Софія Гулевская</h3>
          <p>
            Вибрала тут чудові штори в кімнату.<br />Ціна приємно
            здивувала якість гарна.<br />Мені дуже сподобався цей
            магазин.<br />Рекомендую всім хто хоче придбати гарні штори за
            хорошою ціною.
          </p>
        </div>
        <div>
          <h3>Анастасия Юренко</h3>
          <p>
            Купувала штори. Широкий вибір товару, допомогли вибрати
            потрібний варіант.<br />Гарне обслуговування і ціни.<br />Штори
            гарної якості, рекомендую.
          </p>
```

```
  </div>
  <div>
    <h3>Анна Севрюкова</h3>
    <p>
      Обслуговування на найвищому рівні! Сучасний вибір якісних
      тканин. Штори вийшли шикарні!
    </p>
  </div>
</div>
<div>
  <div>
    <h3>Софія Дмитрієвич</h3>
    <p>
      Дуже вдячна Вам за вашу роботу! В мене проблемне еркерне вікно
      з різною висотою.<br />В багатьох магазинах казали що не
      реально зробити те що я хотіла. Але тут змогли не лише
      підібрати і ідеально встановити карнизи, а ще й підібрати і
      пошити тюль та штори за моїм бажанням. Вийшло просто чудово!
      Всім, хто й досі думає, що їхні задуми майже неможливо
      втілити, нехай переконається в зворотному, звернувшись до
      цього магазину!!
    </p>
  </div>
```

```
          <div>
            <h3>Світлана Окань</h3>
            <p>
              Чудовий магазин! Прекрасний персонал та величезний асортимент.
              Допомогли з вибором, виготовили все в терміни, штори дуже
              красиві, довжина підібрана ідеально.<br />Без поспіху все
              показали, порадили,розповіли як правильно вішати. З Вами дуже
              приємно мати справу! Дякую за вашу роботу!
            </p>
          </div>
        </div>
      </div>
      <div class="justify_center">
        <a target="_blank"
href="https://www.google.com/maps/place/%D0%93%D0%B0%D0%BB%D0%B5%D1%80%D0%B5%D1%8F+%D0%A8
%D1%82%D0%BE%D1%80/@49.5654233,34.5100339,15z/data=!4m18!1m9!3m8!1s0x40d82f2bfeed0a73:0xc08dbc25d777
df10!2z0JPQsNC70LXRgNC10Y8g0KjRgtC-
0YA!8m2!3d49.5654233!4d34.5100339!9m1!1b1!16s%2Fg%2F11h80226zl!3m7!1s0x40d82f2bfeed0a73:0xc08dbc25d777df
10!8m2!3d49.5654233!4d34.5100339!9m1!1b1!16s%2Fg%2F11h80226zl?entry=ttu" >Більше відгуків</a>
      </div>
    </div>
  </div>
  <!-- Company Info Contact -->
  <div class="lv5 container" id="contact">
    <h2>Зв'язатися</h2>
    <div class="info_contact">
```

```
  <label for="">Адресса</label>
  <p>Полтава, вулиця Івана Мазепи, 17</p>
 </div>
 <div class="info_contact">
  <label for="">Години праці</label>
  <p>Щоденно з 9:30 до 18:30</p>
 </div>
 <div class="info_contact">
  <label for="">Телефон</label>
  <p>+380 050 862 7508</p>
 </div>
 <div class="info_contact">
  <label for="">Пошта</label>
  <p>figma.ua@gmail.com</p>
 </div>
 <button class="button-tr" onclick="contactGet()">Зв'язатися</button>
 <iframe
  title="geolocation"
  src="https://maps.google.com/maps?q=ул.+Ивана+Мазепы,+17,+Полтава,+Полтавская+область,+36000&amp;t=&amp;z=18&amp;ie=UTF8&amp;iwloc=&amp;output=embed"
  frameborder="0"
  loading="lazy"
  width="190%"
  height="430"
```

```
        allowfullscreen
      ></iframe>
    </div>
  </main>
  <a href="/#contact" class="req-br" style="z-index: 5">
    <img src="./assets/images/req.svg" alt="req-br" width="13" height="24" />
  </a>
  <footer>Copyright © 2024 “Галерея штор”. Усі права захищені.</footer>
  <script src="./js/header.js"></script>
  <script>
    let validName = false;
    let validEmail = false;
    let validMess = false;
function eventBlockClose(e) {
      if (!e.target.closest("#contact-modal-child")) contactClose();}
    function contactGet() {
      document.getElementById("contact-modal").classList.add("active");
      disScroll();
      setTimeout(
        () => document.addEventListener("click", eventBlockClose),500);}
    function contactClose() {
      document.getElementById("contact-modal").classList.remove("active");
      allScroll();
```

```
  document.removeEventListener("click", eventBlockClose);}

function onChangeName(event) {

  if (event.target.value === "") {

    document.getElementById("name-check").classList.add("error");

    validName = false;

  } else {

    document.getElementById("name-check").classList.remove("error");

    validName = true;}}

function onBlurName(event) {

  if (event.target.value === "")

    document.getElementById("name-check").classList.add("error");}

function onChangeEmail(event) {

  const emailPattern =

    /^(([^<>()[\]\\.,;:\s@"]+(\.[^<>()[\]\\.,;:\s@"]+)*)|(".+"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/;

  if (

    event.target.value === "" ||

    !emailPattern.test(event.target.value)

  ) {

    document.getElementById("email-check").classList.add("error");

    validEmail = false;

  } else {

    document.getElementById("email-check").classList.remove("error");

    validEmail = true; }}
```

```
function onBlurEmail(event) {
  if (event.target.value === "")
    document.getElementById("email-check").classList.add("error");}
function onChangeMess(event) {
  if (event.target.value === "") {
    document.getElementById("mess-check").classList.add("error");
    validMess = false;
  } else {
    document.getElementById("mess-check").classList.remove("error");
    validMess = true;
  }}
function onBlurMess(event) {
  if (event.target.value === "")
    document.getElementById("mess-check").classList.add("error");}
function contactSubmit(event) {
  event.preventDefault();
  if (!validName) {
    document.getElementById("name-check").classList.add("error");}
  if (!validEmail) {
    document.getElementById("email-check").classList.add("error"); }
  if (!validMess) {
    document.getElementById("mess-check").classList.add("error");}
  if (validEmail && validMess && validName) {
```

```
    document.getElementById("contact-btn-d").disabled = true;
    const formData = new FormData(event.target);
    fetch("./server/mail/contact_send.php", {
      method: "POST",
      body: formData,
    }).then((response) => {
      document.getElementById("contact-btn-d").disabled = false;
      if (response.ok) {
        contactClose();}
    });
  }}
</script>
<script>
  let countImage = 1;
  let index = 1;
  const screenWidth = window.innerWidth;
  let listImage = [];
  function calculateImagesToShow() {
    if (screenWidth > 1000) {
      return 6;
    } else if (screenWidth > 760) {
      return 4;
    } else {
```

```
    return 1;
  }}
fetch("./server/slider/slider.php", {
  method: "GET",
  headers: {
    "Content-Type": "application/json",},
})
  .then((response) => response.json())
  .then((images) => {
    listImage = images;
    countImage = images.length;
    const imageGallery = document.getElementById("imagesGallery");
    const listPag = document.getElementsByClassName("list");
    const count = calculateImagesToShow();
    let pageCount = Math.floor(countImage / count);
    if (pageCount <= 1) {
      document.getElementById("arrow-left").disabled = true;
      document.getElementById("arrow-right").disabled = true;}
    if (count === 1 && pageCount > 7) pageCount = 7;
    for (let i = 1; i <= pageCount; i++) {
      const div = document.createElement("div");
      div.id = `page-${i}`;
      if (i === 1) div.className = "active";
```

```
      listPag[0].appendChild(div);}
    images.forEach((image, i) => {
      if (i < count) {
        const imageName = image.split(".")[0];
        const img = document.createElement("img");
        img.src = `./assets/images/home/list/${image}`;
        img.alt = imageName;
        img.id = `img-${imageName}`;
        imageGallery.appendChild(img);}
    });
  })
  .catch((error) => console.error("Помилка завантаження зображень:", error));
function changeSlide(direction) {
  const imageGallery = document.getElementById("imagesGallery");
  const count = calculateImagesToShow();
  const pageCount = Math.ceil(countImage / count);
  let newIndex;
  if (direction === "prev") {
    newIndex = index - count;
    if (newIndex < 1) {
      newIndex += countImage;}
  } else if (direction === "next") {
    newIndex = index + count;
```

```
    if (newIndex > countImage) {
      newIndex -= countImage;
    } }
  index = newIndex;
  const paginationBlocks = document.querySelectorAll(".list > div");
  paginationBlocks.forEach((block) => block.classList.remove("active"));
  const currentPage = Math.ceil(index / count);
  const activePage = document.getElementById(`page-${currentPage}`);
  if (activePage) {
    activePage.classList.add("active");}
  imageGallery.innerHTML = "";
  for (let i = index; i < index + count; i++) {
    const imgIndex = i > countImage ? i - countImage : i;
    const imageName = listImage[imgIndex - 1].split(".")[0];
    const img = document.createElement("img");
    img.src = `./assets/images/home/list/${listImage[imgIndex - 1]}`;
    img.alt = imageName;
    img.id = `img-${imageName}`;
    imageGallery.appendChild(img);}}
function prevSlide() {
  changeSlide("prev");}
function nextSlide() {
  changeSlide("next");}
```

```
</script>

<script>

  if (window.innerWidth <= 760) {

    let slideB = document.getElementById("slide-b");

    let isTouchSlideB = false;

    let startSlideX = 0;

    let startTranslateX = 0;

    let sliderWidth = slideB.scrollWidth;

    let sliderOffset = slideB.offsetWidth;

    function changeTouchedB(isTouch) {

      isTouchSlideB = isTouch;

      if (!isTouch) {

        startSlideX = 0;

        startTranslateX =

          parseInt(

            slideB.style.transform

              .replace("translateX(", "")

              .replace("px)", "")

          ) || 0;

      }}

    function slideMoveB(e) {

      if (isTouchSlideB) {

        const currentX = e.clientX || e.touches[0].clientX;
```

```
    if (startSlideX !== 0) {
      const diff = currentX - startSlideX;
      let newTranslateX = startTranslateX + diff;
      if (newTranslateX >= 0) newTranslateX = 0;
      else if (newTranslateX <= -(sliderWidth - sliderOffset))
        newTranslateX = -(sliderWidth - sliderOffset);
      slideB.style.transform = `translateX(${newTranslateX}px)`;}
  }}
slideB.addEventListener("mousedown", (e) => {
  changeTouchedB(true);
  startSlideX = e.clientX;
  slideB.style.cursor = "grabbing";
});
slideB.addEventListener("mousemove", slideMoveB);
slideB.addEventListener("mouseup", () => {
  changeTouchedB(false);
  slideB.style.removeProperty("cursor");
});
slideB.addEventListener("touchstart", (e) => {
  changeTouchedB(true);
  startSlideX = e.clientX;
  slideB.style.cursor = "grabbing";
});
```

```
        slideB.addEventListener("touchmove", slideMoveB);
        slideB.addEventListener("touchend", () => {
          changeTouchedB(false);
          slideB.style.removeProperty("cursor");
        });
      }
    </script>
    <script src="./js/cart.js"></script>
    <script src="./js/global.js"></script>
  </body>
</html>
```