

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук та технологій  
(повне найменування факультету)

Кафедра комп'ютерних систем та мереж  
(повне найменування кафедри)

**Пояснювальна записка**

до дипломного проекту (роботи)

бакалавра

(ступінь вищої освіти (освітній ступінь))

на тему РОЗРОБКА ІНТЕРНЕТ-МАГАЗИНУ «FASHION ART»

Виконав: студент 4 курсу, групи КНТз-510  
спеціальності \_\_\_\_\_

123 Комп'ютерна інженерія

(код і найменування спеціальності)

Освітня програма (спеціалізація) \_\_\_\_\_

«Комп'ютерна інженерія»

ЛОГВИНА А.С.

(ПРИЗВИЩЕ та ініціали)

Керівник ТЯГУНОВА М.Ю.

(ПРИЗВИЩЕ та ініціали)

Рецензент СТЕПАНЕНКО О.О.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
**Національний університет «Запорізька політехніка»**  
( повне найменування вищого навчального закладу )

Факультет Комп'ютерних наук і технологій  
Кафедра «Комп'ютерні системи та мережі»  
Ступінь вищої освіти бакалаврський  
Спеціальність 123 Комп'ютерна інженерія  
(код і найменування)  
Освітня програма (спеціалізація) Комп'ютерна інженерія  
(назва освітньої програми (спеціалізації))

**ЗАТВЕРДЖУЮ**  
Завідувач кафедри КУДЕРМЕТОВ Р.К.  
“ ” 2024 року

**З А В Д А Н Н Я**  
**НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА**

ЛОГВИНОЇ Ангеліни Сергіївни

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка інтернет-магазину «Fashion Art»

керівник проєкту (роботи) к. т. н., доцент, ТЯГУНОВА Марія Юріївна

(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом вищого навчального закладу від “01” квітня 2024 року № 105

2. Строк подання студентом проєкту (роботи) 26 травня 2024 року

3. Вихідні дані до проєкту (роботи) рекомендації з проєктування структури та інтерфейсу користувача інтернет-магазину, технології для розробки інтернет-магазину

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1) Аналіз предметної області;

2) Моделювання програмного забезпечення;

3) Проєктування вебзастосунку;

4) Реалізація інтернет-магазину

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

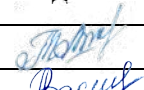
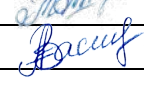
1) Структура теки з версткою

2) Структура клієнтської частини на Vue

3) Структура серверної частини на Laravel

4) Головна сторінка онлайн-магазину

## 6. Консультанти розділів проєкту (роботи)

| Розділ        | ПРІЗВИЩЕ, ініціали та посада консультанта | Підпис, дата                                                                        |                                                                                     |
|---------------|-------------------------------------------|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|
|               |                                           | завдання видав                                                                      | прийняв виконане завдання                                                           |
| 1-4           | ТЯГУНОВА М.Ю., к.т.н., доцент             |  |  |
| нормоконтроль | ПОЛЬСЬКА О.В., ст. викл.                  |  |  |
|               |                                           |                                                                                     |                                                                                     |
|               |                                           |                                                                                     |                                                                                     |
|               |                                           |                                                                                     |                                                                                     |
|               |                                           |                                                                                     |                                                                                     |
|               |                                           |                                                                                     |                                                                                     |

7. Дата видачі завдання 01.03.2024 р.

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів дипломного проєкту (роботи)                                                                 | Строк виконання етапів проєкту (роботи ) | Примітка |
|-------|----------------------------------------------------------------------------------------------------------|------------------------------------------|----------|
| 1     | Аналіз предметної області, огляд рекомендацій з проєктування та технологій, вибір оптимальних технологій | 01.03.2024 р.                            |          |
| 2     | Розробка вимог до програмного забезпечення                                                               | 15.03.2024 р.                            |          |
| 3     | Моделювання програмного забезпечення                                                                     | 20.03.2024 р.                            |          |
| 4     | Проектування застосунку                                                                                  | 25.03.2024 р.                            |          |
| 5     | Реалізація інтернет-магазину                                                                             | 01.04.2024 р.                            |          |
| 6     | Результати роботи                                                                                        | 18.05.2024 р.                            |          |
| 7     | Оформлення отриманих результатів у ПЗ                                                                    | 22.05.2024 р.                            |          |
| 8     | Оформлення графічного матеріалу                                                                          | 24.05.2024 р.                            |          |
| 9     | Оформлення допоміжного матеріалу                                                                         | 26.06.2024 р.                            |          |
|       |                                                                                                          |                                          |          |
|       |                                                                                                          |                                          |          |
|       |                                                                                                          |                                          |          |
|       |                                                                                                          |                                          |          |
|       |                                                                                                          |                                          |          |

Студент (ка)

  
 ( підпис )

**Ангеліна ЛОГВИНА**  
 (Ім'я ПРІЗВИЩЕ)

Керівник проєкту (роботи)

  
 ( підпис )

**Марія ТЯГУНОВА**  
 (Ім'я ПРІЗВИЩЕ)

## РЕФЕРАТ

ПЗ: 85 с., 49 рис., 56 джерел.

МАГАЗИН ОДЯГУ, ПРЕЦЕДЕНТ, СПЕЦИФІКАЦІЯ ВИМОГ, ОНЛАЙН-МАГАЗИН, SRS, UML, LARAVEL, VUE

Мета роботи – підвищити прибутковість компанії «Fashion Art» за рахунок розробки онлайн-магазину.

Об’єкт дослідження – методи та засоби розробки інтернет-магазинів.

Предмет розробки – інтернет-магазин для одягу «Fashion Art».

Диплом складається з чотирьох частин.

У першому розділі проведено аналіз доцільності розробки інтернет магазину для компанії «Fashion Art», та аналіз відповідних технологій для створення інтернет-магазину. Для розробки інтернет-магазину «Fashion Art» обрано Bootstrap (для верстки), Laravel (для серверної частини) та Vue (для клієнтської частини). Також розроблено вимоги до розробки програмного забезпечення.

У другому розділі проведено моделювання програмного забезпечення з використанням мови UML. Розроблено діаграми варіантів використання, діаграми послідовностей на основі опису предметної області, вимог та специфікації вимог. Наведено розгорнутий опис прецедентів.

У третьому розділі описано проєктування архітектури вебзастосунку та макетування інтерфейсу користувача.

У четвертому розділі наведено програмний код реалізації інтернет-магазину та скріншоти працюючого інтернет-магазину одягу «Fashion Art».

## ЗМІСТ

|                                                                            |    |
|----------------------------------------------------------------------------|----|
| Вступ.....                                                                 | 7  |
| 1 Аналіз предметної області.....                                           | 9  |
| 1.1 Доцільність розробки інтернет-магазину для розвитку бізнесу.....       | 9  |
| 1.2 Аналіз технологій для розробки інтернет-магазину.....                  | 12 |
| 1.2.1 SaaS-платформи.....                                                  | 12 |
| 1.2.2 Системи керування контентом.....                                     | 15 |
| 1.2.3 Мови програмування та фреймворки.....                                | 19 |
| 1.3 Вибір оптимальних технологій для інтернет-магазину «Fashion Art» ..... | 32 |
| 1.4 Розробка вимог до програмного забезпечення .....                       | 34 |
| 2 Моделювання програмного забезпечення .....                               | 36 |
| 2.1 Діаграми варіантів використання .....                                  | 37 |
| 2.2 Діаграми послідовності .....                                           | 39 |
| 2.3 Розгорнуті описи прецедентів.....                                      | 44 |
| 3 Проєктування вебзастосунку .....                                         | 48 |
| 3.1 Проєктування архітектури програмного забезпечення.....                 | 48 |
| 3.1.1 Вибір архітектурних рішень для проєкту .....                         | 48 |
| 3.1.2 Побудова діаграми розміщення.....                                    | 51 |
| 3.2 Проєктування інтерфейсу користувача.....                               | 51 |
| 3.2.1 Опис змісту сторінок вебдодатку .....                                | 51 |
| 3.2.2 Ескізи сторінок .....                                                | 53 |
| 4 Реалізація інтернет-магазину .....                                       | 56 |
| 4.1 Дизайн сайту .....                                                     | 56 |

|                                |    |
|--------------------------------|----|
| 4.2 Верстання сайту.....       | 58 |
| 4.3 Програмування сайту.....   | 70 |
| Висновки .....                 | 80 |
| Перелік джерел посилання ..... | 81 |

## ВСТУП

Протягом останніх років можна легко простежити тенденцію до діджиталізації бізнесу, яка благотворно впливає як на розвиток бізнесу, так і на задоволення споживачів товарів або послуг. Власнику вона дозволяє автоматизувати роботу з клієнтами, розширити аудиторію, спростити бухгалтерію, оптимізувати персонал, керувати розвитком компанії, контролювати процеси та багато іншого. Споживачам вона дозволяє швидко ознайомитися з продуктами або послугами компанії і так же само швидко та зручно замовити їх. Отож, як сказано у статті [1], «діджиталізація – це зручність і вигода для обох сторін.» Зручність та швидкість зробила діджиталізацію справжнім трендом, що показує твердження зі статті [1]: «Сьогодні 9 з 10 компаній вже запустили діджиталізацію або перебувають на шляху до неї.» Як ми зараз бачимо, пандемія COVID-19 стала потужним каталізатором цифровізації бізнесу. Як сказано у статті [2], «За даними Salesforce в другому кварталі 2020 роки онлайн-продажі у світі зросли на 71%. Україна — не виняток. За даними СБР в першому півріччі кількість українських інтернет-користувачів, які купують щонебудь онлайн, зросла на 6% до 9,1 млн.» Багато офлайн-магазинів, серед яких «Сільпо», «АТБ», «Фора», «Varus», «Prostor», «MasterZoo», вирішили оцифрувати свій бізнес, щоб забезпечити можливість зручно, швидко та безпечно купувати товари через Інтернет. У цього процесу переосмислення підходу до ведення бізнесу можна виділити чотири рушійні сили:

- перевага покупцями покупок онлайн перед офлайн-покупками. Як сказано у статті [3], «За даними Statista, у 2022 році кількість користувачів електронної комерції в Європі зросла до \$502 млн. А до 2026 року ця цифра обіцяє зрости ще на \$62 млн.»;

- високі очікування сучасних клієнтів, які потребують персоналізованого досвіду через усі можливі канали;

- обов'язковість присутності в Інтернеті. Без цього буде важко досягти успіху;

- технології стимулюють розвиток бізнесу та є мірилом успіху компанії.

З початком пандемії переведення бізнесу в онлайн стало не просто бажаним, а необхідним. Як сказано у статті [3], «Компанія повинна трансформувати внутрішні бізнес-процеси, на яких побудована комерція, щоб досягти чудових результатів: замовлення, ціноутворення, маркетинг, закупівлі, доставка тощо.» Втіленням цієї трансформації є інтернет-магазин. Таким чином, обрана тема дипломної роботи є актуальною.

Метою роботи є підвищити прибутковість компанії «Fashion Art» за рахунок розробки онлайн-магазину.

Для досягнення поставленої мети необхідно виконати наступні завдання:

- проаналізувати предметну область та технології створення інтернет-магазинів;
- розробити вимоги до розробки програмного забезпечення;
- провести моделювання програмного забезпечення;
- спроектувати вебзастосунок;
- реалізувати інтернет-магазин одягу «Fashion Art».

## 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

### 1.1 Доцільність розробки інтернет-магазину для розвитку бізнесу

Як вказано у статті [4], «За останній місяць я купував речі в інтернеті 3 рази. Як показує статистика, я не самотній — кожен п'ятий житель України регулярно робить покупки в онлайн-магазинах. У 2017 році українці залишили в інтернет-магазинах приблизно 14 мільярдів гривень — уявіть, які можливості ви втрачаєте прямо зараз.», багато людей стали віддавати перевагу онлайн-покупкам. Особливо виразним це явище стало з початком пандемії. За даними з джерела [2], «За даними Salesforce в другому кварталі 2020 роки онлайн-продажі у світі зросли на 71%. Україна — не виняток. За даними СБР в першому півріччі кількість українських інтернет-користувачів, які купують що-небудь онлайн, зросла на 6% до 9,1 млн. 16% з них замовляють доставку продуктів харчування, 20% — готові страви, навіть після ослаблення карантину.» Отож, інтернет-магазин у наші часи є майже обов'язковим для успішності бізнесу. Інтернет-магазини мають декілька великих переваг, які зробили їх бажаним вибором при виникненні необхідності щось купити. Усі вони перераховані у списку нижче:

- економія часу та зусиль; не потрібно йти до будівлі магазину та стояти в черзі;
- великий асортимент товарів;
- можна отримати детальну інформацію про будь-який товар;
- хороші знижки/низькі ціни завдяки тому, що власникам не потрібно платити за оренду приміщення для магазину;
- відсутність тиску зі сторони продавців при здійсненні купівлі.

Крім переваг для покупців, Інтернет-магазини несуть масу вигід для підприємців:

- швидке розширення бізнесу з мінімальними витратами;
- Інтернет-магазин вимагає менше коштів на підтримку ніж звичайний магазин;

– простіше обслуговувати клієнтів.

Вищезазначені вигоди виглядають дуже привабливо. А на ділі для того, щоб вдалося успішно втілити цю затію в життя буде потрібно, щоб магазин був безпечним, швидким, зручним у використанні, сучасним, правильно оптимізованим. Не останню роль у цьому відіграє й правильне проєктування сайту, на якому буде розміщуватися додатковий канал збуту товарів. Важливість правильного проєктування підтверджують слова зі статті [6]: «Аксіома: чим менше часу і сил користувач витрачає на роботу з сайтом, тим більш лояльним він стає до компанії. Існують сотні інтернет-магазинів, і ніхто не буде «пересилиювати себе» і терпіти незручності, щоб скористатися саме вашою пропозицією. Користувачеві простіше закрити вкладку і знайти більш «доброзичливий» портал. Для покупця це заощаджений час, для вас – збитки.»

Перша складова правильного проєктування – грамотне проєктування структури інтернет-магазину. Без цього ніяк не можна обійтися через ряд причин:

– якщо користувачу буде важко знайти необхідну інформацію, знизиться відвідуваність та час, проведений одним користувачем на сайті;

– попередня причина також буде негативно впливати на просування сайту у пошукових системах, бо вони орієнтуються на зазначені у попередньому пункті показники: відвідуваність та час, проведений одним користувачем на сайті;

– знизиться лояльність клієнтів, бо, як сказано вище, ніхто не буде пересилиювати себе, а скоріш підуть до конкурента, а ваш магазин буде запасним, а не головним місцем покупки;

– погана структура несе складнощі при підтримці, буде важко додавати нові товари та сторінки.

Найбільш розповсюджена структура Інтернет-магазинів – деревоподібна структура, приклад якої показано на рисунку 1.1.

Існує декілька правил, за якими будується структура онлайн-магазину:

– один товар знаходиться в одному розділі, максимум у двох, але властивостей це не стосується;

- невеликий рівень складеності, невелика кількість підрозділів у категорії, для більш точного вибору є пошуковий фільтр;
- прості назви підрозділів.

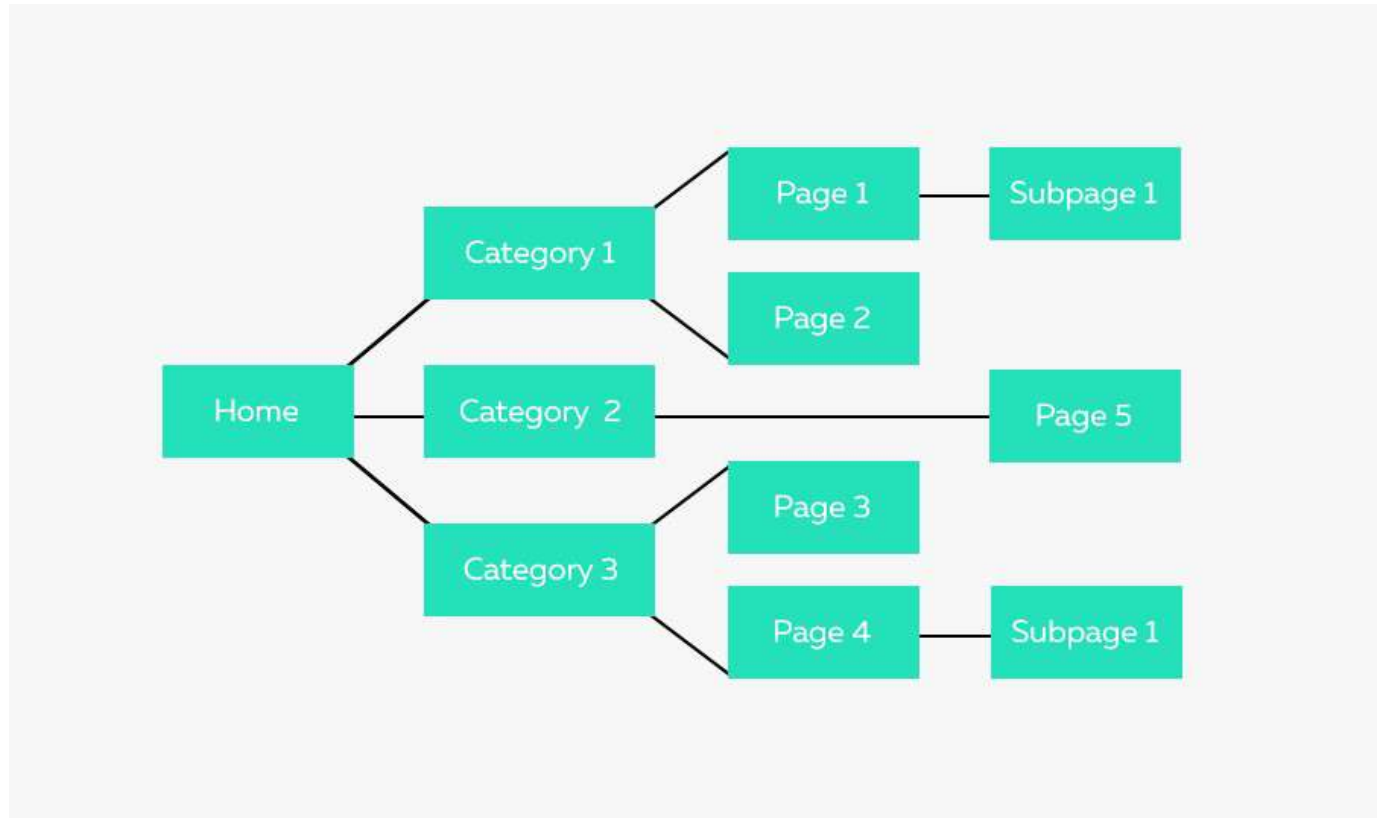


Рисунок 1.1 – Приклад деревоподібної структури Інтернет-магазину [4]

Друга складова успішного проєктування – створення інтерфейсу користувача. Одна зі значних помилок на цьому етапі – перевага краси та яскравості дизайну над його зручністю. Краса може привернути увагу, але вона не є двигуном торгівлі. Звісно, дизайн може бути сучасним (але не сильно яскравим), щоб покупцям було приємніше проводити свій час на сторінках магазину, але зручність та функціональність повинні бути пріоритетом номер один. Інша подібна помилка – складність оформлення замовлення. Форма замовлення повинна бути інтуїтивно зрозумілою і мати мінімальну кількість полів. Потрібна можливість оформлення замовлення без реєстрації, бо багато людей не довіряють свої персональні дані. Зручність користування треба враховувати і при проєктуванні інтерфейсу кошику: у

користувача повинна бути можливість регулювати кількість товару, видаляти товар з кошика та повертатися на попередню сторінку.

## **1.2 Аналіз технологій для розробки інтернет-магазину**

Наступний крок до готового застосунку – вибір засобів розробки. Існують три найбільш популярні рішення для створення сайту:

- SaaS-платформи (англ. software as a service - програмне забезпечення, як послуга);
- розробка на CMS (англ. content management system - система управління сайтом) (сайти можуть бути шаблонні або кастомні);
- розробка на фреймворках.

### **1.2.1 SaaS-платформи**

SaaS-платформа – платформа, яка надає готовий сайт та його підтримку за деякі кошти. Згідно з цією моделлю застосунків, постачальник послуги розробляє ПЗ, розгортає його за рахунок своєї інфраструктури, обслуговує його та надає клієнтам доступ до застосунку в Інтернеті як до готового Інтернет-сервісу. Ще одна особливість такого підходу у тому, що замовник не купує код застосунку, а орендує сам застосунок. А оплата за оренду здійснюється як оплата за підписку: один раз на деякий період (місяць, рік). Підходить для розробки нескладних проєктів або для проєктів для тестування нових напрямків у бізнесі, що вже діє. Для виготовлення сайту достатньо заповнити шаблон інформацією, тому готовий сайт можна створити за декілька днів. Але у цього рішення, нажаль, є ряд недоліків. По-перше, надається тільки шаблонний дизайн та функціональність, змінення та доопрацювання яких неможливе. По-друге, сайти, побудовані на таких платформах, не витримують високих навантажень, вони не швидкі, не відмовостійкі, не продуктивні. По-простому SaaS-платформа ще може називатися конструктором, у випадку якщо мова ведеться саме про вебзастосунки. Прикладами SaaS-платформ є Weblium, Wix,

uCoz. Саме для створення Ecommerce-проектів ще є такі: Хорошоп, Rescommerce, Shop-Express, fesh. (так, саме з точкою в кінці), Solomon0. Звісно, це не усі з них, бо напрямок SaaS-платформ зараз популярний та активно розвивається.

Weblium – це зручний та функціональний конструктор вебзастосунків, який дозволяє створювати проекти різного рівня складності: від односторінкових сайтів до корпоративних сайтів та інтернет-магазинів. Але позиціонується він як конструктор для розробки односторінкових сайтів. Він був розроблений у 2017 році.

Головні переваги цього конструктора:

- велика бібліотека шаблонів для створення сучасних сайтів;
- зручний інтерфейс конструктора;
- усі шаблони та створені вебпроекти адаптивні;
- зручно працювати зі зверненнями відвідувачів;
- інтеграція з різними сторонніми сервісами;
- можливість створення впливаючих вікон;
- ШІ-помічник, який виправляє розповсюджені помилки, дає пропозиції з покращення дизайну та підказки для швидкого та ефективного верстання;
- необмежений простір для зберігання файлів;
- при оплаті одного з тарифів надається безкоштовний домен та SSL-сертифікат;
- цілодобова підтримка;
- регулярні оновлення.

Основні недоліки Weblium:

- обмеження в інтеграції;
- нема можливості приховати деякі блоки на смартфонах;
- більшість готових тем – односторінкові;
- мало функціоналу у блогу;
- мало інформації для аналізу конверсій.



Рисунок 1.2 – Логотип Weblum

Wix – платформа, яка підкорила серця мільйонів користувачів завдяки різноманіттю шаблонів, зрозумілому інтерфейсу та можливостям інтеграції. Wix прекрасно підходить для запуску як лендінгу, так і повноцінного маркетплейсу, його можна застосувати при розробці проєктів абсолютно будь-якої складності: від рекламних сайтів, лендінгів та портфоліо до навіть складних проєктів, які потребують індивідуальної розробки. Конструктор оцінить як новачок, так і досвідчений розробник. Wix має цілий ряд суттєвих переваг, які можуть зробити його дуже бажаним вибором для багатьох:

- величезна бібліотека професіональних шаблонів на будь-який смак;
- потужний візуальний редактор з найширшими можливостями;
- конструктор логотипів;
- сотні корисних розширень;
- просунута SEO-оптимізація;
- розробка мобільних застосунків, що допомагаю підвищити лояльність до бренду;
- підходить для досвідчених розробників завдяки можливості кастомізації;
- цілодобова підтримка;
- регулярне оновлення функціоналу.

Але це рішення має деякі недоліки:

- порівняно висока ціна: базовий пакет Light буде коштувати мінімум 16 доларів на місяць;
- надлишковий функціонал редактора: кількість налаштувань та віджетів можуть налякати початківця;
- частина розширень не працюють: розширення від сторонніх розробників можуть довго не оновлюватися;

- повільна швидкість завантаження застосунку.

Хорошоп – SaaS-платформа, яка дозволяє пройти увесь шлях від подачі заявки до повністю працюючого магазину за декілька днів. При цьому цей шлях дуже простий, у ньому всього декілька кроків: оформлення заявки, вибір дизайну, імпорт товарів та запуск сайту. У конструктора добра репутація: він припав до серця більш ніж 8000 клієнтів. А бажаним вибором для них його зробили такі переваги:

- для SEO-просування проєкту не потрібні якісь доопрацювання;
- у клієнта є вибір з більш ніж ста дизайнів та можливість підігнати кожний з них під свої запити;
- можна керувати структурою сайту, порядком блоків на головній сторінці, змістом сторінок, налаштовувати фільтрацію та категорії товарів;
- готові інтеграції з 1С, платіжними системами та службами доставки;
- наявність повноцінної мобільної версії;
- навчальні матеріали;
- підтримка сайту 24/7;
- регулярні оновлення;
- постійна підтримка.

Основні недоліки Хорошопу наступні:

- обмежений функціонал у базовому тарифі, через це може доведеться переводитися на інший, більш дорогий план;
- повна залежність від планів розробника: тут не вийде внести якісь технічні зміни в сайт.

### **1.2.2 Системи керування контентом**

CMS (Content Management System) – система керування вмістом сайту, каркас сайту, який можна редагувати та заповнювати контентом без необхідності вчити програмування. На практиці CMS працює як вебдодаток для керування змістом сайту, яким можуть бути текст, рисунки, відео тощо, та являє собою програмне забезпечення, яке надає інструментарій для створення, редагування, видалення контенту вебсайту, зміни дизайну та забезпечення безпеки даних. У цьому рішенні

дизайн та вміст розділені на дві незалежні складові, тобто власник бізнесу чи робітник компанії, відповідальний за наповнення сайту, може просто вставляти зміст у призначені форми, а система вже потурбується про вставлення контенту у відповідні місця у шаблоні.

Найпопулярнішими умовно безкоштовними CMS є WordPress, Joomla, OpenCart, Drupal.

CMS не вимагають навичок програмування для роботи, у них вже є усі необхідні функції, значно заощаджують час та бюджет на розробку та її результат легко прогнозувати, якщо сайт стандартний. Але, нажаль, безкоштовні CMS небезпечні, у них немає технічної підтримки та документації. Створення нових функцій витратне. Деякий функціонал так і не використовується, але роботу сайту обтяжує.

Деякі з вищенаведених недоліків вирішуються кастомними CMS – CMS, які дозволяють обирати функціональність та доробляти його. Підходить для складних проєктів або проєктів з нетиповими рішеннями.

WordPress – найпопулярніша CMS, на якій функціонують як маленькі блоги та інтернет-магазини, так і найпопулярніші вебсайти світу, такі як Forbes, The New Yorker, Vogue, People Magazine та The Wall Street Journal. На початку, у 2003 році, WP був створений для ведення блогів Меттом Муленвегом та Майком Літтлом. Але можливості «двигуну» з часом зростали, цим самим перетворюючи CMS на найпопулярніший у світі потужний універсальний конструктор для сайтів будь-якого виду та складності. При всьому цьому знати мови програмування хоча і буде плюсом, але зовсім не обов'язково. Дуже велика частина цієї CMS абсолютно безкоштовна. Має багато різноманітних плагінів та тем. Інтернет переповнений навчальними матеріалами з нього. Створити сайт можна з одними лише навичками користування комп'ютером. Має інтуїтивно зрозумілий інтерфейс. Але навіть ця система не без своїх недоліків. Нажаль, репутація з безпеки у WordPress не бездоганна. Модулі безпеки можуть у деяких випадках вирішити цю проблему, але не завжди. Чим більша кількість розширень на сайті, тим повільніше він буде завантажуватися. Є ризик натрапити на неякісний модуль. CMS, на відміну від

конструкторів, не попіклується про хостинг, оновлення, безпеку та інші нюанси. Це все лягає на плечі власника бізнесу або співробітника компанії, який за це відповідальний. Хоча, як сказано вище, велика частина CMS безкоштовна, повністю безкоштовним сайт на ній не буде. Модулі безпеки, захисту від спаму, хостинг та дизайн можуть немало коштувати.



Рисунок 1.3 – Логотип WordPress

Joomla – CMS з відкритим кодом, на якій можна створити майже будь-який сайт: від лендінгів до інтернет-магазинів та вебдодатків з розгалуженою структурою. Основні можливості системи доступні «з коробки», але якщо цього недостатньо, то на вибір є багато різноманітних плагінів для побудови різних типів сайту. Для створення проєктів за допомогою цього «двигуну» зазвичай не потрібне знання мов програмування, тому Joomla підходить для тих, хто займається створенням свого сайту самостійно, без розробників. Вона майже повністю безкоштовна: треба буде заплатити лише за деякі модулі. Інтерфейс панелі адміністрації зрозумілий. Її легко встановити. Кешування раніше відкритих сторінок дозволяє прискорити завантаження сторінок. Є велика кількість шаблонів у вільному доступі. Безпеку теж можна назвати її сильною стороною. Хоча й уразливість, як й у будь-якої іншої CMS, у неї є, сайти на Joomla зламують рідко. Вбудований режим зневадження допомагає швидко ліквідувати збої. Joomla – SEO-дружня CMS. Вона добре адаптована під пошукові системи та у неї є багато розширень для SEO-просування. Але не позбавлена Joomla й недоліків. Перший з них – багато розширень та шаблонів для цієї системи платні. Другий – немає технічної підтримки, оскільки CMS безкоштовна. Але спільноти та інструкції допоможуть знайти відповіді на різні питання з неї. Третій – розширення з сумнівного джерела часто є причиною зламувань.



Рисунок 1.4 - Логотип Joomla

Drupal – безкоштовна система керування змістом з відкритим вихідним кодом, на якій можна створювати різні типи проєктів з різною складністю. На цьому двигуні працюють навіть міжнародні проєкти, такі як сайт Уряду Франції (<https://gouvernement.fr/>), сайт Сенату штату Нью-Йорк (<https://nysenate.gov/>), офіційний сайт гри «Зоряні війни» (<https://swtor.com/>), платформа гри WarFace (<https://warface.com/>). Цю CMS обирають багато розробників за гнучкість налаштування та простоту використання. І хоча передостаток функцій може збити з пантелику початківця, інтерфейс у Drupal зрозумілий. А щоб не плутатись у налаштуваннях, недосвідчені користувачі можуть завантажити збірку. Як і у Joomla, у Drupal є своя спільнота, форум, де розробники діляться досвідом та допомагають один одному. На сервер можна завантажити ядро CMS або збірку – ядро з набором розширень. CMS можна встановити в один клік з акаунту хостингу. У ній є велика кількість шаблонів. Вебсторінки кешуються, що сприяє більш швидкому їх завантаженню. Але є недолік – модулі можуть виглядати недоробленими через те, що для того, щоб працював один модуль, треба встановити декілька розширень. До того ж, у них треба розбиратися. Ще один мінус - нестабільність роботи після оновлень через те, що не усі модулі отримують оновлення разом з Drupal. Тому після виходу оновлення системи краще трішки почекати перш ніж переносити свій вебдодаток на нову версію CMS. Третій мінус – вебдодатку на Drupal потрібно багато ресурсів – потужний хостинг, інакше навіть з ввімкненим кешуванням сторінки будуть завантажуватися довго.



Рисунок 1.5 – Логотип Drupal

### 1.2.3 Мови програмування та фреймворки

Фреймворк – це набір бібліотек, який полегшує розробку проєкту. Для створення сайту за допомогою фреймворків потребуються навички програмування, порівняно багато часу та коштів, але такі сайти набагато краще за ті, що розроблені за допомогою попередніх рішень, тому воно того варте. Таке рішення дозволяє не обмежуватись функціональністю «з коробки»: прибирати функції які не потрібні та додавати власні, унікальні. До того ж, сайти, побудовані на фреймворках, мають вищу швидкість завантаження та краще оптимізовані.

Є ще один варіант – розробляти проєкт чистою мовою програмування, але це потребує ще більше часу, ніж розробка на фреймворках. Плюси та мінуси у цих двох варіантів аналогічні.

Фреймворки є для різних мов програмування. А мови програмування можна умовно поділити на дві групи: для розробки клієнтської частини та для розробки серверної частини. Основні мови для розробки клієнтської частини – HTML, CSS та JS та похідні від них, для серверної – Python, PHP, Ruby, ще може використовуватись Java. Кожна з них має свою область застосування.

HTML – мова розмітки, яка використовується при розробці вебдодатків. Основні елементи HTML – теги з кутовими дужками. Майже усі вони парні, тобто є відкривальний тег, який показує, де починається елемент (наприклад, абзац, список або блок) та закривальний, щоб визначити, де елемент закінчується. Наприклад, щоб створити абзац, треба помістити текст між відкривальним тегом `<p>` та закривальним тегом `</p>`. HTML за допомогою тегів вказує браузеру, з яких елементів і в якому порядку збирати вебсторінку та де брати стилі та скрипти. Таким чином, можна передавати інформацію різноманітними способами: таблицями,

малюнками, списками тощо. Але його функціоналу мало, щоб оформити зовнішній вигляд елементів та змусити їх реагувати на дії користувачів. З цією метою використовують стилі CSS та мову JavaScript.

CSS (Cascading Style Sheets, каскадні таблиці стилів) – мова для візуального оформлення вебдодатків. За допомогою HTML можна тільки розмістити об'єкти на сторінці. Без CSS сторінка буде виглядати доволі погано, оскільки ця мова відповідає за розмір об'єктів, колір, фон, ступінь прозорості, розташування відносно інших елементів, візуальні ефекти при різних станах та переходах та багато іншого. У CSS дуже простий синтаксис, його не треба довго вчити. Але часом його можливостей може бути замало або опис потрібної властивості на CSS може бути складним, тому були розроблені похідні від нього мови, такі як Sass, SCSS, Less. Але вони все одно компілюються у чистий CSS.

Фреймворки CSS – це готові таблиці стилів. Вони користуються попитом у розробників та дизайнерів, оскільки дозволяють зробити стильне візуальне оформлення вебдодатку з меншими часовими витратами та зусиллями без хвилювань за кросбраузерність. При цьому не обов'язково володіти глибокими знаннями CSS, оскільки не треба робити усю роботу з оформлення вебсайту з нуля. Але є можливість комбінування власних стилів та стилів фреймворку, що підійде більш досвідченим розробникам. Таким чином, їх можуть використовувати розробники з різним досвідом: від початківця до гуру.

Bootstrap – найпопулярніший безкоштовний фреймворк. Розроблений співробітниками Twitter. На початку використовувався для самого додатку. Має відкритий вихідний код. Основна особливість даного фреймворку – концепція mobile-first адаптивного дизайну, яку він і просував. Крім того, що цей фреймворк адаптивний, він кросбраузерний. Вебдодатки, зроблені з використанням фреймворку, будуть однаково відображатися в усіх сучасних браузерах. Ним легко користуватися та можна легко навчитися ним користуватися завдяки його простоті та великій кількості навчальних матеріалів з нього в Інтернеті. Дозволяє створювати сайти в єдиному стилі, оскільки усі елементи фреймворку виглядають гармонічно один з одним. Бонус для розробників: код вебзастосунку з

використанням Bootstrap зрозумілий. Нажаль, не позбавлений Bootstrap і недоліків. Сайти, створені за допомогою фреймворку, схожі один на одного. Але вирішити проблему можна відмовою від готових рішень. Bootstrap може не підійти для реалізації деяких проєктів, оскільки у нього є обмеження. Третій недолік – його елементи погано виглядають у старих браузерях, бо він постійно оновлюється.

Tailwind CSS – utility-first фреймворк, який відрізняється від традиційних фреймворків, таких як Bootstrap, тим, що стилі він надає не для елементів, а для властивостей. Суть підходу utility-first полягає в тому, що стилі задаються не для елементів у цілому, а для класів-утиліт, за допомогою яких задається оформлення властивостей елементів, таких як кольору, розміру та інших. Така особливість може значно прискорити швидкість розробки та робить синтаксис Tailwind простим і зрозумілим. За допомогою Tailwind можна створювати унікальні інтерфейси, які відповідають вимогам проєкту. Не забули й про адаптивність, цей фреймворк надає потужні засоби для її реалізації. Ще одна «родзинка» Tailwind – теми оформлення. У нього наявні вбудовані та можна ще налаштувати власні. Нажаль з деякими з цих «родзинок» приходять і хиби. Наприклад, вищезазначений підхід, який реалізує фреймворк, може разом зі зрозумілим синтаксисом та прискоренням розробки принести відразу декілька проблем:

- незрозумілий в цілому код, у якому може бути важко визначити який саме це елемент за назвами класів;
- дуже велику кількість класів (і усі їх треба вчити);
- дуже велику довжину списку класів у розмітці HTML;
- деякі власні оригінальні рішення може бути складно та часовитратно реалізовувати.

Materialize CSS – розроблений Google фреймворк з відкритим вихідним кодом, який реалізує концепцію Material Design. Фреймворк містить безліч інтерактивних компонентів, які прискорюють розробку та допомагають покращити взаємодію з користувачами. Materialize адаптивний та дозволяє створювати стильні, сучасні та прогресивні інтернет-застосунки. У ньому є можливість зручно реалізувати власні оригінальні ідеї. Але, як і у будь-якого фреймворку, у нього є свої недоліки.

Materialize «важкий», тобто має великий розмір. У ньому нема flexbox, який дозволив би зручніше розміщувати елементи відносно один одного. Шаблони вебсайтів у колекції одноманітні. І старі браузері він не підтримує, хоча це може бути й не важливим.

Foundation – фреймворк з відкритим вихідним кодом, який дозволяє створювати адаптивні вебдодатки. Це ідеальний вибір для досвідчених розробників, яким подобається свобода дій. З використанням цього фреймворку розроблялися такі платформи, як Facebook, eBay і Mozilla. У нього немає власного стилю: компоненти мінімалістичні та легко налаштовуються. Є найнеобхідніші компоненти: панелі навігації, кілька типів контейнерів та зручна для розробників сітка. За допомогою Foundation for Emails можна створити адаптивні шаблони електронних листів для будь-якого клієнта, в тому числі й для старих версій Microsoft Outlook. Можна робити переходи й анімацію з використанням вбудованих ефектів за допомогою Motion UI. Можливостей багато та новачок може у всьому цьому заплутатися та йому буде важко працювати з фреймворком, що є першим недоліком Foundation. До того ж, у нього є залежність від JQuery чи Zepto (бібліотека з синтаксисом як у JQuery, але менше за розміром).

JavaScript – це високорівнева мова програмування, яка додає сайтам інтерактивності. За допомогою цієї мови можна робити анімації, реагувати на дії користувача, валідувати форму та додавати спливаючі повідомлення. Вона популярна, її використовують у багатьох проєктах та компаніях. Виконується в реальному часі у браузері, не потребує попередньої компіляції, може виконувати асинхронні операції. Чудово інтегрується з HTML та CSS. Для JavaScript є велика кількість бібліотек та фреймворків. З ним просто працювати програмістам завдяки динамічній типізації та автоматичному видаленню невикористовуваних об'єктів (збиранню сміття). Ця мова може виконуватися на різних платформах та пристроях. Має велику спільноту. Але при усіх цих перевагах, JS не позбавлений недоліків. По-перше, динамічна типізація може ускладнити підтримку масштабних проєктів. По-друге, не усі браузері обробляють JS однаково. По-третє, JS буває менш продуктивним, особливо у високонавантажених системах. Четвертий недолік –

може бути складнуватий для початківців, але терпіння та праця все перемагають. Далі роздивимось найпопулярніші JS-фреймворки.

Angular – JavaScript-фреймворк, розроблений Google у 2010 році. Дозволяє створювати масштабовані вебдодатки завдяки своїй MVC-архітектурі, яка робить розробку та тестування швидкими та зручними. Він надає зручні та потужні інструменти для розробки складних вебдодатків, такі як компоненти, двостороннє зв'язування, сервіси, маршрутизацію та багато іншого. Підтримка фреймворку такою компанією як Google гарантує його стабільність, актуальність та довгий строк підтримки. Але у Angular є деякі недоліки. По-перше, він надлишковий для маленьких проєктів. По-друге, для того, щоб його вивчити, потрібен час та терпіння, оскільки він складний для початківців. По-третє, збірка вебдодатків на Angular може бути великою.

React – бібліотека для створення користувацьких інтерфейсів. Розроблена Facebook у 2013 році та швидко набула популярності серед розробників. Завдяки віртуальному DOM React дозволяє розробляти відгучливі інтерфейси, бо оновлюється не уся сторінка, а тільки змінений елемент. Односторонній потік даних полегшує відлагодження та тестування застосунків завдяки простоті та передбачуваності управління станом додатків. У React велика спільнота та велика кількість бібліотек та інструментів для розробки. Але, нажаль, і доволі великих недоліків він не позбавлений. По-перше, він дає тільки можливість працювати з представленням, у нього немає багатьох можливостей повноцінного фреймворка, таких як відмінна від нативних система подій, робота з AJAX, шару даних, Promises та ін. І при цьому ця бібліотека доволі велика. По-друге, структура документації може заплутати.

Vue – прогресивний JS-фреймворк, який дозволяє створювати користувацькі інтерфейси. Він легковажний та простий у вивченні та використанні навіть для новачка. При цьому у нього функціонал достатній для розробки складних проєктів. Фреймворк гнучкий: можна обирати тільки потрібні для проєкту функції. Має хорошу документацію та велику спільноту. З мінусів можна виділити те, що їм займається спільнота та творець фреймворку з його командою, тобто він не має такої

грунтовної підтримки та стабільності, як от React або Angular. Другий недолік – кожен великий випуск вносить несумісність з попередніми версіями зміни.

Крім JavaScript є ще декілька похідних від неї мов, які компілюються у код цієї мови. Перша – TypeScript – мова, яка є надбудовою над JS. Була розроблена компанією Microsoft у 2012 році. Дозволяє створювати більш надійні та масштабовані додатки. На відміну від JS, типізація у TS статична, що полегшує командну роботу та робить вебдодатки зроблені з його використанням більш надійними та стабільними. Крім цього, TypeScript додає до JavaScript ще деякі додаткові можливості. Є у TypeScript і декілька недоліків. По-перше, у деяких випадках розробка та підтримка коду можуть займати більше часу. По-друге, деякі інструменти можуть не підтримувати TypeScript. По-третє, підтримка деяких сучасних можливостей JavaScript може бути обмеженою. Фреймворків чисто для TypeScript немає, але вона може добре інтегруватися в JavaScript-фреймворки.

CoffeeScript – маленька мова, яка трансліюється у JavaScript. Вона, по суті, є синтаксичним цукром над JS, тобто це значить, що вона дозволяє писати коротший код ніж JavaScript, який легше читається. У неї дуже компактна та наочна документація. Але актуальність цієї мови зараз знижується через появу ES6 – найновішого стандарту JavaScript, у який було введено багато функцій CoffeeScript. Також деякі бібліотеки JavaScript можуть бути несумісними з цією мовою. Як і у TypeScript, у неї теж нема власних фреймворків. Але якщо її скомпілювати, то вона працює з фреймворками JS.



Рисунок 1.6 – Логотип CoffeeScript

Dart – універсальна мова програмування, створена Google у 2011 році, яка позиціонується як заміна JavaScript. На ньому можна писати як вебсайти та мобільні

додатки, так і утиліти командного рядка та серверні програми. Головна перевага Dart – швидкість праці написаних з його використанням застосунків. Крім цього, додатки з Dart будуть стабільніші, бо ця мова строгіша та не пропустить більшість помилок, властивих JavaScript. Якщо знати JS або Rust, то опанувати її легко. У мови є зрозуміла документація. Інструменти для розробки на ній доступні, також вона підтримується більшістю поширених IDE. Але, нажаль, поки що Flutter – це єдиний великий фреймворк для Dart, тому спільнота цієї мови не дуже розширюється. Також, хоча Dart і кросплатформений, розробникам для того, щоб працювати на Flutter, доведеться ознайомитися з особливостями різних платформ. Це найбільша складність, з якою вони стикаються при переході на даний фреймворк.



Рисунок 1.7 – Логотип Dart

Далі поговоримо про мови для розробки серверної частини вебдодатків. У розробку серверної сторони входить взаємодія з базами даних та видача даних з них на клієнтську сторону, створення API (це те, що визначає, як різні частини ПЗ взаємодіють один з одним) та інша серверна логіка.

RНР – розповсюджена мова загального призначення з відкритим вихідним кодом, який найчастіше використовують як серверний інструмент для генерації HTML-коду. Як вказано у джерелі [39], «На сьогодні з її використанням як мови для реалізації серверної логіки працюють близько 80% вебдодатків. Вона входить у топ-5 найбільш поширених мов програмування серед українських розробників.» Така популярність обумовлена простим синтаксисом, кросплатформеністю, відмінною маштабованістю додатків, написаних на ньому, відмінною інтеграцією в HTML-документи, детальною документацією, великою спільнотою та кількістю різних навчальних матеріалів та широкими перспективами її подальшого розвитку. Але, незважаючи на ці плюси, розробники обирають інші мови через наявність деяких недоліків у безпеці, націленість на веброботку та деякі протиріччя та

неузгодженості синтаксису, такі як різні назви вбудованих функцій та елементи з різних мов.

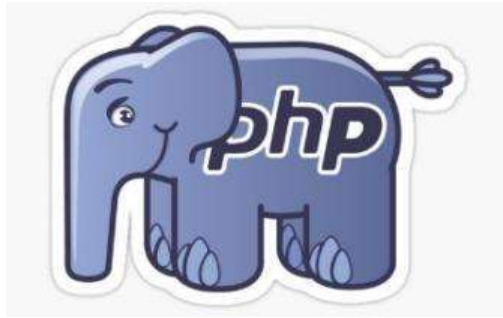


Рисунок 1.8 – Символ PHP – слон

Хоча на PHP програмувати легко, зі збільшенням розміру та складності проєктів збільшується і кількість рутинних та одноманітних завдань, які забирають багато зусиль та часу. Вирішити цю проблему допоможуть PHP-фреймворки, які зроблять процес розробки швидким, ефективним та захоплюючим. Вони дуже сильно спрощують життя розробників та відкривають безмежні можливості для створення вражаючих вебдодатків.

Laravel – один з найпопулярніших PHP-фреймворків, який дозволяє створювати потужні, елегантні та масштабовані вебпроєкти без головної болі. Свою репутацію він заробив завдяки декільком перевагам. По-перше, у нього простий синтаксис та багато функцій та можливостей. Крім цього, завдяки простому синтаксису, його легко вивчити новачку. Є детальна, дуже добре структурована та зрозуміла документація. Він має величезну спільноту з мільйонів розробників. З цього виходить, що і різноманітних курсів, статей, тренінгів та іншого з нього у Інтернеті хоч греблю гати. Ще програміст, якщо забажає, може розширити його функціонал шляхом додавання сторонніх пакетів та бібліотек. Завдяки всьому цьому він робить розробку вебдодатку дуже швидкою. Але деяким розробникам можуть не сподобатися великий розмір фреймворку та дуже велика кількість різних залежностей.

Symfony – потужний PHP-фреймворк, який пропонує широкі можливості для розробки вебдодатків. Він здатний забезпечити гнучкість та адаптивність для різних

типів вебдодатків різної складності, у фреймворку є потужна структура та усі необхідні компоненти та бібліотеки для спрощення розробки. При бажанні можна і розширити функціонал Symfony, оскільки він підтримує багато бібліотек та пакетів. Але, нажаль, порівняно з іншими фреймворками у нього доволі складний синтаксис. Через це на ньому може бути важче розробляти та новачкам його може бути важче вчити.

CodeIgniter – це легковажний фреймворк PHP, який використовується для швидкої розробки вебзастосунків. У нього «з коробки» є бібліотеки для реалізації основних функцій. Вони не залежать один від одного, що спрощує розробку. Фреймворк дуже мало важить: всього близько 2 Мб. Завдяки цьому він встановлюється набагато швидше ніж інші фреймворки. Другий ефект від цього – підвищення продуктивності та швидкості сайту, розробленого з використанням CodeIgniter. Але вищеперечислені плюси надаються за рахунок обмеженого набору функціональності порівняно з іншими фреймворками та відсутності вбудованої підтримки деяких розповсюджених функцій.

Yii – високопродуктивний PHP-фреймворк, який дозволяє швидко розробляти сучасні додатки. Він може бути використаний для розробки сайтів різних типів та рівнів складності. А завдяки наявності кешування та компонентній структурі він особливо підійде для великих та складних застосунків. Надає зручне генерування коду, що допомагає прискорити розробку додатків. Фреймворк добре підтримує різні розширення, завдяки чому можна додавати нову функціональність без головної болі. Він підтримується сильною командою та великою спільнотою. Нажаль, ця спільнота не така велика, як у Laravel або Symfony. Частково це пов'язано з тим, що деякі розробники вважають що він менш гнучкий порівняно з іншими фреймворками.

Крім PHP на серверній стороні ще найчастіше використовуються Ruby та Python.

Python – високорівнева друга за популярністю у світі мова програмування з відкритим вихідним кодом, яка може бути використана за багатьма призначеннями: розробка вебзастосунків, мобільних застосунків, програмного забезпечення інших

напрямків, машинне навчання, ігри, автоматизація тестування та іншими. Мова програмування вирізняється простим та зрозумілим синтаксисом. Через це його дуже легко вивчити новачку. Python стандартизований: навіть є посібник зі стилю програмування на ньому. Це допомагає розробникам легше працювати в команді. Також ця мова однаково працює на різних платформах. На ній є багато бібліотек. Але, попри такі плюси, він має декілька проблем. Перша з них – швидкість програм на ньому, яка не дуже висока через те, що код на цій мові порядково інтерпретується, плюс динамічна типізація. По-друге, Python для керування типізацією та автоматичним збиранням «сміття» споживає більше пам'яті. По-третє, з многопоточними програмами на Python можуть бути проблеми через GIL (Глобальне Блокування Інтерпретатора). GIL – особливість, яка дозволяє тільки одному потоку керувати інтерпретатором цієї мови. У результаті виходить доволі значна проблема з продуктивністю. Крім цього, Python може бути менш ефективним у деяких областях, таких як мобільна розробка та розробка ігор.



Рисунок 1.9 – Логотип Python

У Python є велика екосистема фреймворків для різних напрямків розробки, які надають розробникам готові інструменти та рішення для прискорення розробки. Але, так як завданням є створення онлайн-магазину, поговоримо про фреймворки, які використовуються для веброботи.

Django – один з найпопулярніших, найстаріших та повнофункціональних фреймворків для розробки вебдодатків на Python. Він надає розробникам інструменти для розробки вебзастосунків різної складності. Основні можливості Django включають керування базами даних, аутентифікацію користувачів, маршрутизацію URL та багато іншого. Цей фреймворк також відомий своєю

простотою синтаксису, хорошою документацією та великою та активною спільнотою, що робить його популярним вибором для розробників усіх рівнів. Але його широка функціональність може бути надлишковою для маленьких проєктів.

Flask – легкий та гнучкий додаток для розробки сайтів на Python. Призначений для розробки додатків малої та середньої складності. Фреймворк надає базовий набір інструментів, а далі дає розробникам можливість розширювати згідно з потребами. Може стати ідеальним вибором завдяки своїй легкості та гнучкості, мінімалістичному дизайну та простому синтаксису. Але він не підходить для складних та великих проєктів.

Web2py – безкоштовний Python-фреймворк з відкритим вихідним кодом для розробки вебзастосунків. Надає готові рішення для розповсюджених завдань. Не потребує спеціального встановлення або налаштування. На ньому можна створювати не тільки сайти, а й API. Нажаль, у ньому недостатньо можливостей для розробки складних вебдодатків. А ще він може бути несумісний з деякими технологіями або фреймворками.

Pyramid – легкий, швидкий та простий фреймворк для розробки вебдодатків на мові Python. Одна з його переваг – гнучкість. Він дозволяє розробникам обирати тільки необхідні компоненти згідно з власними потребами не обмежуючи їх. Має просту архітектуру. На ньому зручно створювати власні рішення. Фреймворк підходить для розробки застосунків різної складності: від простих до великих і масштабованих. Але за гнучкість доведеться платити більшим часом налаштування та інтеграції компонентів порівняно з іншими фреймворками.

Підійшли до останньої мови розробки серверної сторони – Ruby. Ruby – високорівнева мова програмування, яка була розроблена у Японії у кінці 80-х років. Її легко вивчити новачку завдяки її простому синтаксису. Підтвердженням цьому є філософія цієї мови: «програмування має бути задоволенням». З цього виходить, що Ruby людиноорієнтована мова, що дозволяє створювати на ній красивий та чистий код, що дуже цінується у спільноті, яка, до речі, велика і активна. Мова ідеально підходить для розробки високопродуктивних додатків завдяки багатопоточності. та доступна на різних системах. На Ruby розроблено багато бібліотек для різних

завдань, що дозволяє не писати код з нуля. Але, нажаль, має Ruby й недоліки. Перший з них – швидкість виконання та продуктивність програм на Ruby, які можуть бути занадто низькими для завдань з великою кількістю обчислень. А не зовсім якісні бібліотеки можуть ще сильніше це проявити. По-друге, ця мова не може використовуватися при обмежених ресурсах через доволі велике споживання пам'яті. По-третє, багатопоточність неможлива у деяких сценаріях, тому можуть виникнути проблеми при створенні додатків з високим ступенем паралелізму або ефективним керуванням ресурсами. Тому ця мова може вважатися непридатною для систем з високими вимогами до продуктивності, швидкості та оптимізації. Але, незважаючи на все це, Ruby залишається популярною мовою програмування у багатьох галузях, особливо у веброботці та стартапах.



Рисунок 1.10 – Логотип Ruby

Ruby on Rails – фреймворк на Ruby для веброботки, створений у 2004 році Девідом Ханнемейером Ханссоном. На ньому можна створити вебзастосунок будь-якого типу: керування проєктами, груповий чат, безпечне сховище даних чи просто блог. Одна з його головних особливостей – зменшення кількості явних конфігурацій шляхом встановлення угод з іменування та розміщення коду. Це допомагає спростити код, але за це треба платити зменшенням гнучкості фреймворку. Він надає безліч інструментів та бібліотек для прискорення створення вебдодатків. Він також працює на будь-якій ОС та підтримує безліч систем керування базами даних. Він часто оновлюється та змінюється. З одного боку, покращення це добре, але новачкам важко адаптуватися до середовища яке часто змінюється. Також у RoR зовсім не найкращі швидкісні показники, такі як швидкість завантаження, особливо при роботі з великими проєктами, та швидкість виконання, яка помітно менше ніж у,

наприклад, Node.js. А помилки проєктування тільки посилюють цей недолік, тому їх ціна у даному випадку доволі висока.

Sinatra – безкоштовний Ruby-фреймворк для розробки вебдодатків з відкритим вихідним кодом. Основна ідея – швидке створення застосунків на Ruby без зайвих зусиль. Фреймворк простий у використанні, його легко вивчити та з ним легко працювати. Навчитися розробляти на Sinatra можна дуже швидко, для цього не потрібно ніяких інших великих знань, крім мови Ruby. До того ж, він майже не змінюється, тому новачкам до нього легше адаптуватися. Він гнучкий, він не нав’язує якусь визначену структуру проєкту, а розробник може сам обирати потрібні йому компоненти. Добре поєднується з іншими бібліотеками та фреймворками, що дозволяє інтегрувати його у більші вебпроєкти. Застосунки на цьому фреймворку володіють високою продуктивністю. Але, нажаль, у Sinatra за усі вищеперечислені переваги розробникам доводиться розплачуватися обмеженою функціональністю, тобто у фреймворку немає стільки функціональності, як, наприклад, у Rails, що завадить створенню складних проєктів. Крім цього, у фреймворку немає стандартних рішень для деяких завдань, тому розробникам доводиться вигадувати власні.

Hanami – сучасний, швидкий та легковажний фреймворк для створення вебзастосунків на Ruby. Є безпечним і в той же час гнучким каркасом та дозволяє швидко розробляти застосунки які потім легко за потреби змінити. До січня 2016 року носив назву «Lotus». Використовує на 60% менше пам’яті ніж інші Ruby-фреймворки [51]. Основна концепція Hanami – чиста архітектура, що розділює додаток на «шари» і зменшує залежність між ними. Підходи, які використовує фреймворк, допомагають розділити відповідальності між компонентами застосунку, що спрощує розробку, тестування та підтримку проєкту. Hanami дозволяє використовувати як функціональне, так і об’єктно-орієнтоване програмування. Одне з двох можна обирати згідно з потребами проєкту. Додатки на Hanami модульні, що полегшує додавання функціональностей, яких бракує, та прибирання непотрібних. Фреймворк має інструменти для уникнення загроз безпеки. Недоліки такі ж самі, як і у Sinatra.

### 1.3 Вибір оптимальних технологій для інтернет-магазину «Fashion Art»

Далі треба обрати рішення, оптимальне для саме цього проєкту. Вибір рішення залежить від декількох чинників: масштаб, тип та складність проєкту та бажана швидкість розробки. За складністю проєкти можуть умовно поділятися на три категорії: прості, середні, та складні. До простих проєктів належать візитки, лендінги та інші прості додатки з відвідуваністю до мільйона користувачів на місяць, до проєктів середньої складності – інтернет-магазини, торгові майданчики, масштабні портали, різні сервіси та інші просунуті додатки з відвідуваністю в мільйони, до складних – величезні портали, соціальні мережі, інші інноваційні та нетипові рішення з відвідуваністю в сотні мільйонів та мільярдів користувачів. Прості проєкти зазвичай розробляються на SaaS-платформах та CMS, проєкти середньої складності – на фреймворках, складні – на чистій мові.

Таким чином, інтернет-магазин – проєкт середньої складності, тому він буде розроблятися на фреймворках. Вирішено, що сайт буде мати деревоподібну структуру. А також ми ознайомилися з рекомендаціями щодо проєктування інтерфейсу користувача та правилами, за якими будується структура сайту онлайн-магазину.

Спочатку визначимося з тим, на чому будемо розробляти на стороні клієнта. Для оформлення зовнішнього вигляду сторінок буде використаний Bootstrap. Він допоможе швидко та з мінімальними зусиллями створити якісний, зручний, сучасний та адаптивний інтерфейс. Його можливостей вистачить для розробки усіх елементів сайту. При цьому код буде коротким та зрозумілим. Також цей фреймворк абсолютно безкоштовний, що дасть можливість зменшити витрати на проєкт. Як JS-фреймворк використаємо Vue. Він найкраще підійде для вебзастосунку середнього розміру та складності, оскільки Angular у даному випадку може бути надлишковим, а у React може не вистачити функціональності, оскільки він працює переважно з представленням. Цим фреймворком легко користуватися завдяки його простому синтаксису. Компонентний підхід дозволить розділити інтерфейс на частини. Для

Vue легко знайти бібліотеку для реалізації будь-якого завдання. Однією з причин є активна спільнота. Вона також може допомогти знайти відповідь на будь-яке питання по Vue. Це все дозволяє спростити процес розробки та добре зекономити час. Інтерфейси на Vue будуть швидкі та відгучливі, що точно сподобається користувачам та, у віддаленій перспективі, пошуковим системам. Така перевага отримується завдяки системі оновлень фреймворку, яка змінює тільки той компонент, який треба, а не оновлює усю сторінку. А компонентом може бути й уся сторінка, наприклад, каталог товарів. Таким чином, перехід між сторінками теж відбувається без перезавантаження сайту у браузері. Далі перейдемо до вибору мови та фреймворку для розробки обчислювальної логіки нашого магазину. Як мову оберемо PHP. Він має простий синтаксис, велику та активну спільноту, багато бібліотек для вирішення різних завдань та фреймворків для зменшення кількості рутинної роботи, був створений для веброзробки та заточений під неї, що дозволяє програмістам не мати зайвої головної болі та економить їхній час. Підтримує велику кількість СКБД, що важливо для роботи з даними у інтернет-магазині. PHP володіє хорошою продуктивністю та швидким часом виконання, що дозволить стороні клієнта швидше отримувати відгук від сервера, завдяки чому підвищиться швидкість роботи сайту. Усе це робить її відмінною мовою для онлайн-магазину. Мова обрана. Добре. Але якщо розробляти на чистій мові то доведеться створювати шляхи уникання великої кількості рутинної роботи, що вже зробили за нас наявні фреймворки. Тому нема сенсу вигадувати велосипед, обираємо фреймворк згідно з нашими потребами. Ним буде Laravel. На ньому можна побудувати проєкт будь-якого типу: від порталу з новинами до інтернет-магазину. При цьому не відбувається обтяжування застосунку непотрібними функціями, які споживають ресурси, а користі не приносять. Можна легко втілити будь-яку функціональність, змінити дизайн, реалізацію елемента, тобто фреймворк надає великі можливості кастомізації. У Laravel дуже велика та активна спільнота. З цього виходить, що і пакетів для нього безліч. На будь-який смак для виконання будь-яких завдань. Фреймворк надає захист від загроз безпеки, таких як XSS-атаки та SQL-ін'єкції. Потенційно небезпечний HTML-код екранується, після чого його не можна

виконати. Власні методи обробляють SQL-запит таким чином, щоб у них не було того, що може нашкодити. Застосунки на Laravel швидко завантажуються завдяки технології кешування. Це підвищує лояльність користувачів та показники в пошукових системах. Крім цього з коробки Laravel надає багато функцій, такі як аутентифікація та авторизація, можливість зробити декілька мовних версій вебсайту, працювати з поштою та повідомленнями, кешування, розробникам надає потужні інструменти тестування та валідації та багато іншого. Підсумовуючи, можна сказати, що Laravel дозволяє швидко та легко розробити потужний, масштабований та швидкий вебдодаток, який принесе користувачу приємний досвід.

Після того як ми розібралися з вибором технологій для нашого інтернет-магазину, можемо перейти до формулювання вимог до нього.

#### **1.4 Розробка вимог до програмного забезпечення**

Важливим етапом проєктування ПЗ є розробка вимог до нього. Будемо починати з загальних вимог, а потім переходити до більш конкретних.

Перечислимо основні загальні вимоги до Інтернет-магазину:

- він повинен забезпечувати основні функції: додавання в кошик, оформлення замовлення, оплата, керування каталогом товарів тощо;
- інтерфейс сайту та процес купівлі повинні бути інтуїтивно зрозумілими та простими;
- інтернет-магазин повинен забезпечувати захист персональних даних покупців та безпечність платежів;
- сайт повинен бути адаптований для перегляду на різних пристроях: ноутбуках, планшетах, смартфонах.

Тепер детальніше розпишемо усі вимоги для інтернет-магазину.

По-перше, сайт повинен мати зручний інтерфейс та забезпечувати усе необхідне для здійснення найголовнішого призначення – покупки. Сайт повинен

добре відображатися та бути зручним у використанні на приладах різного розміру. Повинен бути реалізований пошук товарів за різними характеристиками, фільтрація результатів пошуку та підказки у полі пошуку для зручності користувачів. Обов'язкова можливість залишати та переглядати відгуки до товару. Треба реалізувати додавання товарів у кошик з можливістю їх видалення або змінення їхньої кількості, розрахунок вартості замовлення з усіма додатковими зборами, вибір варіантів доставки та оплати, перевірка наявності товарів, можливість оформлення замовлення без реєстрації. Процес оформлення замовлення повинен бути простим та зрозумілим з вказівкою адреси доставки, метода сплати та інших необхідних даних. Для успішної купівлі реалізація інтеграції з платіжними системами є невід'ємною складовою. Для розрахунку повної вартості замовлення та відстежування посилок повинна бути реалізована інтеграція зі службами доставки. Сайт повинен пропонувати різні варіанти сплати: кредитні карти, електронні платіжні системи або готівкою при отриманні. Повинні бути доступні різні способи доставки: кур'єрська доставка, поштова служба або самовивіз з пунктів видачі. Треба реалізувати повідомлення про стан замовлення, підтвердження замовлення та відстежування доставки. Крім цього, можливість зручно зв'язатися з менеджерами через онлайн-чат, пошту або телефон («зателефонуйте мені» і т.п.) може бути необхідною багатьом відвідувачам/клієнтам для обговорення різних питань. В наявності повинна бути інформація про політики повернення та обміну, відповіді на поширені запитання.

По-друге, необхідно зробити так, щоб клієнтам захотілося повертатися у магазин знову і знову. Можливість реєстрації користувача, як через інтерфейс сайту, так і через Google та соціальні мережі, збереження його історії замовлень та переглянутих товарів, відновлення та змінення паролю та інших персональних даних є головним для лояльності покупців до магазину. Закохають у магазин покупців також знижки, акції та спеціальні пропозиції. При цьому треба відображувати інформацію про них та умови їхнього використання.

По-третє, керування магазином повинне бути зручним для його власників та людей, відповідальних за це. Повинна бути можливість зручно видалити,

редагувати, створити та переглянути записи з БД категорій, продуктів та їхніх властивостей. Це буде забезпечувати адмін-панель. Додавання та редагування продуктів, тегів, кольорів і т.д. повинно відбуватися через форми, відображення – у таблицях, видалення - відбуватися після натискання на кнопку. Для комфортного управління важлива й правильна організація структури бази даних та взаємозв'язків між таблицями. Кожний товар має зв'язок «один до багатьох» з категорією (одна категорія – багато товарів), замовленням (одне замовлення – багато товарів) та відгуком (один товар – багато відгуків), «багато до багатьох» - з властивостями (кольором, тегом). Користувачі мають зв'язок «один до багатьох» зі своїми замовленнями (один користувач – багато замовлень). Крім цього, для відстежування відвідуваності та продажів треба зробити систему аналітики. Бажана система моніторингу, яка відстежує роботу сайту, інформує про збої і реагує на них, журналювання дій користувачів і адміністраторів для виявлення несправностей і аномалій. Вона зробить легшим життя розробників.

По-четверте, безпека – одна з найголовніших вимог. Покупець повинен бути впевнений в тому, що його дані будуть цілими та недоторканими. Для цього необхідно реалізувати безпечність передавання даних за допомогою протоколу HTTPS та SSL-сертифікату. Для безпечності оплат сайт повинен відповідати вимогам PCI DSS.

## **2 МОДЕЛЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

Розробники сучасних інформаційних систем постійно стикаються зі зростанням їхньої складності. Це пов'язано з підвищенням кількості вимог до систем, застосуванням складніших архітектурних рішень (наприклад, з переходом від простих дволанкових клієнт-серверних систем до розподілених багатоланкових) і як наслідок зі збільшенням обсягів програмного коду. У цих умовах розробка системи шляхом прямого переходу від вимог до програмування призводить до

величезної кількості помилок, у результаті такі проекти зазвичай залишаються незавершеними. Наразі вже досить ясно всім, що створення сучасних програмних систем неможливе без модельно-базованого підходу.

Модель програмного забезпечення створюється для того, щоб краще розуміти систему, що розробляється. Якщо система досить складна, іншого способу зрозуміти її як єдине ціле для людини просто немає.

Проектування програмних систем починається з побудови моделі, що описує систему з різних точок зору. Це означає, що повна модель складається з окремих проекцій, що відбивають різні аспекти системи. Вибір проекцій моделі залежить від підходу до проблеми та прийнятих рішень. Ключем до побудови проекцій є абстрагування, коли зосереджуються на найбільш істотних деталях, ігноруючи менш важливі. Надалі для стислості кожен проекцію ми називатимемо моделлю, вказуючи її тип. Наприклад, функціональна модель показує, що робить система у відповідь на запити користувача (але не як вона це робить), логічна модель описує основні сутності та відносини між ними.

Почну зі складання діаграм використання.

## **2.1 Діаграми варіантів використання**

Суть діаграми полягає в наступному: розроблена система представлена у вигляді сутностей або суб'єктів, які взаємодіють із системою, використовуючи випадки використання. У моєму випадку учасником може бути будь-яка організація, яка взаємодіє із системою ззовні. Це може бути людина, технічний пристрій, програма або будь-яка інша система і може виступати джерелом впливу на систему моделювання, визначену розробником. У свою чергу, випадки взаємодії використовуються для опису послуг, які система надає учасникам. Іншими словами, кожен випадок використання визначає певний набір операцій, які система виконує після попереднього спілкування з учасниками.

Учасниками можуть бути: користувачі інтернет-магазину, адміністратор (або особи з цими правами), система бази даних.

Варіанти використання можуть бути такими: авторизація, перегляд товару, пошук товару, додавання товару в кошик, оформлення замовлення, редагування даних та додавання товару адміністратором, відгуки-коментарі покупців та перегляд замовлень адмінами.

Приклад діаграми використання інтернет-магазину представлено на рисунку 2.1.

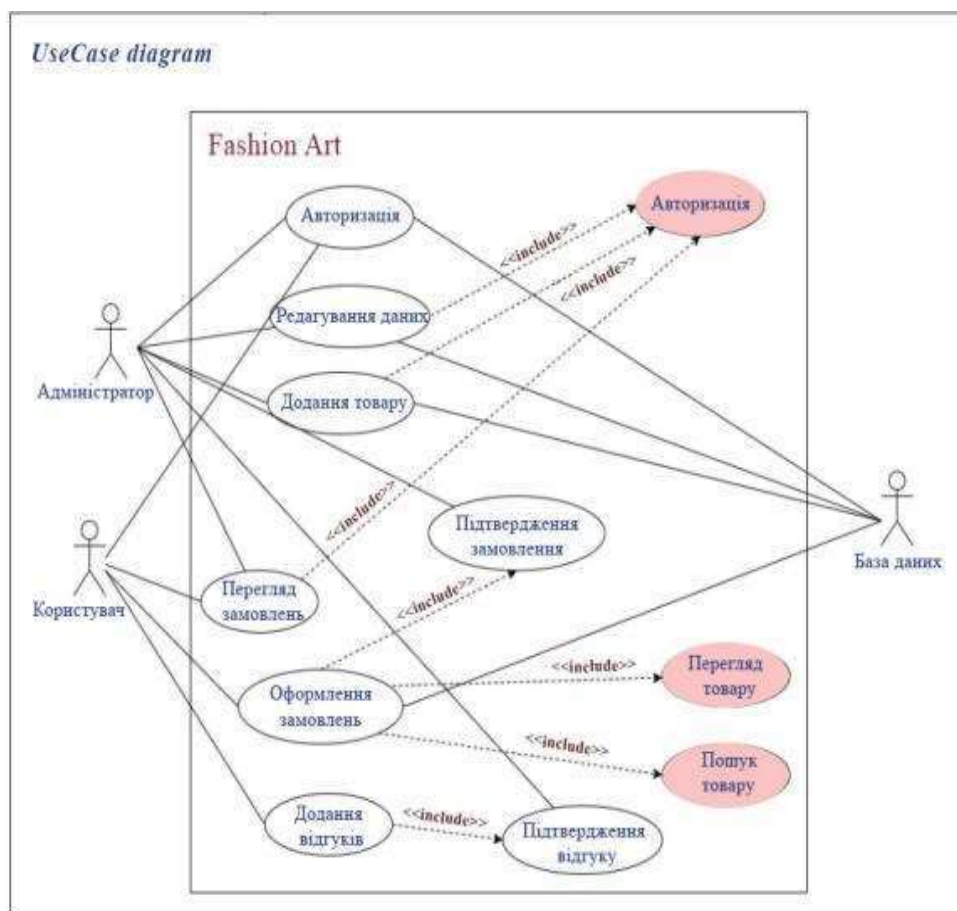


Рисунок 2.1 – Діаграма варіантів використання

Можна було б окремо створити діаграми варіантів використання для актора «Адміністратор» та для актора «Користувач». Але, як видно, ця діаграма поєднує використання системи цими двома суб'єктами. В цьому є раціональне зерно тому, що послуги та однакові операції перетинаються в одній і тій самій системі.

## 2.2 Діаграми послідовності

Завдяки діаграмі прецедентів, варіантів використання виявляються основні користувачі системи і завдання, які ця система повинна вирішувати. За допомогою діаграми діяльності описується послідовність дій для кожного прецеденту, це необхідно для досягнення поставленої мети.

Діаграма комунікацій – це спеціальна діаграма взаємодії, яка фокусується на обміні даними між різними учасниками взаємодії. Діаграма комунікацій не потребує відображати кожного учасника, як життєву лінію та відображати послідовність повідомлень вертикально, як діаграму послідовностей. Діаграми комунікацій для різних процесів у застосунку показані на рисунках 2.2 – 2.10.

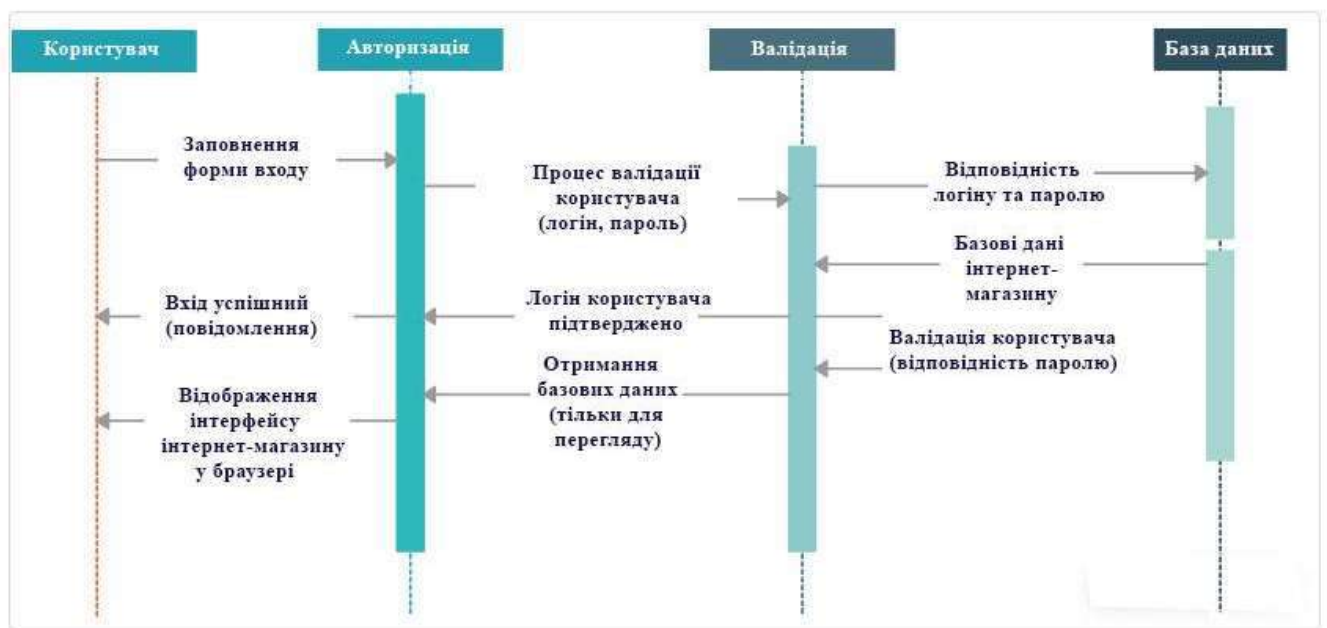


Рисунок 2.2 – Діаграма послідовності входу в систему

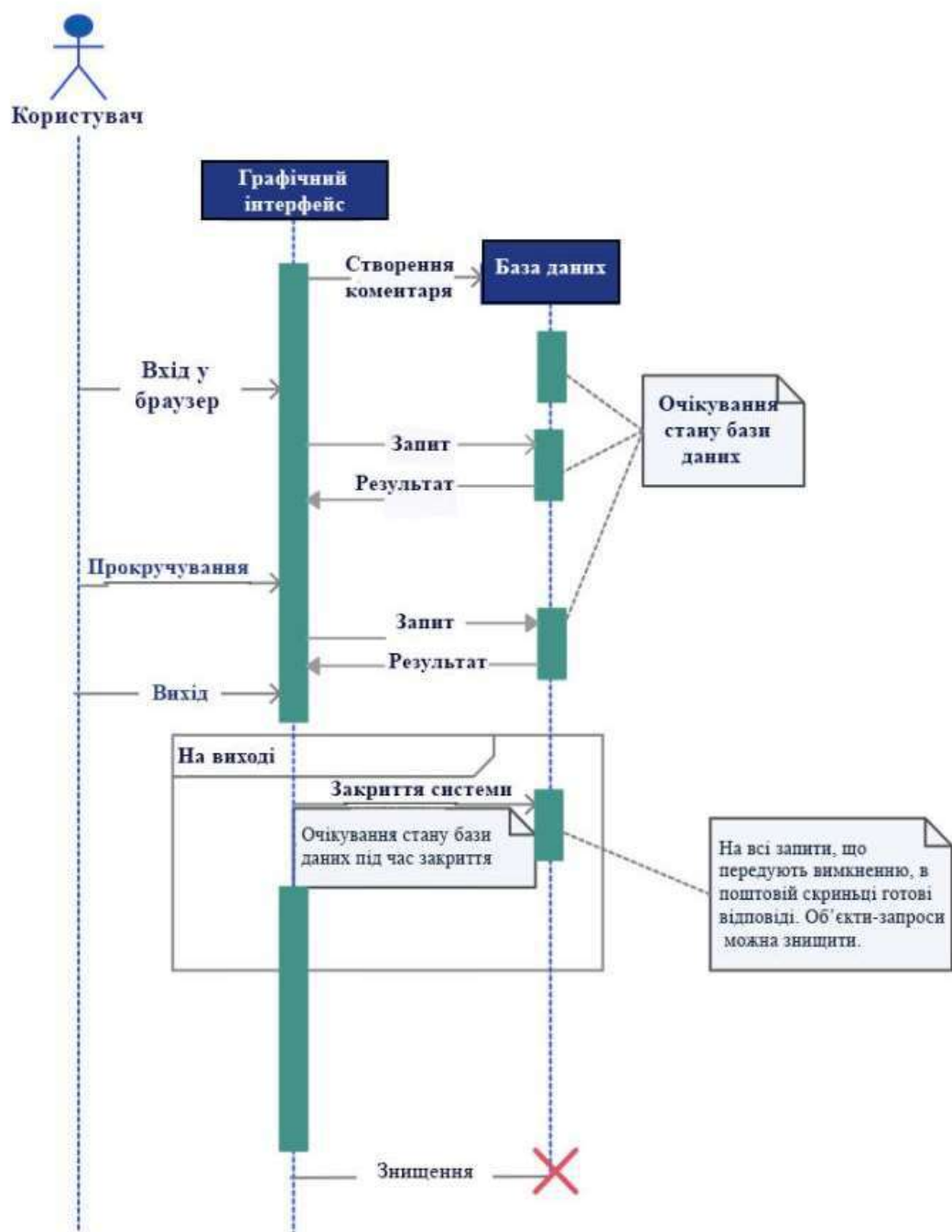


Рисунок 2.3 – Діаграма послідовності коментування (залишення відгуку)

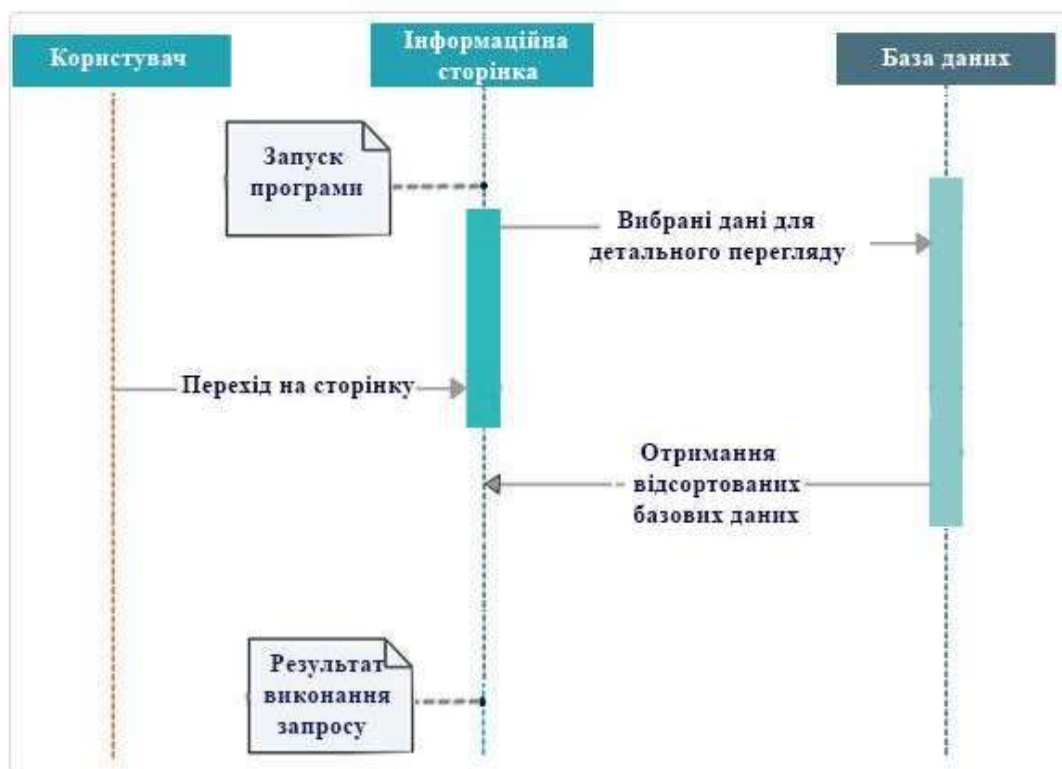


Рисунок 2.4 – Діаграма послідовності перегляду інформації

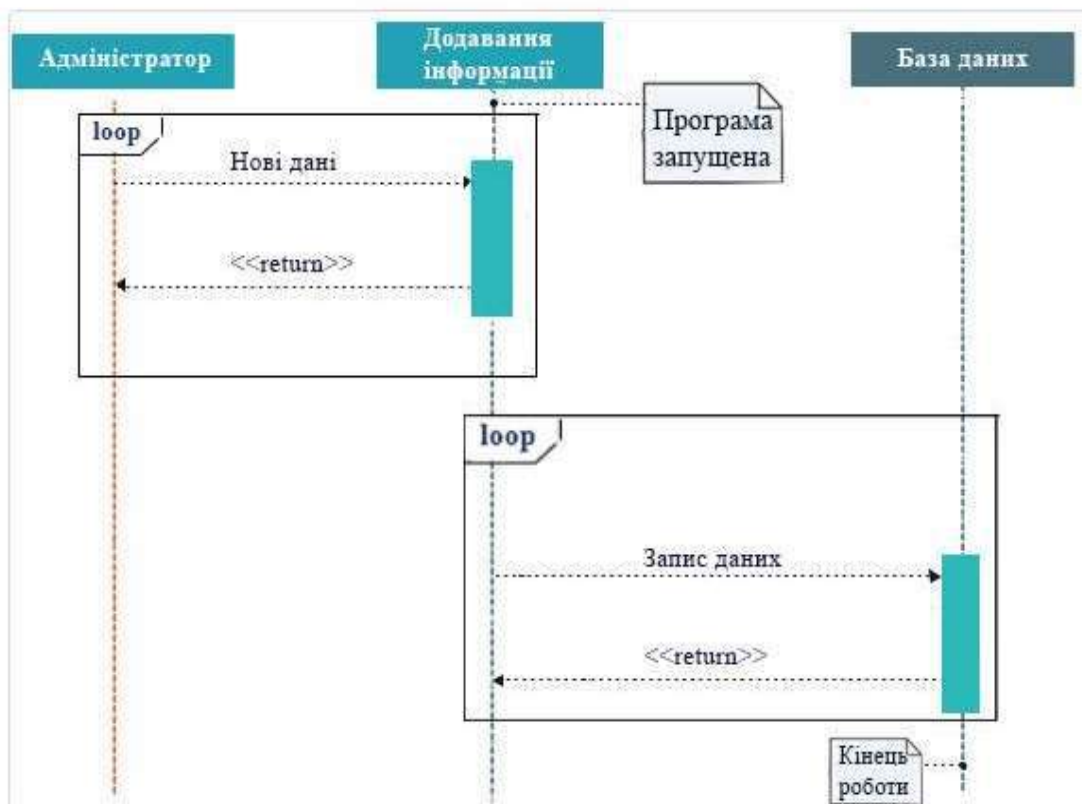


Рисунок 2.5 – Діаграма послідовності додавання інформації

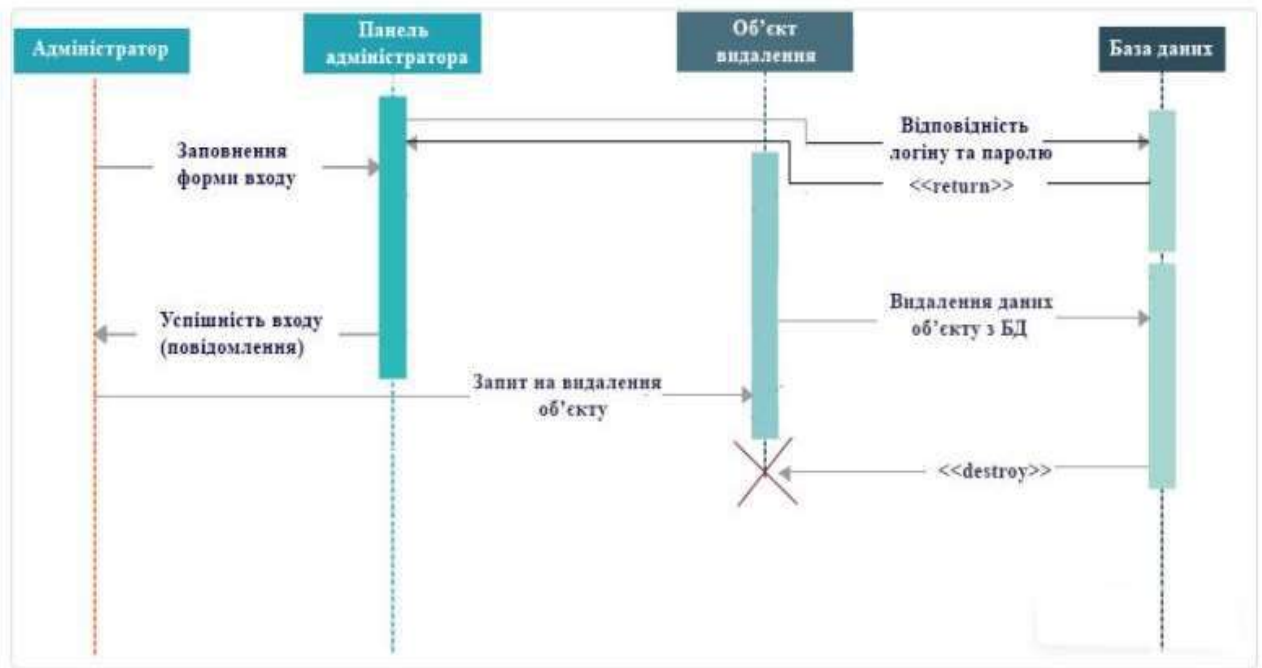


Рисунок 2.6 – Діаграма послідовності видалення інформації

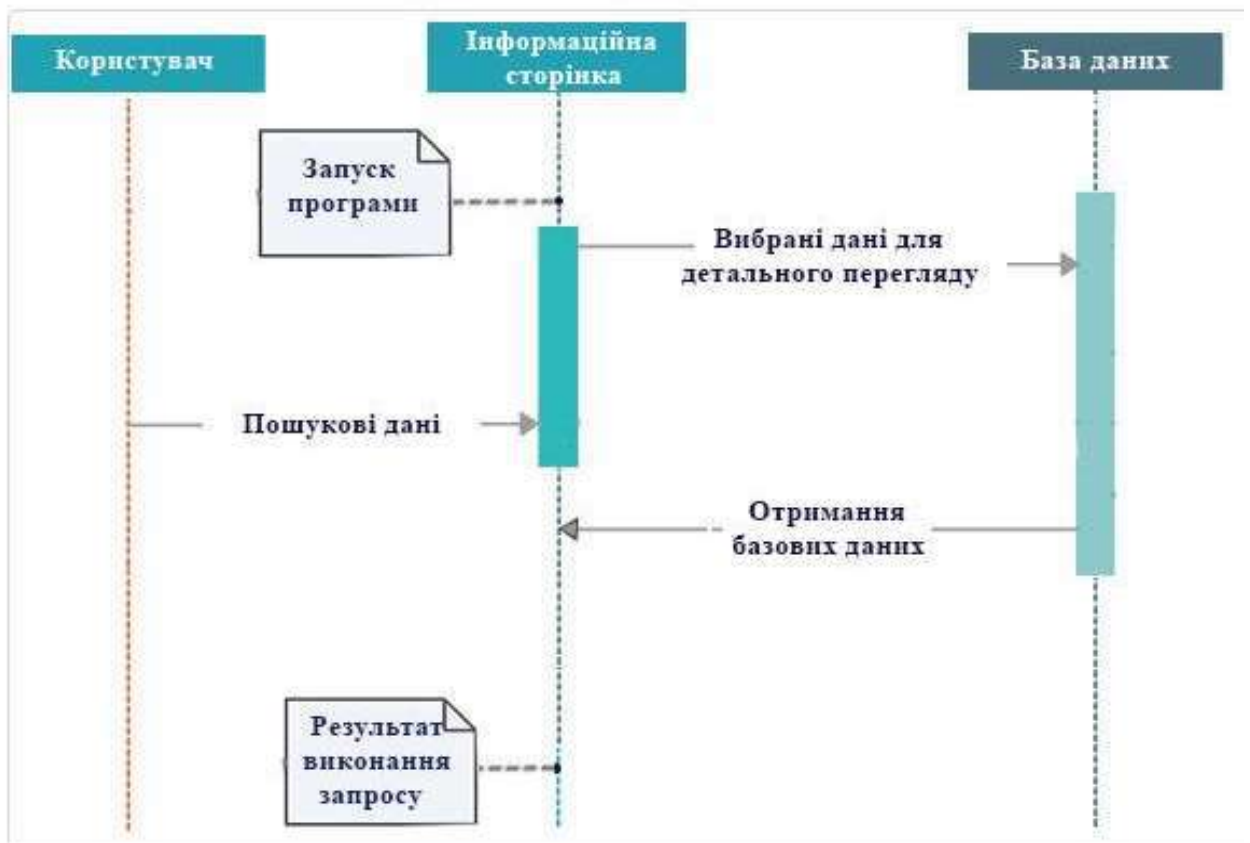


Рисунок 2.7 – Діаграма послідовності пошуку інформації

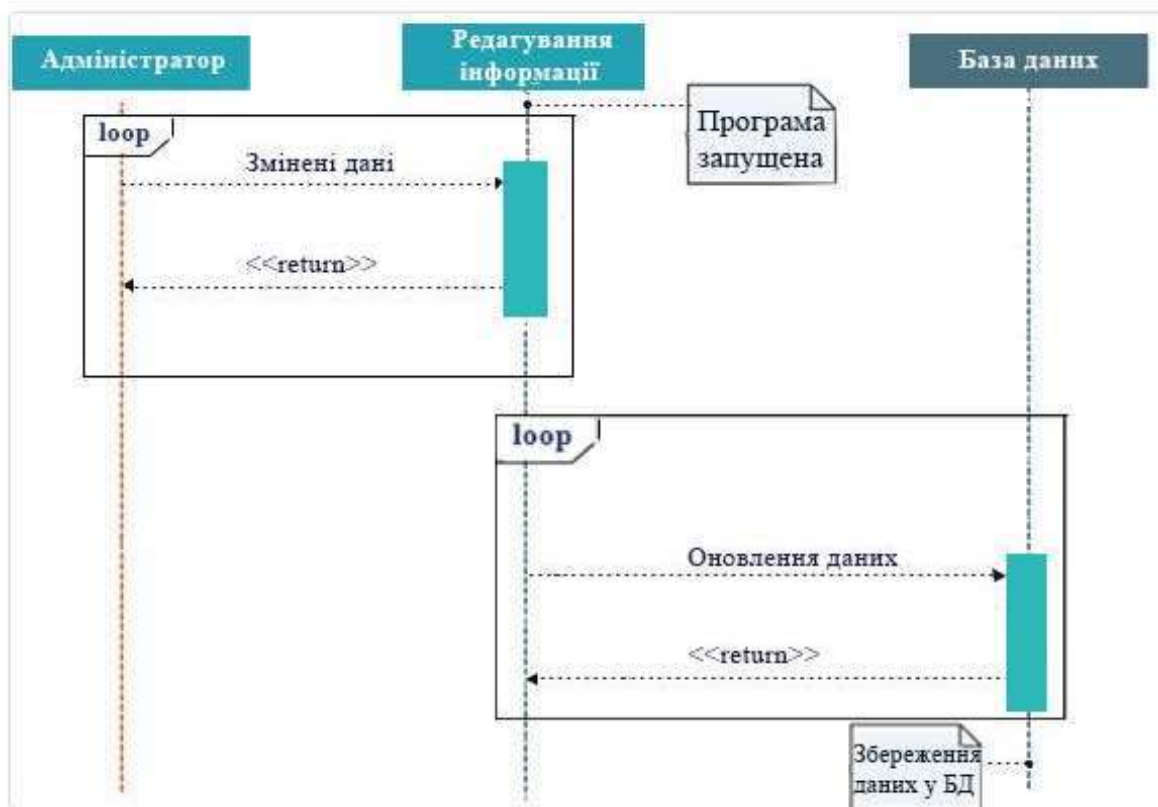


Рисунок 2.8 – Діаграма послідовності редагування інформації

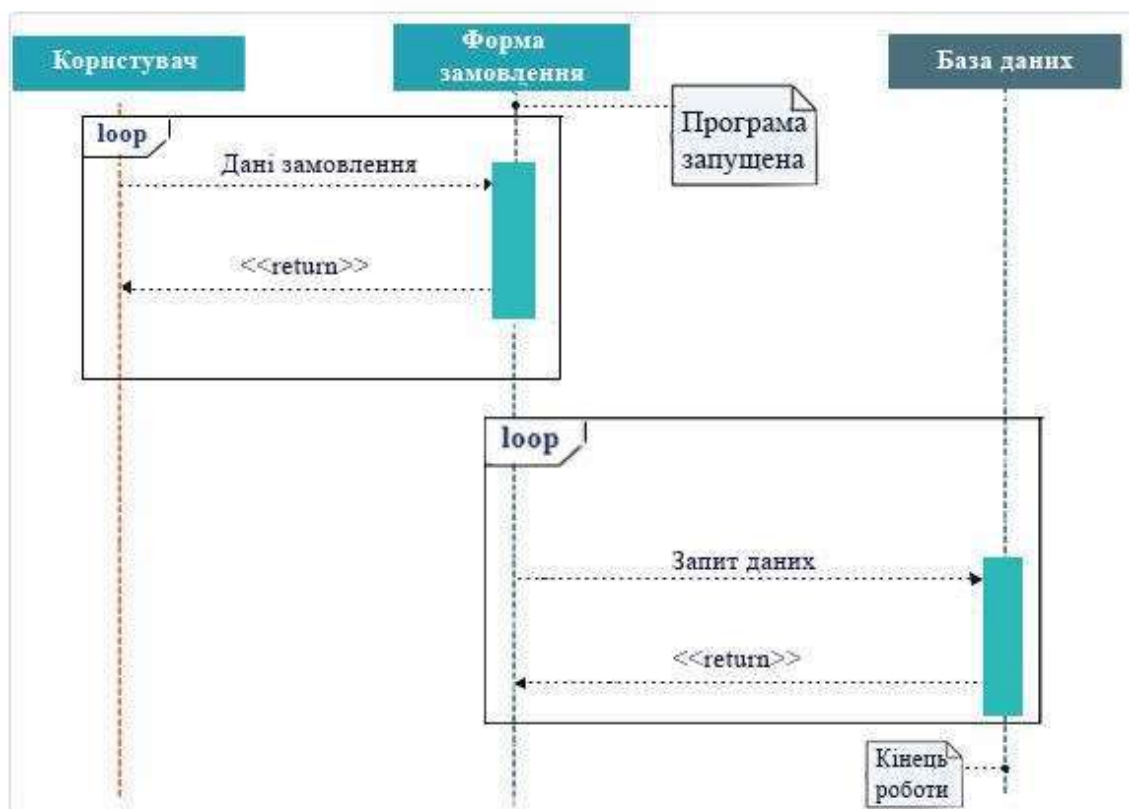


Рисунок 2.9 – Діаграма послідовності оформлення замовлення

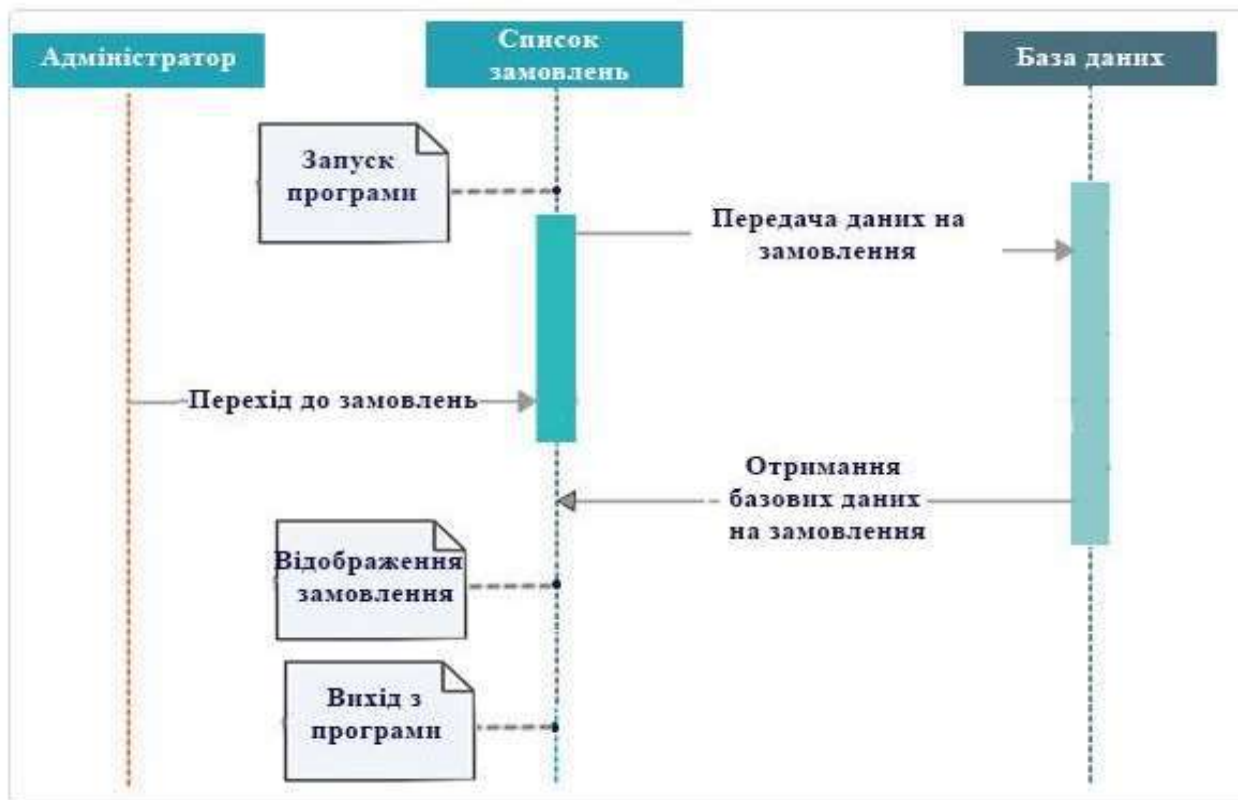


Рисунок 2.10 – Діаграма послідовності перегляду замовлень

### 2.3 Розгорнуті описи прецедентів

Use Case (варіант використання, ВІ, Прецедент, Юзкейс) – це сценарна техніка опису взаємодії. За допомогою Use Case може бути описана і вимога користувача, і вимога до взаємодії систем, і опис взаємодії людей і компаній в реальному житті.

У загальному випадку, за допомогою Use Case може описуватися взаємодія двох або більше учасників, що має конкретну мету:

- купівля товару в магазині (Покупець-Продавець);
- надсилання листа електронною поштою (Адресант-Поштовий клієнт);
- запит сторінки браузером (браузер-вебсервер).

У розробці програмного забезпечення цю техніку часто застосовують для проектування та опису взаємодії користувача та системи, тому назва Use Case часто

сприймає як синонім вимоги людини-користувача до вирішення певного завдання у системі.

Для розгорнутого опису прецедентів є різні шаблони форматування. Однак найчастіше використовується шаблон, наведений нижче. Цей приклад детального опису прецеденту показує безліч типових елементів та питань.

Розглянемо прецедент П1 «Оформлення покупки». Основним виконавцем є користувач інтернет-магазину. Розглянемо нижче зацікавлених осіб та їх вимоги.

«Користувач інтернет-магазину» бажає купити товари та швидко оформити покупку з мінімальними зусиллями, а також отримати підтвердження факту придбання для можливого повернення товару.

«Адміністратор» відноситься до користувачів системи та хоче отримати свої комісійні від продажу.

«Компанія» (інтернет-магазин, база даних) повинна акуратно записати транзакцію та задовольнити інтереси покупця. Вона повинна переконатись, що служба авторизації платежів зафіксувала всі дані про платіж. Компанія зацікавлена у забезпеченні стійкості до збоїв; повинна продовжувати реєструвати продажі, навіть якщо серверні компоненти (наприклад, служба віддаленої перевірки кредитоспроможності) недоступні. Повинна автоматично оновлювати бухгалтерську документацію та вести складський облік.

«Державні податкові служби» бажають отримувати податок від кожного продажу. Таких служб може бути кілька, у тому числі національна та місцева.

Далі опишемо основний успішний сценарій або основний процес.

Спочатку користувач заходить на вебсайт інтернет-магазину та обирає собі товар. Він може зареєструватися та авторизуватися. Далі переглядає замовлення та оформлює його.

Адміністратор у разі прямого звернення до нього по телефону проводить у системі оформлення заказу замість покупця.

Система записує найменування товару та видає його ідентифікаційний номер, опис, ціну та загальну вартість. Вона обчислює загальну вартість покупки з

податком, повідомляє покупцю загальну вартість та пропонує сплатити покупку зручним для нього способом.

Далі адміністратор пересилає товар, у разі післяплати одержує гроші у поштовому відділенні.

Користувач оплачує покупку.

Система обробляє платіж, реєструє продаж та надсилає інформацію про неї до зовнішньої бухгалтерської системи (для оновлення бухгалтерських документів та нарахування комісійних) та системі складського обліку (для оновлення даних).

Адміністратор контролює рух товару, а також наявність відповідних товарних карток на сторінці інтернет-магазину.

Після покупки користувач залишає відгуки та коментарі.

Адміністратор їх переглядає та приймає подальші рішення.

Розглянемо розширення (альтернативні потоки) – відхилення від основного сценарію.

При кожному виході системи з ладу для введення системи в дію та коректної обробки платежу необхідно забезпечити відновлення всіх транзакцій та подій з будь-якого кроку сценарію.

Для цього спочатку адміністратор перезапускає систему.

Після цього система відновлює попередній стан. Вона робить це у декілька кроків. Перший з них - визначає аномалію, що спричинила збій. Далі система повідомляє адміністратора про помилку, реєструє помилку та переходить у початковий стан. Після цього вона починає новий продаж.

Іноді система не підтверджує авторизацію. Найчастіше причиною цього є те, що користувач ввів невірний логін або пароль. Але іноді в результаті несанкціонованого доступу до бази даних можуть бути змінені дані користувача. Для налагодження потрібно зв'язатися з адміністратором та виправити помилку.

Користувачу може бути потрібно зробити зміни в замовленні, що є ще одним відхиленням від основного сценарію.

Перед оплатою користувач перевіряє якість та комплектність товару, і при їхньому порушенні товар відправляється назад до інтернет-магазину. Адміністратор

приймає повернення, зв'язується з покупцем, виправляє недоліки і повторно відправляє товар.

При оновленні різновидів товару, цін, кодів, адміністратор ставить відмітку про їх наявність у картках. Він повідомляє керівництво про відсутність чи наявність різних видів товару.

Перечислимо спеціальні вимоги до магазину:

- можливість оформлення замовлення користувачем у різний спосіб: через наповнення кошика, за допомогою особистого кабінету (при його наявності), по телефону;

- простота порядку оформлення;

- різноманітні способи доставки заказів для зручності покупців;

- можливість локалізації (подання різними мовами) тексту, що відображається;

- можливість додавання нових бізнес-правил у процесі функціонування системи;

- обов'язкове повернення товару згідно правил, регульованих існуючим законодавством;

- програми лояльності(знижки на товар, заміна на дешевші аналоги та інше).

Перечислимо технології, які будуть використовуватися при розробці проєкту. Для створення інтерфейсу користувача використовується фреймворки Vue та Bootstrap. Система використовує базу даних MySQL, типи даних – VARCHAR. Для створення адмін-панелі використовується Laravel та AdminLTE.

Відкриті питання, які будуть виникати при розробці та експлуатації магазину, такі:

- вивчення законодавства з оподаткування;

- відновлення бізнес-правил у процесі функціонування системи;

- зміна налаштувань при коливанні вартості пересилання відділками зв'язку;

- питання конкуренції та програм лояльності.

## 3 ПРОЄКТУВАННЯ ВЕБЗАСТОСУНКУ

### 3.1 Проєктування архітектури програмного забезпечення

#### 3.1.1 Вибір архітектурних рішень для проєкту

Для розробки панелі адміністрації та API для клієнтської частини додатку буду використовувати фреймворк, який використовує патерн проєктування MVC.

MVC – це такий спосіб організації коду, який розділяє код застосунку на три частини:

- Model (модель) – частина, яка відповідає за усі дані у застосунку. У нашому випадку дані – це список товарів та їх характеристик;
- View (представлення) – це частина, яка відповідає за відображення даних та інших елементів користувацького інтерфейсу;
- Controller (контролер) – частина, яка відповідає за взаємодію моделі та користувача.

Взаємодіють ці три частини так, як показано на рисунку 3.1.



Рисунок 3.1 – Взаємодія між частинами MVC

Головна особливість MVC – відділення бізнес-логіки від представлення, тобто можливість незалежного внесення змін у кожну з трьох вищезазначених частин. Це забезпечує кращу організацію коду, що допомагає спростити процес розробки та покращити супроводжуваність додатку. Інші переваги цього архітектурного рішення такі:

- розділення відповідальності: кожний компонент додатку має лише одну роль, що спрощує розуміння коду та його підтримку;
- повторне використання коду: кожний компонент додатку може використовуватися повторно у різних частинах додатку;
- легкість тестування: кожен компонент може бути протестований окремо, що спрощує сам процес тестування та знаходження помилок.

Для розробки інтерфейсу користувацької частини сайту використаю фреймворк, який втілює MVVM. MVVM – шаблон проектування архітектури, концепція якого – безпосередня прив’язка даних до представлення. Він ділиться на три частини:

- Model (модель) – частина, яка описує дані та способи роботи з ними;
- View (представлення) – частина, яка відображує дані з моделі та приймає дії користувача;
- ViewModel (модель представлення) – частина, яка пов’язує модель та представлення: підготовляє дані з моделі для представлення та обробляє дії користувача.

Взаємодіють модель, представлення та модель представлення так, як показано на рисунку 3.2.

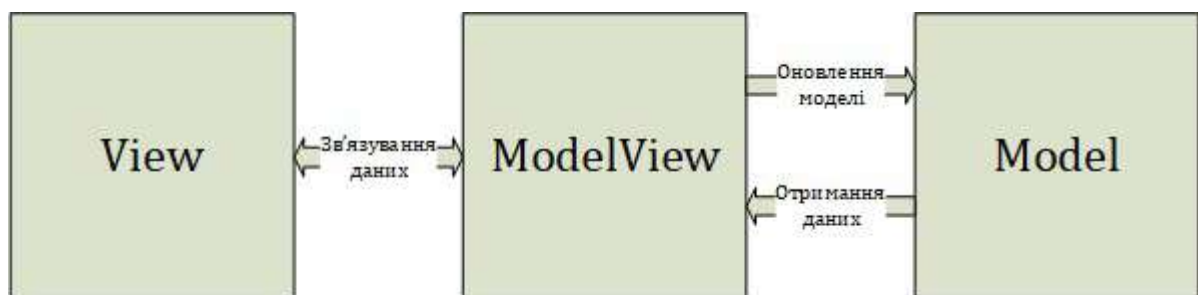


Рисунок 3.2 – Взаємодія між частинами MVVM

Основні переваги MVVM такі:

- розділення відповідальності: забезпечується розділення бізнес-логіки та користувацького інтерфейсу;
- спрощення тестування: ViewModel можна протестувати незалежно від представлення;
- двосторонній зв'язок (data binding): MVVM забезпечує двостороннє зв'язування даних між представленням та моделлю, що дозволяє автоматично оновлювати дані в обидві сторони при зміні даних на одній з них.

Для розробки оптимальними варіантами будуть Laravel та Vue, бо на них можна розробляти проєкти різної складності з мінімальними зусиллями. До того ж, вони мають цілий ряд інших переваг.

Тепер розглянемо більш детально, як MVC реалізується в Laravel. Почну з того, що файли контролерів, моделей та представлень зберігаються у різних папках. Тепер для повного розуміння опишу процес обробки запиту на зміну запису БД. Спочатку з форми надходить запит. Потім відбувається перенапрявлення у необхідний метод необхідного контролеру у файлі з маршрутами додатку, тобто знаходиться маршрут, URI якого відповідає URI запиту. Маршрути виглядають так, як у лістингу 3.1. У методі контролеру відбувається робота з базою даних шляхом викликання функцій класу моделі, який представляє таблицю БД. Тобто видно, що запит спочатку напрямляється у контролер, який працює з моделлю. У Laravel це клас, який представляє таблицю бази даних. А далі модель оновлюється та дані з неї відображуються у представленні.

### Лістинг 3.1 – Зразок маршруту Laravel

```
Route::get(
    '/create', \App\Http\Controllers\Category\CreateController::class
)->name('category.create');
```

Перейдемо до пояснення реалізації MVVM у Vue.js. Інтерфейс у Vue складається з компонентів. Компонент – це деяка частина користувацького інтерфейсу, наприклад, форма або список. У компоненті містяться розмітка, стилі

оформлення та дані. Стили оформлення та дані – необов’язкові частини, але у більшості компонентів дані, власні чи взяті з батьківського, все ж таки є. Дані у Vue – об’єкт. Найпростіший спосіб позначити місце відображення даних у представленні – синтаксис `{{ ключ_елементу_об’єкту_з_даними }}`. При цьому відбувається зв’язування виразу у фігурних дужках з властивістю об’єкту з даними, і при зміні цього фрагменту даних він перевідображується у представленні. Але найяскравіше двостороннє зв’язування даних проявляється у компоненті з полем форми. Коротко кажучи, поле форми у Vue прив’язується до деякої властивості об’єкту з даними, і при зміні значення поля форми змінюється і значення прив’язаного фрагменту даних, а при зміні властивості об’єкту з даними, прив’язаної до цього поля, змінюється і відображуваний в полі ввід. Так і реалізовується основна ідея MVVM у Vue.

### 3.1.2 Побудова діаграми розміщення

Діаграма розміщення вебдодатку магазину одягу буде розташована на рисунку 3.3.

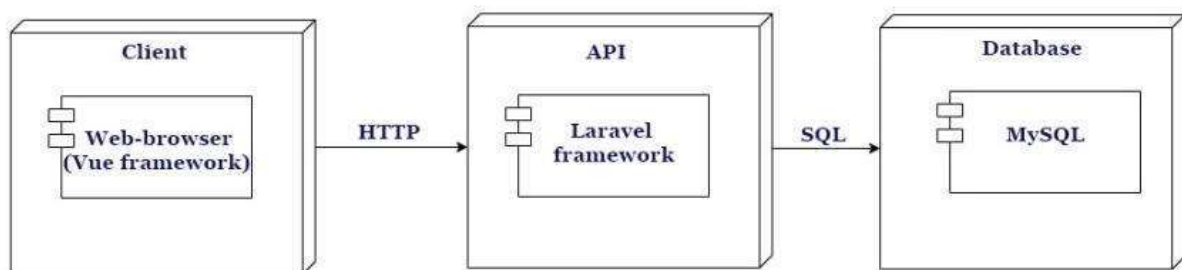


Рисунок 3.3 – Діаграма розміщення додатку онлайн-магазин

## 3.2 Проєктування інтерфейсу користувача

### 3.2.1 Опис змісту сторінок вебдодатку

Додаток онлайн-магазину буде складатися з частини користувача та панелі адміністратора.

Адміністративна частина буде надавати можливість керування товарами та їх властивостями, а саме: створення, змінення та видалення.

Користувацька сторона буде забезпечувати усе необхідне для того, щоб клієнту магазину можна було комфортно здійснити процес купівлі.

У додатку магазину на стороні користувача будуть наступні сторінки:

а) головна сторінка, на якій будуть знаходитися:

- 1) список категорій товарів;
- 2) банер (наприклад, щоб відобразити, що завезли нову колекцію);
- 3) список найпопулярніших товарів;

б) каталог товарів з можливістю пошуку та сортування за різними характеристиками, такими як ціна, назва, категорія, теги, кольори;

в) сторінка з інформацією про магазин;

г) сторінка з інформацією про політику конфіденційності;

д) сторінка з інформацією про політику повернення;

е) сторінка з найчастіше задаваними питаннями;

ж) сторінка входу/реєстрації;

з) кошик.

У адміністративній частині будуть:

– сторінки відображення записів кожної з таблиць бази даних, тобто сторінки відображення усіх наявних товарів, кольорів, тегів тощо;

– сторінка створення запису таблиці, на якій можна додати товар, категорію, колір тощо;

– сторінка перегляду запису таблиці, на якій можна переглянути усі властивості одного товару, кольору, тегу тощо.

– сторінки редагування одного запису таблиці, тобто сторінки редагування товару, тегу, категорії тощо.

Обговорю деякі подробиці щодо сторінок на панелі адміністрування. Маю сумніви у потрібності сторінок створення користувачів та замовлень, тому створювати їх буду тільки якщо у процесі експлуатації додатку магазину виникне така потреба.

### 3.2.2 Ескізи сторінок

Для того, щоб мати більш ясне уявлення про те, як повинен виглядати інтерфейс користувача та що з чого він повинен складатися, я зробила макети. Вони показані на рисунках 3.4-3.6.

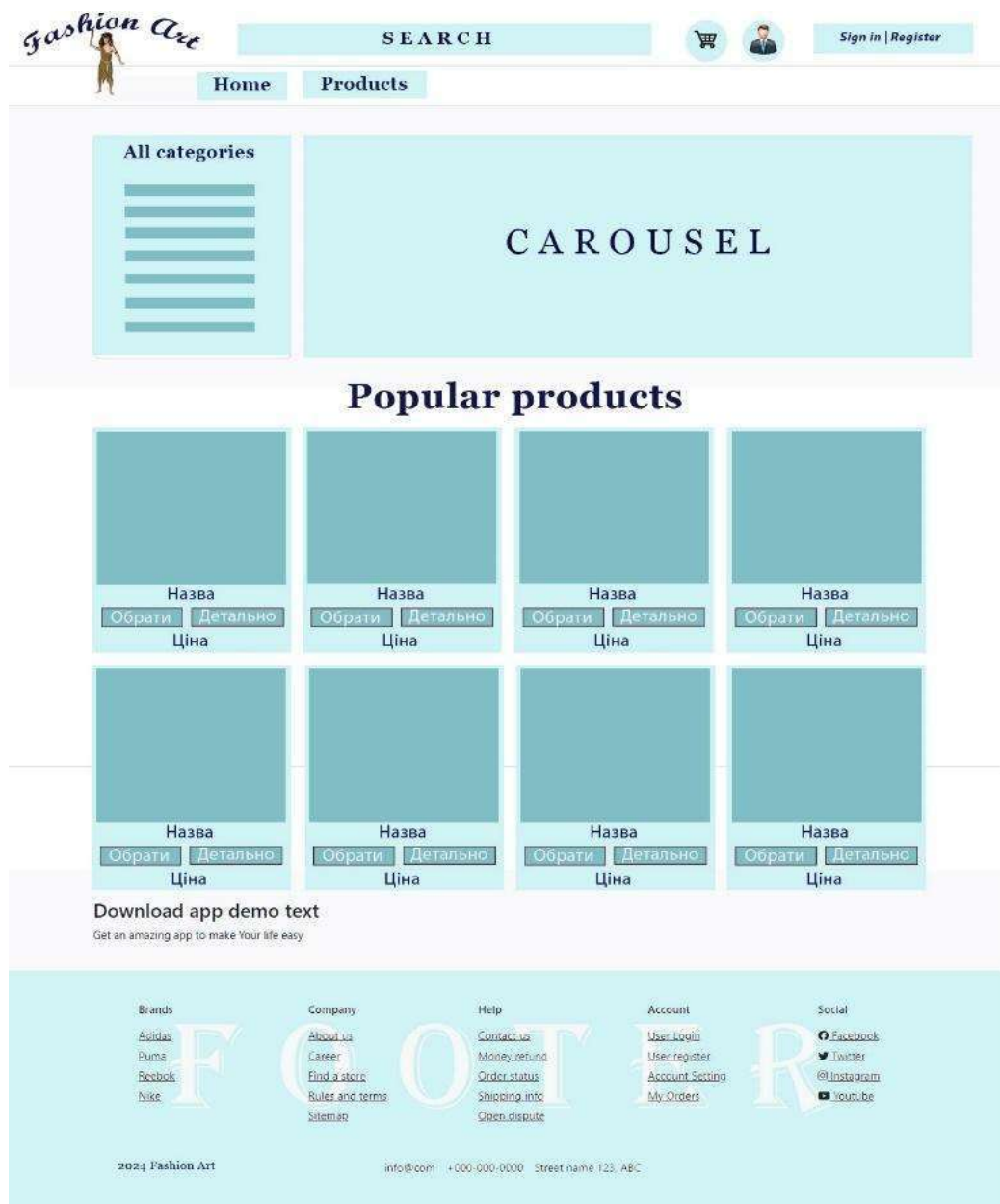


Рисунок 3.4 – Ескіз головної сторінки

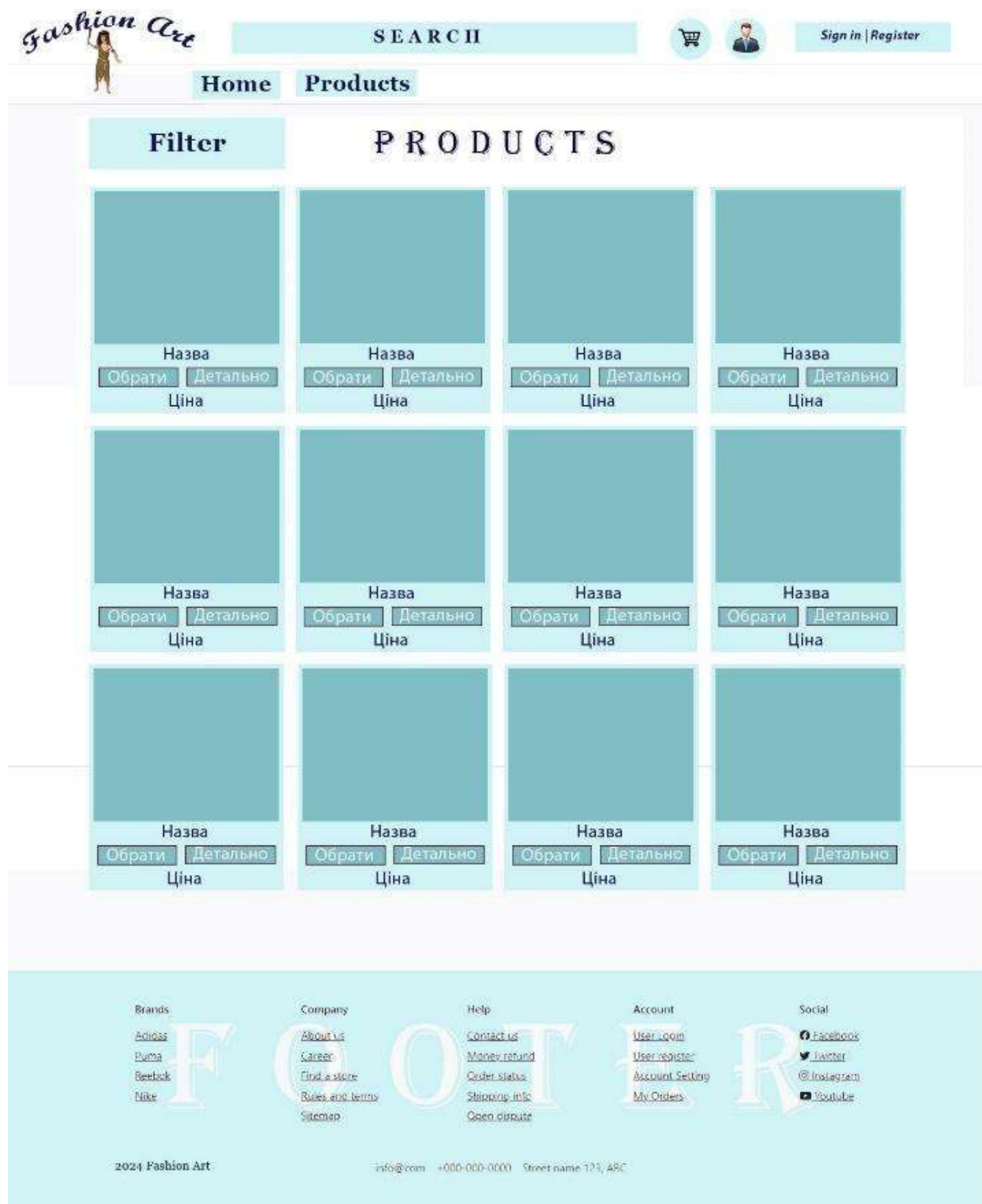


Рисунок 3.5 – Ескіз сторінки каталогу продуктів

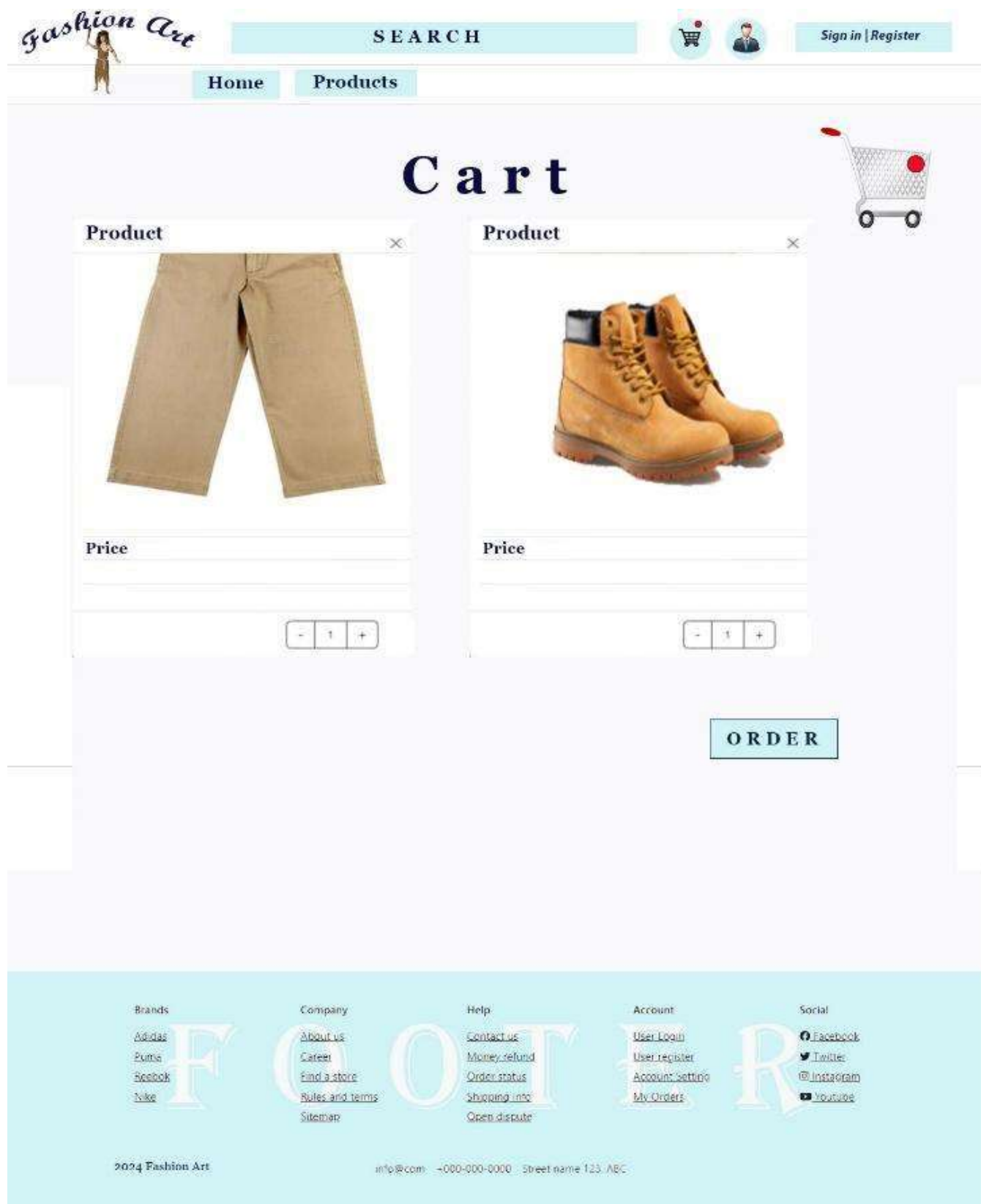


Рисунок 3.6 – Ескіз сторінки з кошиком

## 4 РЕАЛІЗАЦІЯ ІНТЕРНЕТ-МАГАЗИНУ

### 4.1 Дизайн сайту

Як відомо, створення кожного сайту починається з дизайну його сторінок. Раніше для цього використовувались графічні редактори Adobe: Photoshop, Illustrator, After Effect тощо. В даний час для вебдизайну існують спеціальні інструменти, серед яких знаходиться і онлайн-редактор Figma. Перевага цього інструменту в тому, що в одному проєкті одночасно можуть працювати кілька дизайнерів, замовник на відстані (через інтернет) може побачити процес створення та щось відкоригувати, висловивши свої побажання. Крім цього верстальнику легше копіювати у файли розробки проєкту властивості об'єктів, розміщених у фрейми. Це можна бачити на скріншоті на рисунку 4.1, де виділено фрейм з відображенням його характеристики (праворуч), готової до експорту.

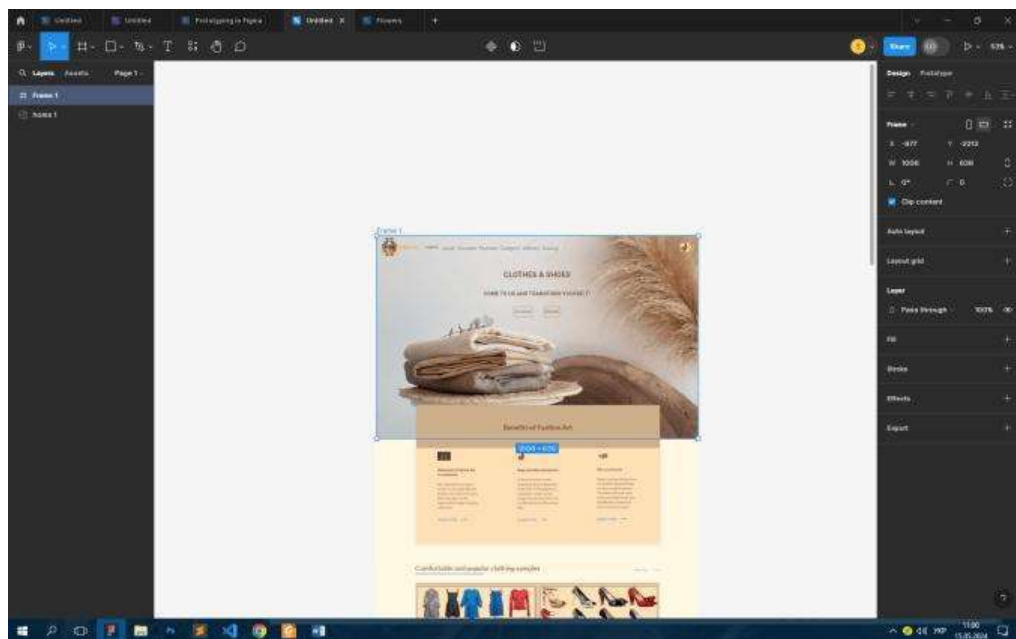


Рисунок 4.1 – Робоча область редактора Figma з макетом сайту

Під час створення дизайну потрібно урахувувати головне: у користувача повинно бути бажання залишатися на сторінці сайту. Для цього має виникнути візуальний контакт із зображеннями, забарвленням, текстовими шрифтами. Велике

значення має зручність розташування об'єктів користувача (кнопок), мінімалізм і ненав'язливість. Інакше кажучи, все має бути гармонійним. На рисунку 4.2 знаходиться дизайн сторінки сайту інтернет-магазину «Fashion Art», що пропонує взуття.



Рисунок 4.2 – Прототип сторінки shoes.html із взуттям

Крім головної сторінки, зображення якої знаходиться на плакаті 4 графічної частини, створюються візуальні прототипи інших необхідних сторінок проекту: кошику (cart.php), доставки (delivery.php), контактів (contacts.php), продукції (product.php) тощо. Їхніми незмінними елементами можуть бути заголовок

(«шапка») та нижня частина («футер»). Але в даному сайті верхня частина головної сторінки відрізняється від «шапки» інших сторінок. На рис 4.3 видно дизайн притемненої шапки головної сторінки з боковою панеллю модального вікна зі списком товарів у кошику.

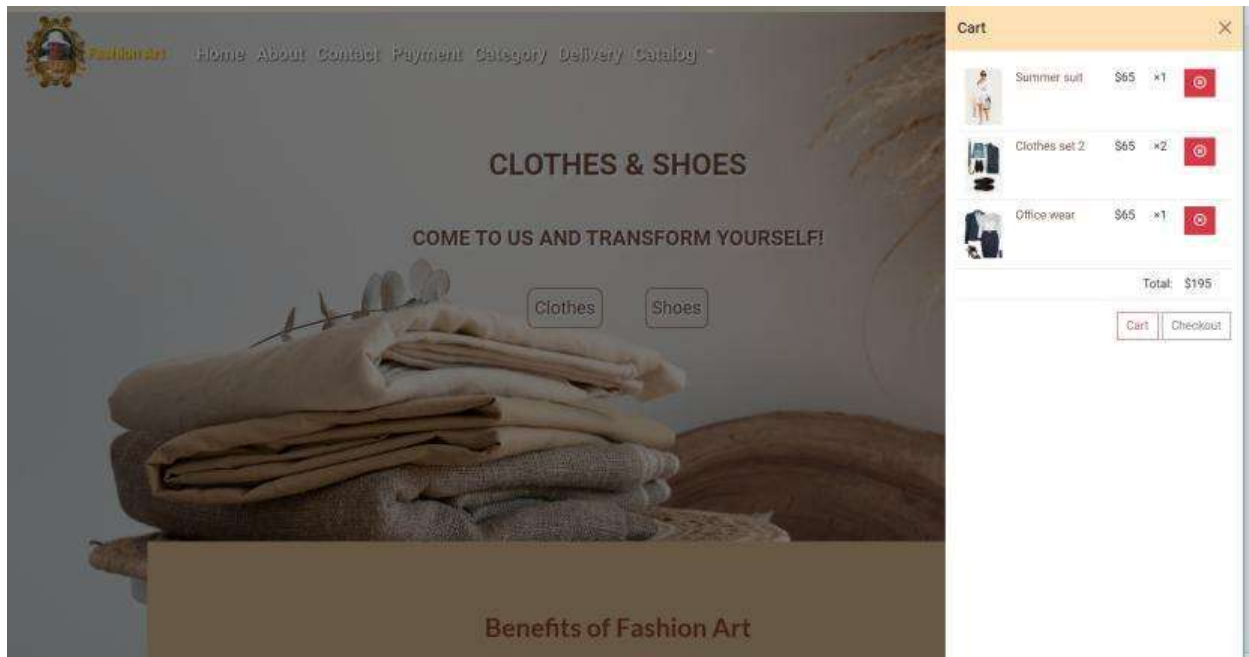


Рисунок 4.3 – Шапка сторінки index.html з боковою панеллю з кошиком

## 4.2 Верстання сайту

Для перетворення макета у статичний сайт буде використано фреймворк Bootstrap 5.3. Цей вибір обумовлений можливістю адаптації єдиного прототипу проєкту «Fashion Art» під роздільну здатність як великих моніторів, так і мобільних пристроїв. В іншому випадку дизайнеру потрібно робити додаткові міні-макети для їхнього подальшого відтворення у планшетах та смартфонах.

Зазвичай є питання, яку файлову структуру слід створити. Тут є два різні підходи. У першому випадку кожному розділу відповідає HTML-документ, що знаходиться в корені сайту. Альтернативний варіант – створення папок, які відповідатимуть певним розділам. І тоді у кожній папці розміщено файл index.html.

Оскільки в нашому випадку існує одна головна сторінка, то зупиняємось на першому варіанті. На будь-якому сайті не обійтися без додаткових файлів, які зазвичай розміщуються в корені сайту. Сюди входять файли, призначені до виконання певних завдань і мають обов'язкове ім'я, а також файли, ім'я яких визначається розробником. Серед обов'язкових імен файлів: `index.html`, `.htaccess`, `robots.txt`, `favicon.ico`.

`index.html` — назва головної сторінки, яка має відкриватися при вказівці на адресу. Це ім'я може змінюватися залежно від типу вебсервера та його налаштувань. Але зазвичай воно саме таке.

`.htaccess` — конфігураційний файл вебсервера Apache. Зазначений сервер є найпопулярнішим і найпоширенішим у світі, тому і цей файл можна зустріти повсюдно. У нашому випадку він також використовується.

`robots.txt` — файл, призначений для пошукових систем. При індексуванні сайту насамперед розшукується він. За допомогою файлу `robots.txt` ми можемо вказувати, які файли на нашому сайті буде видно пошуковим роботам. З його допомогою можна обмежити кількість запитів на сканування і цим знизити навантаження на сайт. Приклад заборони для всіх пошукових систем відвідувати папки сайту `fonts` та `img` вказаний у лістингу 4.1.

#### Лістинг 4.1 — Декілька правил `.htaccess`

```
User-agent: *
Disallow: /fonts/
Disallow: /img/
```

`favicon.ico` - іконка сайту, яка з'являється поряд з назвою вкладки у браузері. Це малюнок розміром 16 на 16 пікселів формату ICO.

Файлова структура сайту «Fashion Art» з кореневою папкою, яка носить таку ж назву, розміщено на першому плакаті графічної частини. При цьому потрібно враховувати, що при переносі файлів у проєкт Vue їхнє розширення зміниться з `.html` на `.vue`. Звісно, при цьому зміниться кількість файлів: як мінімум, будуть додані «шапка» (`Header.vue`) та «футер» (`Footer.vue`).

До початку кодування створюється папка «Fashion Art». Її тимчасове розташування може бути в будь-якому місці комп'ютера. До переносу на сервер в ній будуть знаходитися HTML-файли сторінок, тека assets з папками «css» (файли стилів), «img» (зображення), «font» (шрифти), «js» (файли JavaScript).

Для розмітки та застосування стилів використано Visual Studio Code (VS Code). Це кросплатформений редактор коду від компанії Microsoft, розроблений на базі фреймворку Electron. З його допомогою можна розробляти кросплатформені десктопні програми, використовуючи вебтехнології.

Верстку починають з головної сторінки (index.html), попередньо розібравшись у використовуваних елементах. Як видно на скриншоті головної сторінки, зображення якої є у графічній частині, у верхній частині шапки знаходиться навігаційна панель, а для перегляду та реклами нових товарів працюватиме «карусель». Використання цього елемента обумовлено великою кількістю контенту, що переглядається. Нижче розташовані картки з рекомендованим магазином одягом, його ціною та переходом до кошика. Також на макеті вставлена Google-карта з вказаним місцезнаходженням магазину. Інші елементи (текстовий контент, зображення) прикріплені статично до відповідних секцій. В результаті отримуємо те, що зображено на плакаті 4.

На початку крім файлу index.html у редакторі створюється файл style.css (у теці «css»), в якому виконується налаштування шрифтів, кольорів, відступів (зовнішніх та внутрішніх) та інших стилів. Фрагмент файлу style.css зі стилями елементу навігаційної панелі (navbar) показано нижче.

#### Лістинг 4.2 – Фрагмент style.css для задання вигляду навігаційної панелі

```
/*=====Start Navbar-----*/

.navbar-nav {
    margin-left: 40px;
}
.navbar-brand {
    margin-right: 80px;
}
.nav-link {
    color: white;
    font-weight: 500;
    text-shadow: 2px 2px 3px black ;
```

```

    font-size: 1.5em;
    &:hover{
        background-color:white;
        color: var(--text-color);
        text-shadow: none;
    }
}
.nav-link.active, .nav-link:active {
    color: var(--title-color) !important;
    text-shadow: 2px 2px 3px white !important;
}
.nav-link:focus {
    color: var(--accent-color) ;
    text-shadow: 1px 1px 2px white;
}
.headernav-scroll{ /*клас скроля навбара*/
    background-color: #E6C8A3;
    color: var(--text-color);
}
.cart-shop {
    float: right;
    text-align: center;
}
.img-cart {
    background-color: #FFE0B2;
    margin: 40px 0;
}
#img-title {
    margin-top: 280px;
}

/*=====End Navbar-----*/

```

Разом з кодом HTML, який видно внизу, буде створено навігаційну панель (navbar) сторінки сайту.

### Лістинг 4.3 – Фрагмент index.html для описання навігаційної панелі

```

<nav class="navbar navbar-expand-lg navbar-scroll" >
    <div class="container">
        <a class="navbar-brand" href="#" data-bs-toggle="modal" data-bs-
            target="#modalPush"><strong>style="color:#EAF00EFF; text-shadow:2px
            2px 3px #EC5912FF; ">Fashion Art</strong>
        </a>
        <button class="navbar-toggler" type="button" data-bs-toggle="offcanvas"
            data-bs-target="#offcanvasNavbar" aria-controls="offcanvasNavbar"
            aria-label="Toggle navigation"> <span class="navbar-toggler-
            icon"></span></button>
            <div class="offcanvas offcanvas-start" tabindex="-1"
                id="offcanvasNavbar" aria-labelledby="offcanvasNavbarLabel" >
                <div class="offcanvas-header">
                    <h5 class="offcanvas-title"
id="offcanvasNavbarLabel">Catalog</h5>
                    <button type="button" class="btn-close" data-bs-dismiss="offcanvas"
                        aria-label="Закреть"></button>
                </div>
                <div class="offcanvas-body" >
                    <ul class="navbar-nav " >

```

```

<li class="nav-item">
  <a class="nav-link" aria-current="page"
    href="index.html">Home</a>
</li>
<li class="nav-item">
  <a class="nav-link" href="about.html">About</a>
</li>
<li class="nav-item">
  <a class="nav-link" href="contacts.html">Contact</a>
</li>
<li class="nav-item">
  <a class="nav-link" href="payment.html">Payment</a>
</li>
<li class="nav-item">
  <a class="nav-link" href="category.html">Category</a>
</li>
<li class="nav-item">
  <a class="nav-link" href="delivery.html">Delivery</a>
</li>
<li class="nav-item dropdown">
  <a class="nav-link dropdown-toggle" href="#" role="button" data-
    bs-toggle="dropdown" aria-expanded="false" data-bs-auto-
    close="outside"> Catalog</a>
  <ul class="dropdown-menu" >
    <li><a class="dropdown-item"
      href="sneakers.html">Sneakers</a> </li>
    <li><a class="dropdown-item"
      href="jeans.html">Jeans</a> </li>
    <li><hr class="dropdown-divider"></li>
    <li class="nav-item-dropdown dropend">
      <a class="nav-link dropdown-toggle" href="#"
        role="button" data-bs-toggle="dropdown" aria-
        expanded="false">Sportswear</a>
      <ul class="dropdown-menu dropdown-menu-end" >
        <li>
          <a class="dropdown-item"
            href="men's.html">Men's Sportswear</a>
        </li>
        <li>
          <a class="dropdown-item"
            href="women's.html">Women's Sportswear</a>
        </li>
        <li>
          <a class="dropdown-item"
            href="baby's.html">Beby's Sportswear</a>
        </li>
      </ul>
    </li>
  </ul>
</li>
</ul>
</div>
</div>
</nav>

```

Таким чином з'явилося візуальне відтворення коду, яке можна побачити у вікні браузера (показано на рисунку 4.4).



Рисунок 4.4 – Навігаційна панель сайту «Fashion Art»

Як видно з дизайну цього проєкту, колірне оформлення тяжіє до коричневих відтінків, де основним кольором є «кава з молоком». З цієї причини головним кольором шрифтів вибрано темно-коричневий, а для темного фону (у панелі) – білий з темною тінню.

У свою чергу, з точки зору типографіки використовуються Google-шрифти сімейства Lato різного розміру.

#### Лістинг 4.4 – Встановлення шрифтів у файлі style.css

```
@font-face {
    font-display: swap; /* Check https://developer.mozilla.org/en-US/docs/Web/CSS/@font-face/font-display for other options. */
    font-family: 'Lato';
    font-style: normal;
    font-weight: 400;
    src: url('../fonts/lato-v24-latin-regular.woff2') format('woff2'); /* Chrome 36+, Opera 23+, Firefox 39+, Safari 12+, iOS 10+ */
}
/* lato-700 - latin */
@font-face {
    font-display: swap; /* Check https://developer.mozilla.org/en-US/docs/Web/CSS/@font-face/font-display for other options. */
    font-family: 'Lato';
    font-style: normal;
    font-weight: 700;
    src: url('../fonts/lato-v24-latin-700.woff2') format('woff2'); /* Chrome 36+, Opera 23+, Firefox 39+, Safari 12+, iOS 10+ */
}
/* lato-900 - latin */
@font-face {
    font-display: swap; /* Check https://developer.mozilla.org/en-US/docs/Web/CSS/@font-face/font-display for other options. */
    font-family: 'Lato';
    font-style: normal;
    font-weight: 900;
    src: url('../fonts/lato-v24-latin-900.woff2') format('woff2'); /* Chrome 36+, Opera 23+, Firefox 39+, Safari 12+, iOS 10+ */
}
```

Заголовний текст шапки, який показано на рисунку 4.5, достатньо характеризує вибір шрифту.



Рисунок 4.5 – Текст верхньої частини сайту

Описувати все кодування неможливо з причини громіздкості цього процесу, але послідовність легко можна розкрити під час верстки. Все починається з головної сторінки (home.html). Для вибору товару користувач клікає по картці, яка зображена на малюнку 4.6.



Рисунок 4.6 – Картка з товаром

Картка функціонує, як посилання. Також у шапці можна перейти до категорій товарів завдяки кнопок (Clothes, Shoes) або у навігаційній панелі натиснути кнопку

«Category». Після цього користувач переходить на сторінку самого товару, яка показана на рисунку 4.7, або вибраної категорії товарів (shoes.html, sneakers.html, clothes.html та інше).



Рисунок 4.7 – Сторінка сайту з вибраним товаром

На даній сторінці існують: характеристики товару; відгуки користувачів; описи доставки, гарантії, оплати; кнопка переходу до корзини. Також внизу знаходиться карусель з картками для можливого вибору додаткового товару. Якщо покупець ознайомився і залишився задоволеним, він переходить до корзини, тискаючи на відповідну кнопку. Щоб подивитись, що вже вибрано, можна клікнути по значку корзинки на верхній панелі, і відкриється бокова панель з товарами та загальною сумою витрат (рисунки 4.3). В цьому вікні теж є кнопка переходу до сторінки корзини (cart.html).

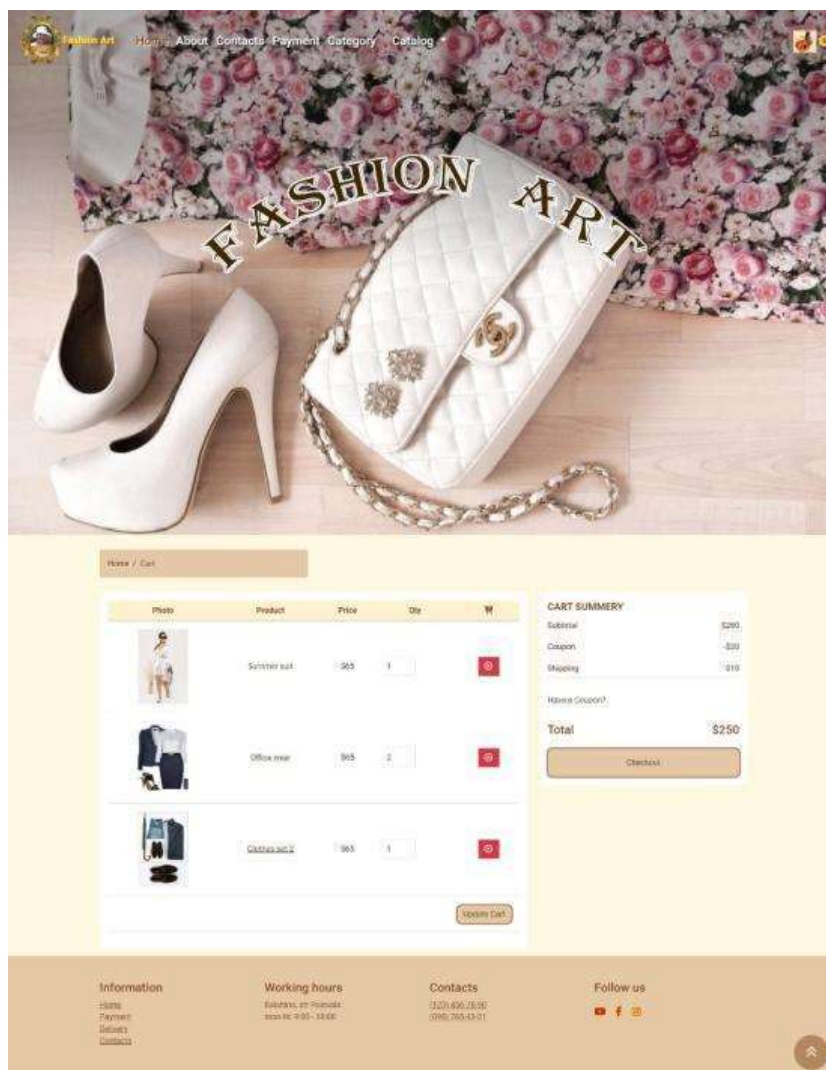


Рисунок 4.8 – Сторінка корзини cart.html з вибраними товарами

На сторінці кошику (рис 4.8) при необхідності можна підвищити, понизити кількість товару або видалити. Після коректування покупець клікає на кнопку «Checkout» для оформлення оплати своєї купівлі.

Оплата проводиться на сторінці checkout.html. Її фрагмент з формою, необхідною для наповнення даними користувача, знаходиться на рисунку 4.9.

Home / Checkout

**BILLING ADDRESS**

Name:  Email:

Phone:  Country:

City:  Address:

Comment:

**ORDER TOTAL**

| Product                  | Total        |
|--------------------------|--------------|
| Summer suit (\$65) x 1   | \$65         |
| Clothes set 2 (\$65) x 2 | \$130        |
| Office wear(\$65) x 1    | \$65         |
| <b>Total</b>             | <b>\$260</b> |

☐ Card payment  
☒ Payment on delivery

Рисунок 4.9 – Форма введення даних на сторінці оплати

На зображенні видно, що оплату можна проводити і карткою, натиснувши радіокнопку «Card payment». В цьому разі вийде інша форма, зображена на рисунку 4.10, яку потрібно наповнити своїми даними.

Welcome to Fashion Art

Тестовий режим

Русский

Fondy

Demo checkout  
https://fondy.eu/  
Demo order

5,00 EUR

Другие способы оплаты

☒ Банковская карта

☐ Интернет-банкнг

☐ Локальные способы оплаты

☐ Most popular

☐ Installments

☐ Электронная почта

Виза card details:  
 NUMBER: 4444 5555 6666 1111  
 NAME: 05/24 \*\*\*  
 CVC1/CVC2: \*\*\*

Рисунок 4.10 - Форма оплати покупки банківською карткою

До цього був розкритий алгоритм вибору та оформлення товару з точки зору клієнтської сторони. Під час верстки однієї HTML-сторінки розробляються кнопки з посиланням на іншу HTML-сторінку, і відбувається перехід у вікні браузера. Наприклад, у шапці є дві кнопки, за допомогою яких можна перейти до сторінок з одягом та взуттям. Їхній код показаний у лістингу 4.4, а вигляд – на рисунку 4.11.

#### Лістинг 4.4 – Код кнопок шапки головної сторінки

```
<div class=" d-flex justify-content-center">
  <a id="intro-btn" type="button" href="clothes.html" class="btn-lg btn-
    outline p-2 hoverable mb-4 ">Clothes</a>
  <a id="intro-btn" href="shoes.html" class="btn-lg btn-outline p-2
    Hoverable mb-4 ">Shoes</a>
</div>
```



Рисунок 4.11 - Кнопки переходу до сторінок одягу та взуття

Після натискання на кнопку «Clothes» відбувається перехід на сторінку одягу, показану на рисунку 4.12.

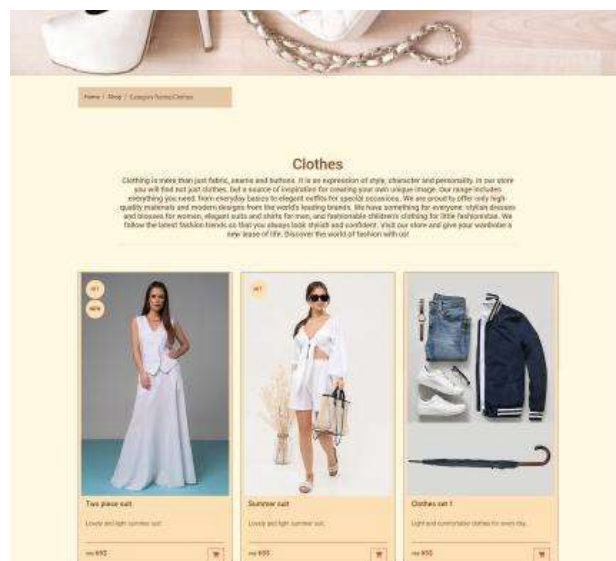


Рисунок 4.12 – Фрагмент сторінки одягу clothes.html у вікні браузера

Перед програмуванням сайту на серверних мовах готову верстку треба впровадити у фреймворк Vue.js. Для цього потрібно розбити її на частини (компоненти) та встановити через менеджер npm усі необхідні пакети (такі як Bootstrap, маршрутизатор Vue та за необхідності інші) за допомогою команди `npm install`.

Спочатку виконуємо команду `npm create vue@latest` та заповнюємо усі дані про проєкт, такі як його ім'я.

У щойно створеному проєкті існує тека `src`. У ній додаємо підпапку `assets` для `css`, малюнків та інших прикріплень до проєкту. Потім створимо каталог для сторінок застосунку `views` та теку для частин інтерфейсу `components`. Також у процесі розробки за потреби можемо додати інші теки. Розбиваємо верстку на частини та розміщуємо у каталоги з компонентами, які будемо збирати у папку зі сторінками. Компоненти, які використовуються на усіх сторінках, такі як верхня та нижня частини, вставимо у головний файл `App.vue`. Це усе робиться для того, щоб програмувати та підтримувати проєкт було простіше і не довелося копатися серед сотень рядків коду.

Після цих маніпуляцій загальна структура проєкту стане такою, як на рисунку «Структура клієнтської частини на Vue» у графічній частині.

Після впровадження верстки починається розробка взаємодії з серверною стороною. Для цього існує бібліотека `Axios`. За допомогою неї можна легко та без великої кількості коду посилати асинхронні запити та обробляти відповіді від серверу. Приклад формування такого запиту показаний у лістингу 4.5.

#### Лістинг 4.5 – Формування запиту діставання товарів за допомогою `Axios`

```
this.axios.post('http://localhost:8000/api/products', {
  ...this.filterValues,
  'page': page
})
.then(res => {
  console.log(res);
  this.products = res.data.data;
  this.paginationData = res.data.meta;
})
.finally(() => {
  this.loading = false;
});
```

Не менш головним при створенні онлайн-магазину є робота з базою даних, оплатою послуг, інтеграцією зі службами доставки. Крім цього, для повноцінного функціонування форм потрібні програми, написані серверними мовами, тому потрібен web-сервер.

### 4.3 Програмування сайту

При програмуванні сайту «Fashion Art» використовується web-сервер ХАМРР. ХАМРР – збірка вебсервера, в якій міститься набір додатків для повноцінного функціонування сайту. Нижче видно її панель налаштування. В ній, включивши «Config» та поставивши «галочки» біля необхідних інструментів, не потрібне налаштування кожного компонента окремо. Далі можна вести розробку на окремому комп'ютері без підключення до мережі. Інтерфейс програми користувача настільки простий, що її називають «складанням для ледачих» («lazy man's WAMP/LAMP installation»). Програма регулярно оновлюється для включення до складу новітніх версій Apache/MySQL/PHP і Perl. Тому, звісно, для програмування буде використовуватися мова PHP 8.

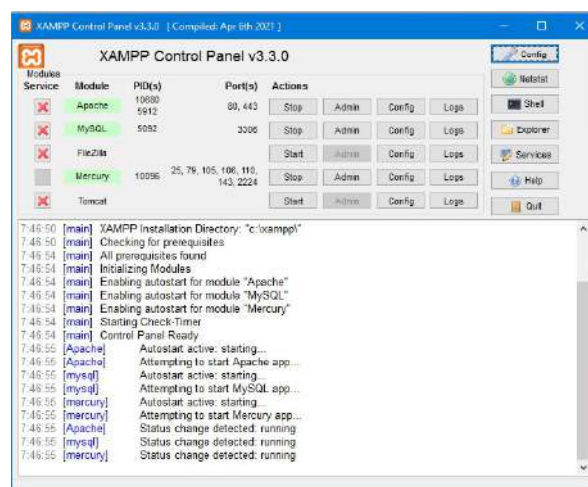


Рисунок 4.13 – Панель налаштування web-серверу ХАМРР

Особливостями мови програмування PHP є інтегрування з кодом HTML, а також взаємодія з мовою SQL. Це дає можливість віртуозно перемішувати в одному файлі елементи цих мов, отримуючи прекрасний результат. Приклад того, як це стається, є у лістингу 4.6.

#### Лістинг 4.6 – Фрагмент коду форми реєстрації, показаної на рисунку 4.14

```
<form class="text-center mt-2 p-5" method="post"
  action="admin.php?action=login" >
  <input type="hidden" name="login" value="true" />
  <?php if ( isset( $results['errorMessage'] ) ) { ?>
    <div class="errorMessage"><?php echo $results['errorMessage'] ?>
    </div>
  <?php } ?>
  <input type="text" id="FormName" class="form-control mb-4"
    name="name" placeholder="name">
  <input type="email" id="FormEmail" class="form-control mb-4"
    name="email" placeholder="Email">
  .....
</form>
```

В цьому коді втиснений скрипт PHP у HTML-форму. Цей скрипт за допомогою мови SQL перевіряє у таблиці імен бази phpMyAdmin введені користувачем дані. Якщо вони не співпадають, то з'являється повідомлення про помилку (рисунок 4.15).

The image shows a web form titled "REGISTRATION" in a bold, dark font. The form itself has a light orange background and a thin border. It contains four labeled input fields, each with a red asterisk indicating a required field: "Name" (containing "Ангеліна"), "Email" (containing "loglinn05@gmail.com"), "Phone" (containing "+380960536645"), and "Password" (containing masked characters "\*\*\*\*"). Below these fields is a dark orange button with the text "Registration" in white.

Рисунок 4.14 – Форма реєстрації користувача

**REGISTRATION**

You're not signed up.

\* Name  
Name

\* Email  
Email

\* Phone  
Phone

\* Password  
Password

Registration

Рисунок 4.15 – Повідомлення про помилку

Мовою PHP написано багато фреймворків та CMS, які використовуються для будування web-додатків. Серед них і фреймворк Laravel, на якому створюється цей проєкт.

Після того, як розглянено взаємодію з сайтом очима клієнта, потрібно розкрити незримую роботу серверних програм. Спочатку створено проєкт за допомогою однієї з команд менеджера пакетів Composer. Його структуру показано в графічному матеріалі.

Далі потрібно створити так звані «маршрути» Laravel, вигляд яких показаний у лістингу 3.1. Після цього створюємо контролери з методами для прийняття запитів з клієнтської сторони. Приклад контролера для отримання частини товарів (на одну сторінку) показаний у лістингу 4.7.

#### Лістинг 4.7 – Приклад контролера

```
<?php

namespace App\Http\Controllers\API\Product;

use App\Http\Controllers\Controller;
use App\Http\Requests\API\Product\FilterRequest;
use App\Http\Resources\Product\ProductResource;
use App\Models\Product;
use App\Custom\Filters\ProductFilter;

class IndexController extends Controller
{
    public function __invoke(FilterRequest $request)
```

```

    {
        $data = $request->validated();
        $filter = app()->make(ProductFilter::class, ['queryParams' =>
array_filter($data)]);
        $products = (new Product)->filter(Product::query(), $filter)
            ->paginate(8, ['*'], 'page', $data['page']);
        return ProductResource::collection($products);
    }
}

```

Ніякий магазин не обходиться без адмін-панелі, тому її обов'язково потрібно згадати. Її дизайн не був описаний, бо він доволі простий. До того ж, за її зовнішній вигляд буде більшою частиною відповідати AdminLTE. На рисунку 4.16 показаний вигляд головної сторінки панелі адміністрації.

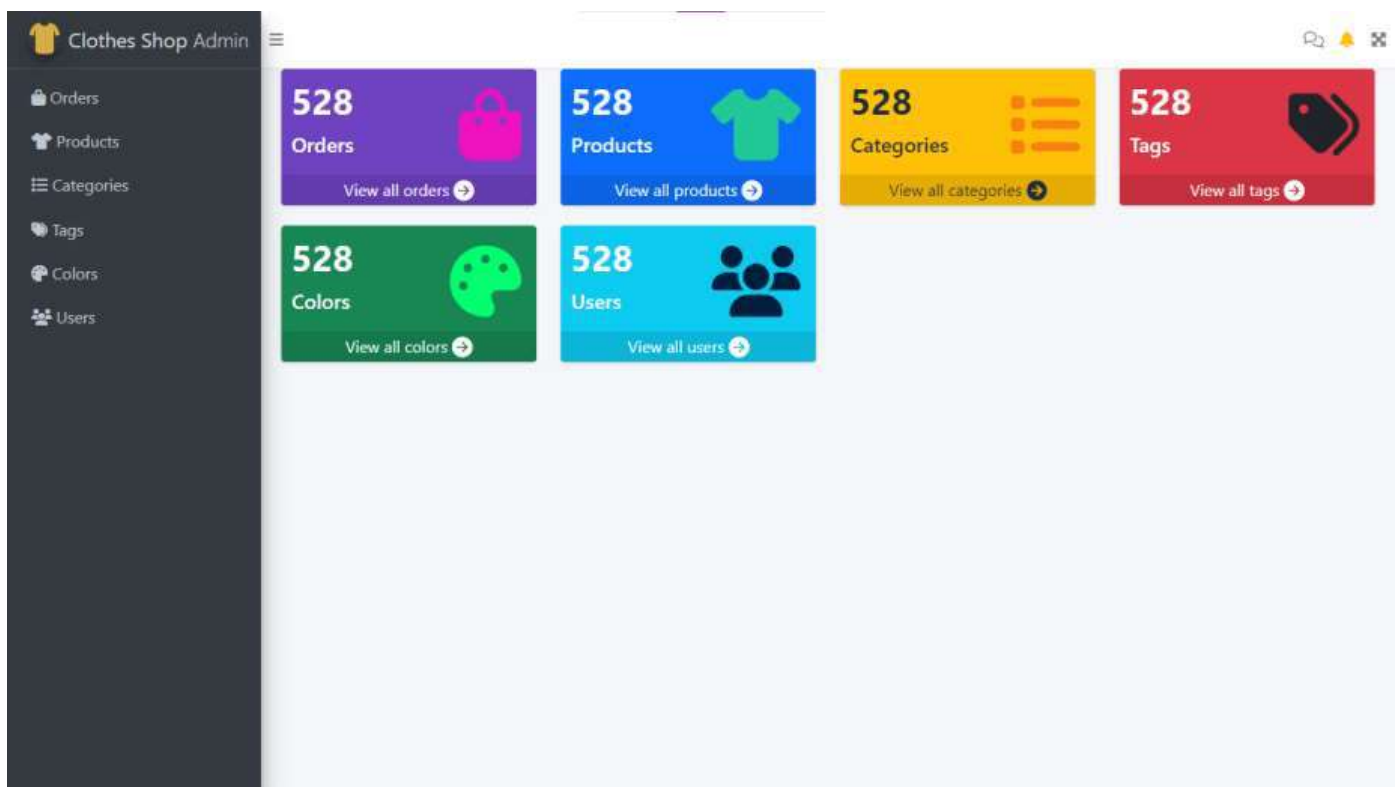


Рисунок 4.16 – Головна сторінка панелі адміністрації

Далі покажемо реалізацію вимог до інтерфейсу адмін-панелі на прикладі сторінок додавання, редагування та перегляду товару. Це все можна побачити на рисунках 4.17-4.23.

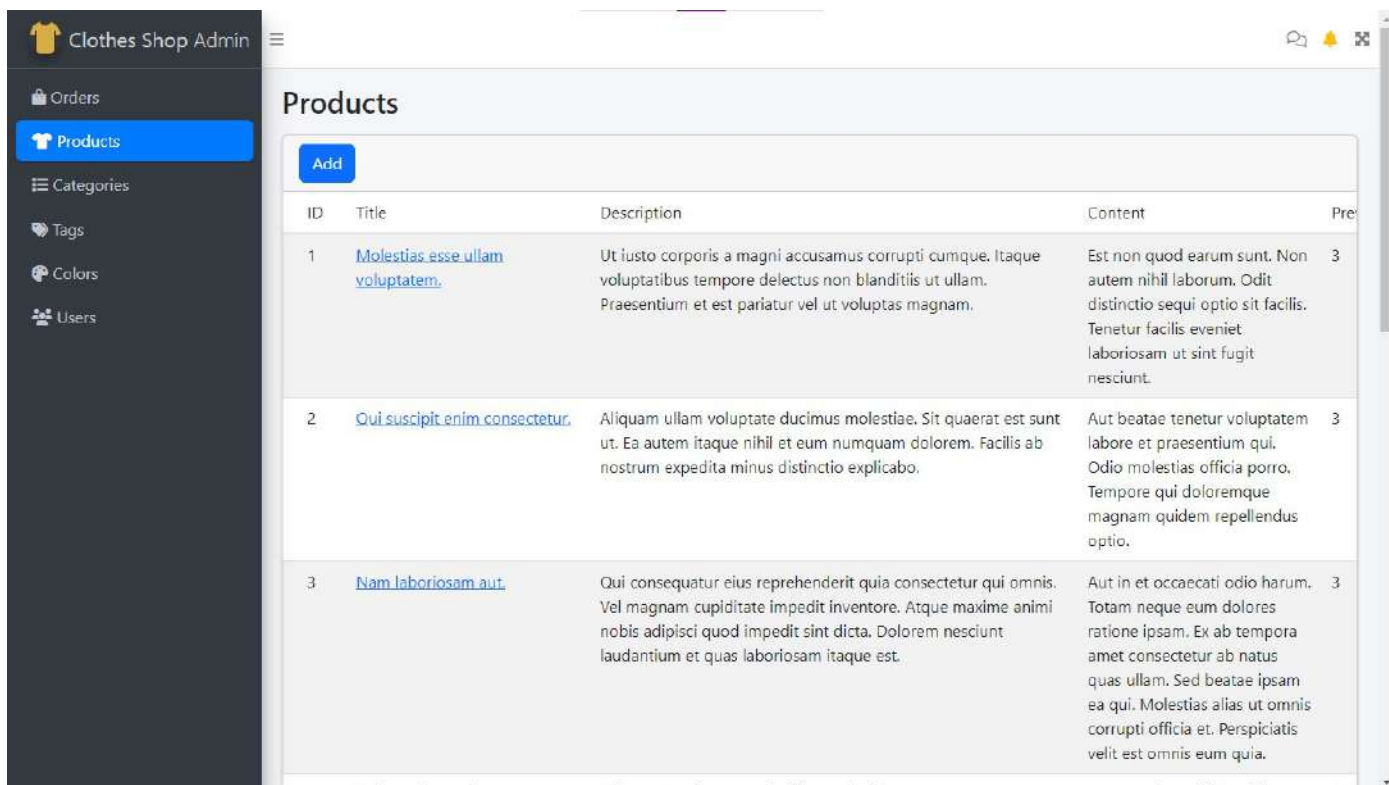


Рисунок 4.17 – Сторінка перегляду усіх товарів

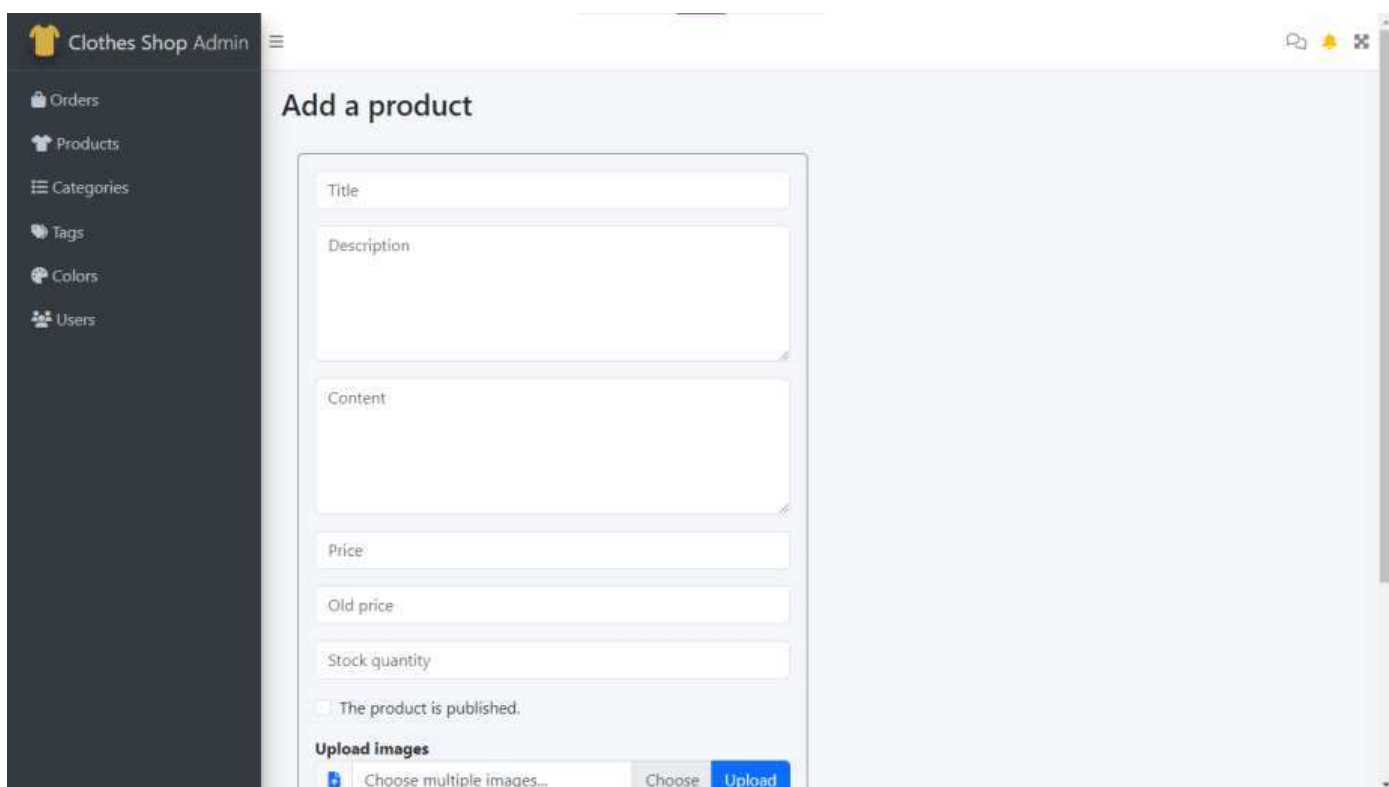


Рисунок 4.18 – Сторінка додавання товару (верхня частина)

Users

Content

Price

Old price

Stock quantity

☐ The product is published.

**Upload images**

Choose multiple images...

Select a category

Select tags

Select colors

Рисунок 4.19 – Сторінка додавання товару (нижня частина)

Clothes Shop Admin

Orders

Products

Categories

Tags

Colors

Users

**Product 1**

|               |                                                                                                                                                                        |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ID            | 1                                                                                                                                                                      |
| Title         | Molestias esse ullam voluptatem.                                                                                                                                       |
| Description   | Ut iusto corporis a magni accusamus corrupti cumque. Itaque voluptatibus tempore delectus non blanditiis ut ullam. Praesentium et est pariatur vel ut voluptas magnam. |
| Content       | Est non quod earum sunt. Non autem nihil laborum. Odit distinctio sequi optio sit facilis. Tenetur facilis eveniet laboriosam ut sint fugit nesciunt.                  |
| Preview image |                                                                                                                                                                        |
| Price         | 26                                                                                                                                                                     |
| Old price     |                                                                                                                                                                        |
| Count         | 1796                                                                                                                                                                   |
| Is published  |                                                                                                                                                                        |
| Category      | Accessories                                                                                                                                                            |
| Tags          | modi                                                                                                                                                                   |
|               | corporis                                                                                                                                                               |
|               | ut                                                                                                                                                                     |

Рисунок 4.20 – Сторінка перегляду товару (верхня частина)

|              |                                                                                                                                                                                                                                     |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Count        | 1796                                                                                                                                                                                                                                |
| Is published |                                                                                                                                                                                                                                     |
| Category     | Accessories                                                                                                                                                                                                                         |
| Tags         | modi<br>corporis<br>ut                                                                                                                                                                                                              |
| Colors       | <div>#b27188</div> <div>#6b83d7</div> <div>#7ec234</div> <div>#a09ab2</div> <div>#e1765f</div> <div>#57fdfa</div> <div>#3ad923</div> <div>#44c348</div> <div>#fc77df</div> <div>#3a43d0</div> <div>#a048c6</div> <div>#cab8f0</div> |

Рисунок 4.21 – Сторінка перегляду товару (нижня частина)

Clothes Shop Admin
 

- Orders
- Products
- Categories
- Tags
- Colors
- Users

## Edit a product

Щоб вийти із повноекранного режиму, натисніть **F11**

Molestias esse ullam voluptatem.

Ut iusto corporis a magni accusamus corrupti cumque. Itaque voluptatibus tempore delectus non blanditiis ut ullam. Praesentium et est pariatur vel ut voluptas magnam.

Est non quod earum sunt. Non autem nihil laborum. Odit distinctio sequi optio sit facilis. Tenetur facilis eveniet laboriosam ut sint fugit nesciunt.

26

Old price

1796

☐ The product is published.

Рисунок 4.22 – Сторінка редагування даних товару (верхня частина)

Рисунок 4.23 – Сторінка редагування даних товару (нижня частина)

Кожній з них відповідає один файл `.blade.php`. Blade – вбудований шаблонізатор Laravel. Всі подібні файли знаходяться у теці `resources/views`. Сторінки перегляду, додавання, редагування для деякого об'єкту або властивості знаходяться у підтеках з їх ім'ям, що можна побачити на структурі проєкту, зображеній на рисунку «Структура серверної частини на Laravel» у графічній частині, а вигляд файлу додавання знаходиться у лістингу 4.8.

#### Лістинг 4.8 – Файл сторінки додавання тегу

```
@extends('layouts.with-validation-errors')

@section('content_header')
    <h1>Add a tag</h1>
@stop

@section('main')
    <div class="row">
        <div class="col-6">
            <form
                action="{{ route('tag.store') }}"
                method="post"
                class="addition-form"
            >
                @csrf
                <div class="form-group">
                    <input type="text" name="title" class="form-control">
```

```

placeholder="Tag name" value="{{ old('title') }}">
</div>
<div class="form-group">
    <input type="submit" class="btn btn-primary" value="Add">
</div>
</form>
</div>
</div>
@stop

```

Валідують дані з форм так звані Request-класи, у яких містяться правила валідації. Існують Request-класи як для сторони клієнта, так і для адмін-панелі. Приклад такого класу показаний у лістингу 4.9.

#### Лістинг 4.9 – Клас для валідації даних з форми додавання товару

```

<?php

namespace App\Http\Requests\Product;

use Illuminate\Foundation\Http\FormRequest;

class StoreRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     */
    public function authorize(): bool
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array<string,
    \Illuminate\Contracts\Validation\Rule|array|string>
     */
    public function rules(): array
    {
        return [
            'title' => 'required|string|unique:products|max:255',
            'description' => 'required|string',
            'content' => 'required|string',
            'preview_images' => 'required|array|max:3',
            'preview_images.*' => 'image',
            'price' => 'required|numeric',
            'old_price' => 'required|numeric',
            'count' => 'required|numeric',
            'is_published' => 'nullable|boolean',
            'category_id' => 'required',
            'tags' => 'required|array',
            'colors' => 'required|array',
        ];
    }
}

```

Після проходження валідації дані відправляються у контролер. Це відбувається, якщо вони її пройшли.

Як вже відмічалось, досконало описати весь процес роботи фреймворків Laravel та Vue неможливо. Навіть стовідсоткове знання їхньої документації нічого не дає без практики програмування. У програмуванні головне – це досвід. І чим він більший, тим більше можна писати, бо з кожним новим проєктом відкриваються потаєні неочікувані куточки пізнання.

## ВИСНОВКИ

У результаті виконання дипломної роботи був розроблений інтернет-магазин одягу «Fashion Art», який допомагає магазину одягу збільшити прибутки та придбати та утримати лояльних клієнтів.

У дипломній роботі розкрито загальні теоретичні питання проєктування такі, як необхідність проєктування, загальна структура додатку та вибір класу засобів розробки.

Далі було сформульовано та описано вимоги до розробки програмного забезпечення онлайн-магазину. Це допомогло визначити, чому необхідно приділити увагу у процесі розробки та чого у цьому процесі робити не можна/треба.

Одним із ключових точок роботи стало моделювання за допомогою UML. У роботі було побудовано діаграми, які показують різні процеси інтернет-магазину та описано ситуації використання додатку (прецеденти). Такий підхід допомагає створити загальне уявлення про будову та функціонування системи.

У роботі також було розроблено архітектуру вебзастосунку, спроектовано користувацький інтерфейс та розроблено складові кожної сторінки інтернет-магазину.

У роботі описано весь процес розробки інтернет-магазину «Fashion Art», починаючи від проєктування і завершуючи скріншотами вже працюючого інтернет-магазину, із зазначенням програмного коду, який було для цього розроблено.

Розроблений інтернет-магазин підвищив прибуток компанії «Fashion Art» та кількість клієнтів.

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Діджиталізація бізнесу: перехід з наземного бізнесу в Інтернет - особливості процесу: [Електронний ресурс] – Режим доступу: <https://rasim.pro/uk/blog/didzhitalizacziya-biznesu-perehid-z-nazemnogo-biznesu-v-internet-osoblivosti-proczesu/>
2. Як пандемія та карантин прискорили діджиталізацію ритейлу в Україні: [Електронний ресурс] – Режим доступу: <https://delo.ua/business/jak-pandemija-ta-karantin-priskorili-didzhitaliz-373710/>
3. Як діджиталізація змінить процеси електронної комерції в 2023 році?: [Електронний ресурс] – Режим доступу: <https://isitlab.com/blog/impact-of-digital-technologies-on-e-commerce>
4. Правильна структура сайту: приклади, вимоги: [Електронний ресурс] – Режим доступу: <https://atriples.com.ua/pravylna-struktura-saytu/>
5. Правильна структура інтернет-магазину: [Електронний ресурс] – Режим доступу: <https://ilion.digital/ua/pravilna-struktura-internet-magazynu/>
6. Українське ІТ: добірка SaaS-сервісів для бізнесу: [Електронний ресурс] – Режим доступу: <https://serpstat.com/uk/blog/ukrayinski-saas-servisy-dlya-biznesu/#6>
7. Як обрати конструктор сайтів саме для ваших завдань: [Електронний ресурс] – Режим доступу: <https://leadgen.studio/uk/yak-obrati-konstruktor-sayt-v-same-dlya-vashih-zavdan/>
8. Огляд платформи Хорошоп. Плюси і мінуси сервісу: [Електронний ресурс] – Режим доступу: <https://konkurent.ua/publication/75670/oglyad-platformi-horoshop-plusi-i-minusi-servisu/>
9. Що таке CMS: [Електронний ресурс] – Режим доступу: <https://www.seotm.com/ua/blog/sho-take-cms.html>
10. Рейтинг CMS для інтернет-магазину: [Електронний ресурс] – Режим доступу: <https://brainlab.com.ua/uk/blog-uk/rejtyng-cms-dlya-internet-magazynu>

11. Переваги та недоліки WordPress: [Електронний ресурс] – Режим доступу: <https://infosoft.ua/ua/perevahy-i-nedoliky-wordpress>

12. Огляд CMS Joomla: що це, плюси та мінуси: [Електронний ресурс] – Режим доступу: <https://interkassa.com/blog/oglyad-cms-joomla-shho-ce-plyusi-ta-minusi>

13. CMS Drupal: огляд переваг і недоліків: [Електронний ресурс] – Режим доступу: <https://onehostplanet.ua/ua/news/cms-drupal-oglyad-perevag-i-nedolikiv>

14. Які мови програмування треба знати для створення сайтів?: [Електронний ресурс] – Режим доступу: <https://foxminded.ua/movy-prohramuvannia-dlia-stvorennia-saitiv/>

15. Top Backend Languages to Know in 2024: [Електронний ресурс] – Режим доступу: <https://herovired.com/learning-hub/blogs/back-end-languages/>

16. Що таке HTML та як за допомогою нього увійти в ІТ: [Електронний ресурс] – Режим доступу: <https://mc.today/uk/shho-take-html-ta-yak-za-dopomogoyu-nogo-uvijti-do-it/>

17. Тім Бернерс-Лі: [Електронний ресурс] – Режим доступу: <https://www.w3.org/People/Berners-Lee/>

18. Що таке CSS: [Електронний ресурс] – Режим доступу: [https://css.in.ua/article/shcho-take-html\\_10](https://css.in.ua/article/shcho-take-html_10)

19. 12 Best Open-Source CSS Frameworks 2024: [Електронний ресурс] – Режим доступу: <https://colorlib.com/wp/free-css3-frameworks/>

20. 9 актуальних CSS-фреймворків для Front-end розробника: [Електронний ресурс] – Режим доступу: <https://web-academy.ua/blog/junior/junior-9-css-frameworks-for-front-end>

21. Використання CSS фреймворку Bootstrap для адаптивної верстки: [Електронний ресурс] – Режим доступу: <https://web-developer.in.ua/assets/articles/bootstrap/bootstrap-responsive/bootstrap-responsive.html>

22. Використання Tailwind CSS для швидкої розробки стильового дизайну: [Електронний ресурс] – Режим доступу: <https://it-rating.ua/vikoristannya-tailwind-css-dlya-shvidkogo-rozrobki-stilovogo-dizaynu>

23. Що таке JavaScript і для чого він потрібен: [Електронний ресурс] – Режим доступу: <https://goit.global/ua/articles/shcho-take-javascript-i-dlia-choho-vin-potriben/>

24. Перші кроки в Angular: усе, що потрібно знати розробнику-початківцю: [Електронний ресурс] – Режим доступу: <https://foxminded.ua/angular/>

25. Що таке React JS? Як почати вивчати Реакт? Навички для react developer: [Електронний ресурс] – Режим доступу: <https://cases.media/article/sho-take-react-js-yak-pochati-vivchati-reakt-navichki-dlya-react-developer>

26. Vue.js як JavaScript-фреймворк: [Електронний ресурс] – Режим доступу: <https://foxminded.ua/vue-js/>

27. Порівнюємо React, Angular і Vue — найпопулярніші бібліотеки й фреймворки у 2022 році: [Електронний ресурс] – Режим доступу: <https://dou.ua/forums/topic/39933/>

28. Що таке Typescript і навіщо він потрібен: [Електронний ресурс] – Режим доступу: <https://foxminded.ua/typescript/>

29. CoffeeScript: Pros & Cons: [Електронний ресурс] – Режим доступу: [https://www.linkedin.com/pulse/coffeescript-pros-cons-shahriar-kabir?trk=articles\\_directory](https://www.linkedin.com/pulse/coffeescript-pros-cons-shahriar-kabir?trk=articles_directory)

30. Створення сайтів і вебдодатків на Dart: [Електронний ресурс] – Режим доступу: <https://brander.ua/technologies/dart>

31. «Вбити JavaScript не вийшло»: за що ми любимо Dart — мову створення застосунків на Flutter (та кому вона підходить): [Електронний ресурс] – Режим доступу: <https://highload.today/uk/blogs/za-shho-my-lyubymo-dart-golovnu-ta-yedynu-movu-stvorenniya-dodatkov-na-flutter/>

32. What is PHP?: [Електронний ресурс] – Режим доступу: <https://www.php.net/manual/en/intro-what-is.php>

33. Backend розробник: шлях до освоєння “зворотного” боку веброзробки: [Електронний ресурс] – Режим доступу: <https://foxminded.ua/backend-rozrobnyk/>

34. Як стати PHP-розробником. План дій для початківців: [Електронний ресурс] – Режим доступу: <https://dou.ua/lenta/articles/how-to-learn-php/>

35. Що таке PHP?: [Електронний ресурс] – Режим доступу: <https://freehost.com.ua/ukr/faq/wiki/chto-takoe-php/>

36. Переваги та недоліки PHP-фреймворків: [Електронний ресурс] – Режим одоступу: <https://foxminded.ua/php-freimvorky/>

37. Що таке Laravel? Огляд переваг для веброзробки: [Електронний ресурс] – Режим доступу: <https://asabix.com.ua/what-is-laravel/>

38. 5 головних переваг Laravel при розробці інтернет-магазинів: [Електронний ресурс] – Режим доступу: <https://marketer.ua/ru/5-glavnykh-preimushestv-laravel-pri-razrabotke-internet-magazinov>

39. Using Symfony for a wide variety of projects: [Електронний ресурс] – Режим доступу: <https://www.trustradius.com/reviews/symfony-2021-05-09-16-09-45>

40. Що таке CodeIgniter? Як працює PHP CI Framework?: [Електронний ресурс] – Режим доступу: <https://www.guru99.com/uk/what-is-codeigniter.html>

41. Що таке Yii?: [Електронний ресурс] – Режим доступу: <https://yii2-framework.readthedocs.io/en/stable/guide-uk/intro-yii/>

42. Що таке Python і для чого він потрібен: [Електронний ресурс] – Режим доступу: <https://goit.global/ua/articles/shcho-take-python-i-dlia-choho-vin-potriben/>

43. Що таке мова програмування Python?: [Електронний ресурс] – Режим доступу: <https://freehost.com.ua/ukr/faq/wiki/chto-takoe-jazik-programmirovanija-python/>

44. Вступ до Python фреймворків: які вони бувають і для чого використовуються: [Електронний ресурс] – Режим доступу: <https://foxminded.ua/freimvorky-python/>

45. Popular Python Frameworks for Web Development: [Електронний ресурс] – Режим доступу: <https://www.browserstack.com/guide/top-python-web-development-frameworks>

46. Навіщо потрібна мова програмування Ruby?: [Електронний ресурс] – Режим доступу: <https://lemon.school/blog/navishho-potribna-mova-programuvannya-ruby>

47. Pros & Cons of Ruby on Rails You Should Know Before Choosing the Technology for Your Startup: [Електронний ресурс] – Режим доступу: <https://sloboda-studio.com/blog/pros-and-cons-of-ruby-on-rails/>

48. Переваги фреймворку Sinatra: [Електронний ресурс] – Режим доступу: <https://rubydevelopers.org/t/sinatra/172>

49. Головне про фреймворк Hanami: [Електронний ресурс] – Режим доступу: <https://rubydevelopers.org/t/hanami/151>

50. Помилки при виготовленні інтернет магазину: [Електронний ресурс] – Режим доступу: <https://xn----7sbbaqhlkm9ah9aiq.net/ua/news-new/oshibki-pri-sozdanii-internet-magazina.html>

51. Платіжні системи для інтернет-магазину: як обрати: [Електронний ресурс] – Режим доступу: <https://brander.ua/blog/platizhni-sistemi-dlya-internet-magazinu-yak-obrati>

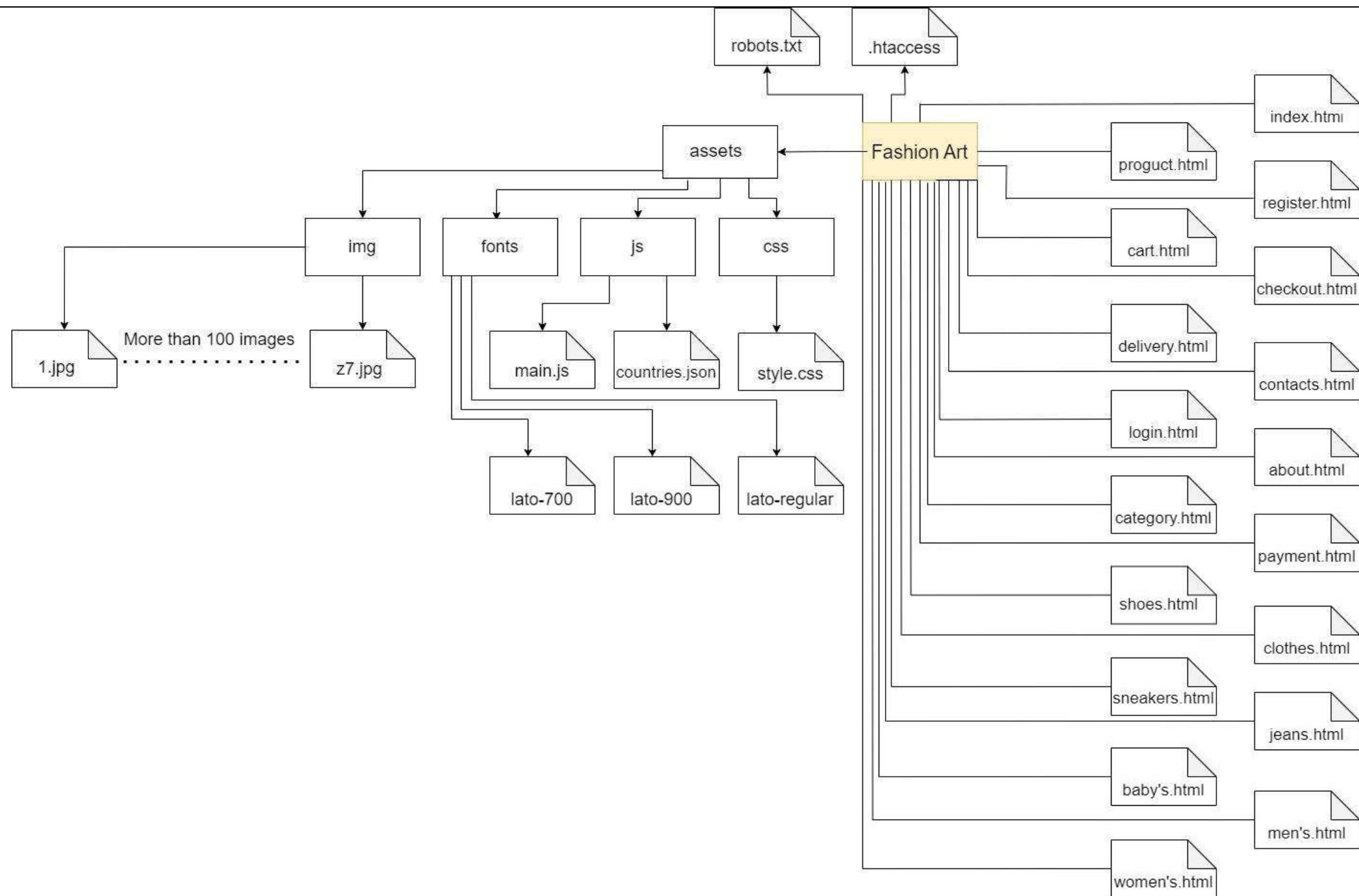
52. Прийом платежів на сайті: як приймати оплату в інтернет-магазині: [Електронний ресурс] – Режим доступу: <https://tranzzo.com/uk-ua/blog/pryiom-platezhiv-na-saiti-yak-pryimaty-oplatu-v-internet-mahazyni>

53. Оптимізація продуктивності сайту: Поради та методи: [Електронний ресурс] – Режим доступу: <https://www.ranktracker.com/uk/blog/optimizing-website-performance-tips-and-techniques/>

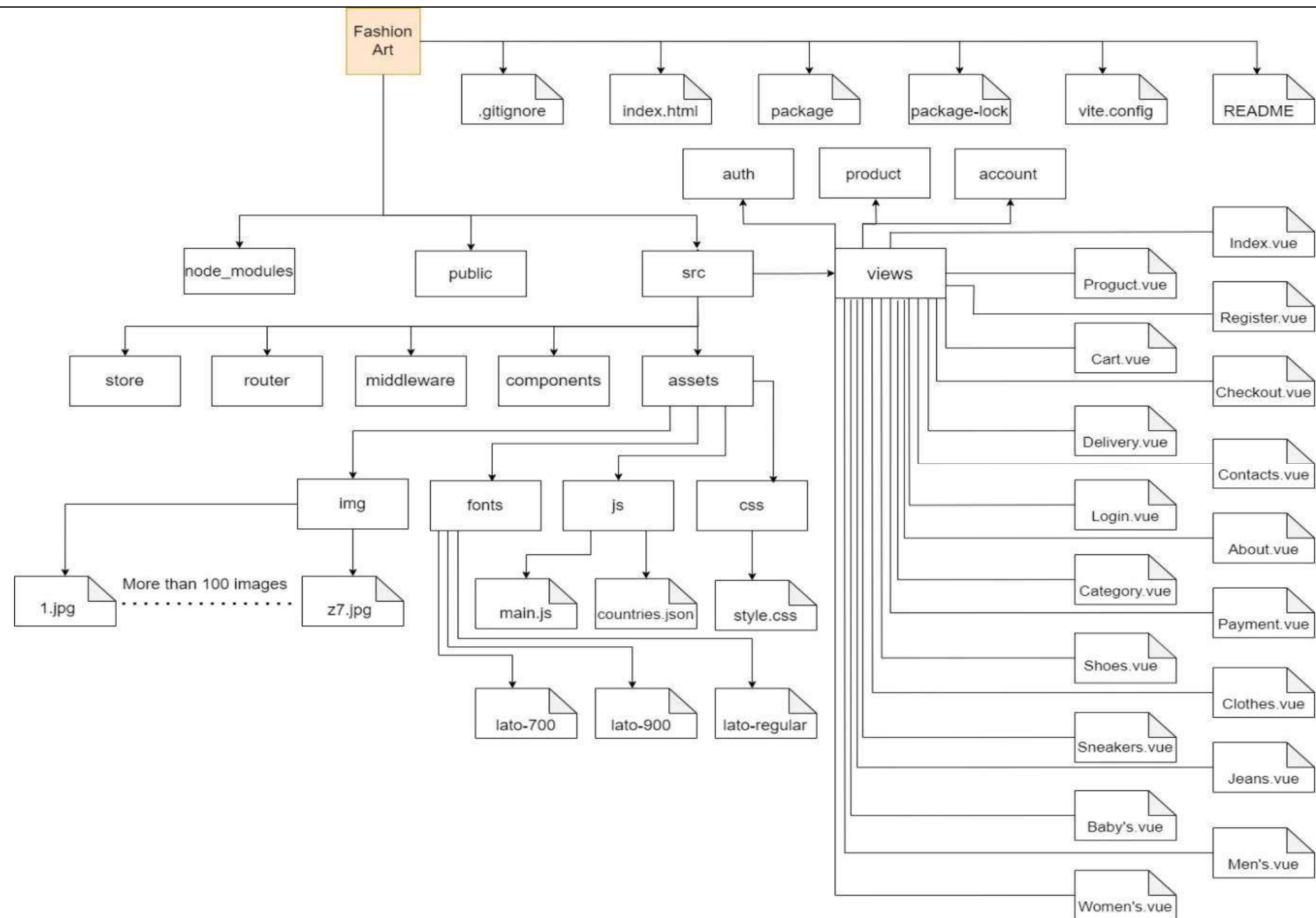
54. Larman, Craig. Applying UML and patterns [Текст] : 3rd edition, Craig Larman, foreword by Philippe Kruchten : Pearson, 2004. – 736 с. : іл. – ISBN 978-0131489066.

55. Ознайомлення з патерном MVC (Model-View-Controller): [Електронний ресурс] – Режим доступу: <https://javarush.com/ua/groups/posts/uk.2536.chastina-7-oznayomlennja-z-paternom-mvc-model-view-controller>

56. Розробка мобільних додатків на MVVM: [Електронний ресурс] – Режим доступу: <https://brander.ua/technologies/mvvm>

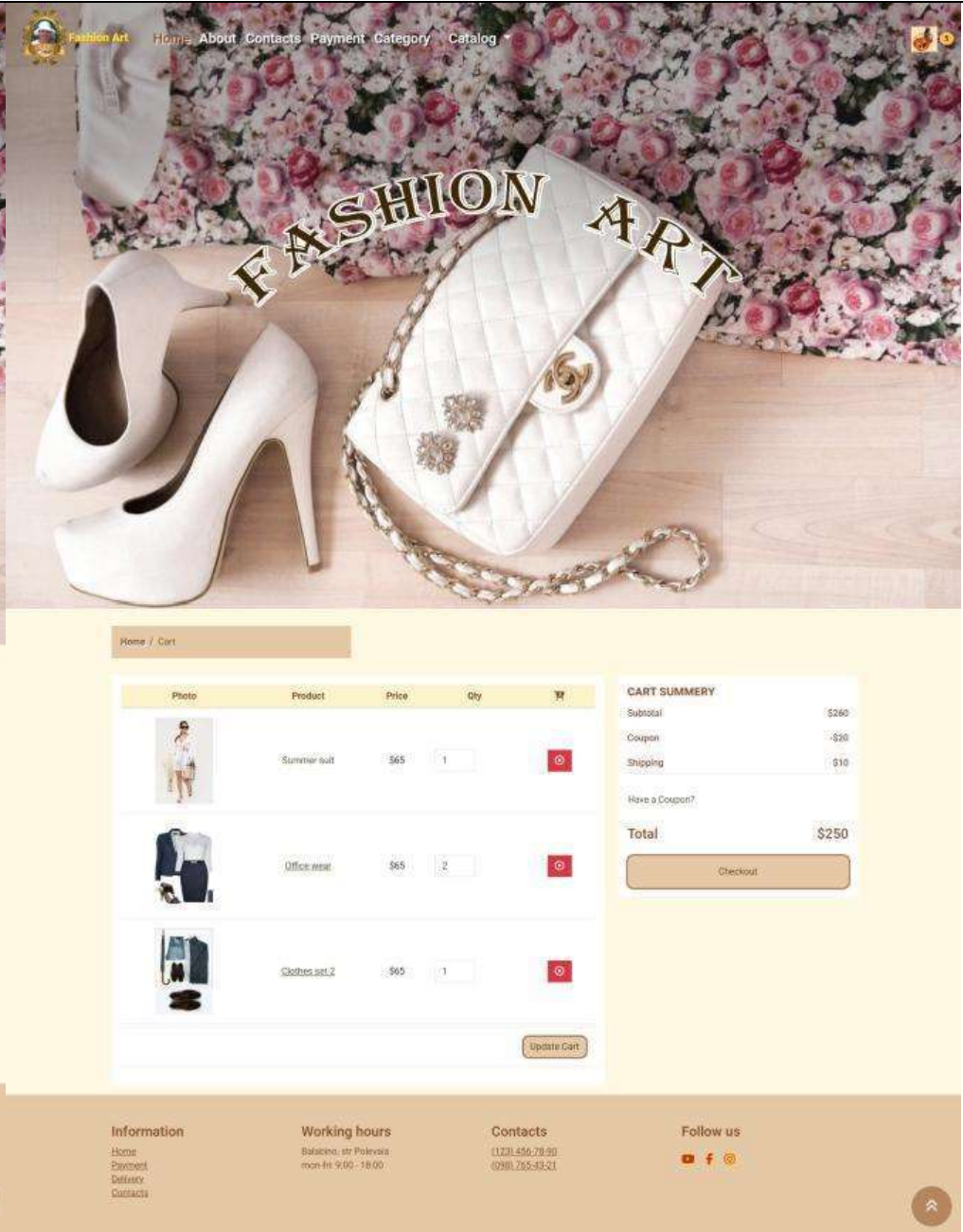
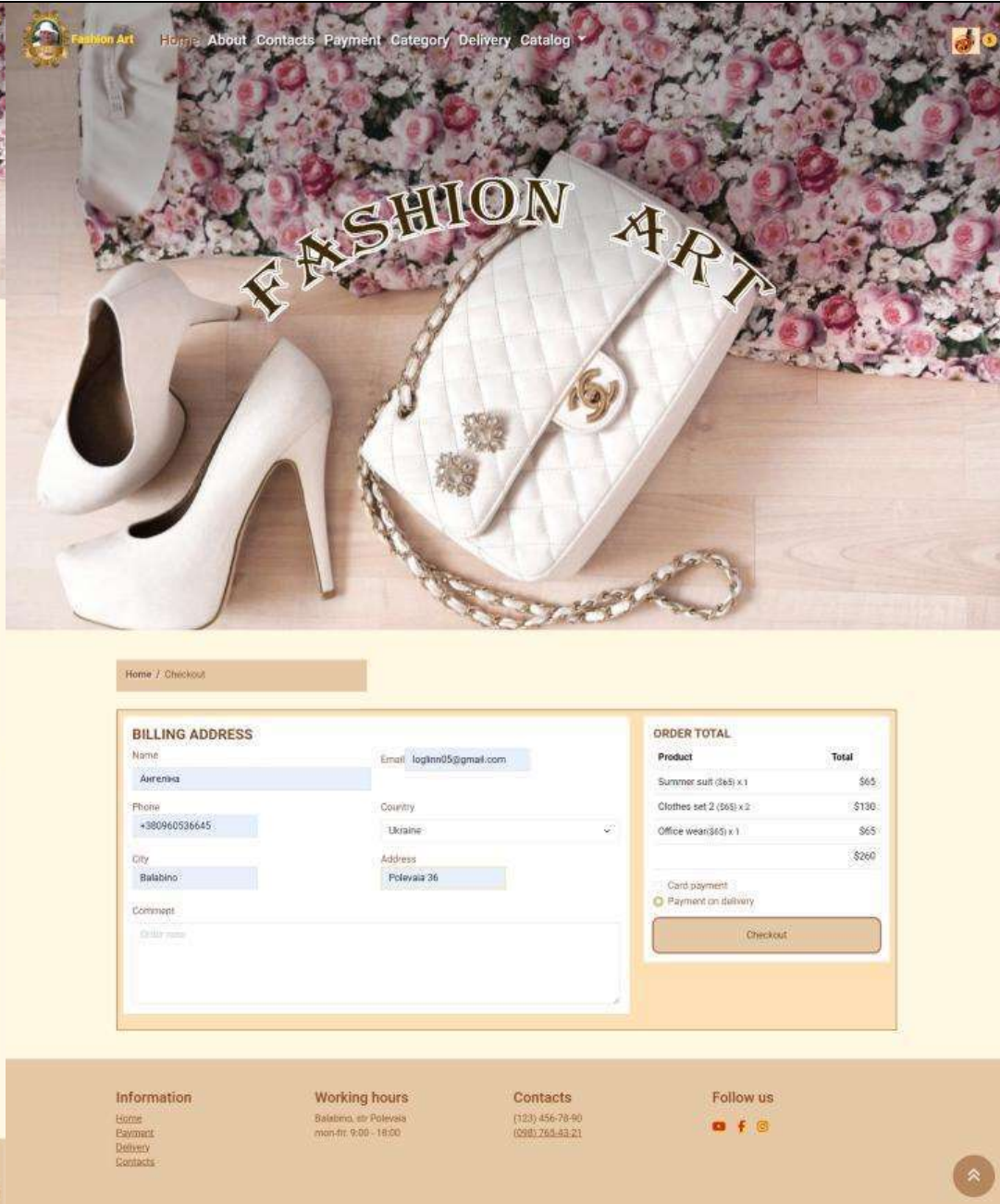


|           |            |          |       |      |  |                                                                          |                                              |  |        |        |  |
|-----------|------------|----------|-------|------|--|--------------------------------------------------------------------------|----------------------------------------------|--|--------|--------|--|
|           |            |          |       |      |  | 13.02070849.00055 ПЛ1                                                    |                                              |  |        |        |  |
|           |            |          |       |      |  | Розробка інтернет-магазину<br>«Fashion Art»<br>Структура теки з версткою | Лім.                                         |  | Маса   | Маштаб |  |
| Зм.       | Лист       | № докум. | Підп. | Дата |  |                                                                          |                                              |  |        |        |  |
| Розроб.   | Логвина    |          |       |      |  |                                                                          |                                              |  |        |        |  |
| Перевір.  | Тягунова   |          |       |      |  |                                                                          |                                              |  |        |        |  |
| Т. контр. |            |          |       |      |  |                                                                          | Лист                                         |  | Листів |        |  |
|           |            |          |       |      |  |                                                                          | НУ «Запорізька<br>політехніка», гр. КНТЗ-510 |  |        |        |  |
| Н. контр. | Польська   |          |       |      |  |                                                                          |                                              |  |        |        |  |
| Затвердж. | Кудерметов |          |       |      |  |                                                                          |                                              |  |        |        |  |



|           |            |          |       |      |                                                                                        |                                              |  |        |
|-----------|------------|----------|-------|------|----------------------------------------------------------------------------------------|----------------------------------------------|--|--------|
|           |            |          |       |      | 13.02070849.00055 ПЛ2                                                                  |                                              |  |        |
|           |            |          |       |      | Розробка інтернет-магазину<br>«Fashion Art»<br>Структура клієнтської частини на<br>Vue | Лім. Маса Маштаб                             |  |        |
| Зм.       | Лист       | № докум. | Підп. | Дата |                                                                                        |                                              |  |        |
| Розроб.   | Логвина    |          |       |      |                                                                                        |                                              |  |        |
| Перевір.  | Тягунова   |          |       |      |                                                                                        |                                              |  |        |
| Т. контр. |            |          |       |      |                                                                                        | Лист                                         |  | Листів |
| Н. контр. | Польська   |          |       |      |                                                                                        | НУ «Запорізька<br>політехніка», гр. КНТЗ-510 |  |        |
| Затвердж. | Кудерметов |          |       |      |                                                                                        |                                              |  |        |





|           |            |          |       |      |                                                                                                                                                                       |                                              |  |  |        |  |        |  |
|-----------|------------|----------|-------|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------|--|--|--------|--|--------|--|
|           |            |          |       |      | 13.02070849.00055 ПЛ4                                                                                                                                                 |                                              |  |  |        |  |        |  |
|           |            |          |       |      | Розробка інтернет-магазину<br>«Fashion Art»<br><br>Сторінки інтернет-магазину:<br>головна сторінка, сторінка з одягом,<br>сторінка кошику та оформлення<br>замовлення | Лім.                                         |  |  | Маса   |  | Маштаб |  |
| Зм.       | Лист       | № докум. | Підп. | Дата |                                                                                                                                                                       |                                              |  |  |        |  |        |  |
| Розроб.   | Логвина    |          |       |      |                                                                                                                                                                       |                                              |  |  |        |  |        |  |
| Перевір.  | Тягунова   |          |       |      |                                                                                                                                                                       |                                              |  |  |        |  |        |  |
| Т. контр. |            |          |       |      |                                                                                                                                                                       |                                              |  |  |        |  |        |  |
|           |            |          |       |      |                                                                                                                                                                       | Лист                                         |  |  | Листів |  |        |  |
| Н. контр. | Польська   |          |       |      |                                                                                                                                                                       | НУ «Запорізька<br>політехніка», гр. КНТЗ-510 |  |  |        |  |        |  |
| Затвердж. | Кудерметов |          |       |      |                                                                                                                                                                       |                                              |  |  |        |  |        |  |