

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»  
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ  
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## **ПОЯСНЮВАЛЬНА ЗАПИСКА**

до кваліфікаційної роботи освітнього ступеня «бакалавр»  
за спеціальністю 121 «Інженерія програмного забезпечення»  
(освітня програма «Інженерія програмного забезпечення»)

на тему:

### **«Розробка інтернет-магазину»**

Виконав студент групи ЗІПЗ-19-1  
МЕЛЬНИК Богдан Леонідович

Керівник роботи:  
МОРОЗОВ Андрій Васильович

Рецензент:  
ВАКАЛЮК Тетяна Анатоліївна

Житомир – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»  
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ  
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

**«ЗАТВЕРДЖУЮ»**

В.о. зав. кафедри інженерії  
програмного забезпечення  
Андрій МОРОЗОВ  
«14» лютого 2023 р.

**ЗАВДАННЯ**  
**на кваліфікаційну роботу**

Здобувач вищої освіти: **МЕЛЬНИК Богдан Леонідович**

Керівник роботи: **МОРОЗОВ Андрій Васильович**

Тема роботи: **«Розробка інтернет-магазину»,**

затверджена Наказом закладу вищої освіти від «14» лютого 2023 р., N71/с

Вихідні дані для роботи: **Розробка інтернет-магазину для продажу товарів або послуг**

Предмет дослідження: Сайт інтернет-магазину.

Об'єкт дослідження: Особливості функціонування та вплив сайту інтернет-магазину на покупців та бізнес-процеси.

Консультанти з бакалаврської роботи із зазначенням розділів, що їх стосуються:

| Розділ | Консультант  | Завдання видав | Завдання прийняв |
|--------|--------------|----------------|------------------|
| 1      | Морозов А.В. | 05.03.2023р.   | 05.03.2023р.     |
| 2      | Морозов А.В. | 03.04.2023р.   | 03.04.2023р.     |
| 3      | Морозов А.В. | 05.05.2023р.   | 05.05.2023р.     |

## РЕФЕРАТ

Пояснювальна записка до проекту на тему "Інтернет-магазин" складається з вступу, трьох розділів, висновків, списку використаних джерел та додатку. У списку використаних джерел міститься 9 найменувань, займаючи 1 сторінку. В роботі наведено 28 рисунків. Загальний обсяг роботи становить 61 сторінку. У першому розділі було обґрунтовано та проаналізовано процес створення інтернет-магазину. Були досліджені та висвітлені основні аспекти, пов'язані з його створенням та функціональними вимогами. У другому розділі проведено проектування і розробку програмного продукту, пов'язаного з інтернет-магазином. Була розроблена архітектура програми, вибрані технології та інструменти для реалізації, а також описані основні функції та можливості програмного продукту. У третьому розділі проведено тестування розробленого програмного продукту. Були оцінені його функціональні можливості, виконані тестові сценарії та виявлені можливі помилки чи недоліки. У висновку містяться результати виконаної роботи зі створення "Інтернет-магазину". Були підведені підсумки роботи, проаналізовані отримані результати та сформульовані висновки щодо досягнутих цілей та виконаних завдань. У додатку представлений лістинг розробленого програмного продукту, який містить кодову базу та структуру програми. Ключові слова, пов'язані з проектом, включають РНР, MYSQL, WEB-САЙТ, MVC, СЕРВЕР, АДМІН, АВТОРИЗАЦІЯ, МАГАЗИН,

|                  |             |                 |               |             |                                                               |                                             |             |                |
|------------------|-------------|-----------------|---------------|-------------|---------------------------------------------------------------|---------------------------------------------|-------------|----------------|
|                  |             |                 |               |             | ІПЗ.КР.Б – 121 – 23 – ПЗ                                      |                                             |             |                |
| <b>Змн.</b>      | <b>Арк.</b> | <b>№ докум.</b> | <b>Підпис</b> | <b>Дата</b> | <u>Розробка інтернет-магазину</u><br><br>Пояснювальна записка | <b>Літ.</b>                                 | <b>Арк.</b> | <b>Аркушів</b> |
| <b>Розроб.</b>   |             | Мельник Б.Л.    |               |             |                                                               |                                             | 3           | 61             |
| <b>Керівник</b>  |             | Морозов А.В.    |               |             |                                                               |                                             |             |                |
| <b>Реценз.</b>   |             | Вакалюк В.А.    |               |             |                                                               |                                             |             |                |
| <b>Н. Контр.</b> |             | Тичина О.П.     |               |             |                                                               |                                             |             |                |
| <b>Зав.каф..</b> |             | Морозов А.В.    |               |             |                                                               | Житомирська політехніка,<br>група ЗІПЗ-19-1 |             |                |

## ABSTRACT

The explanatory note to the project on the topic "Online store" consists of an introduction, three chapters, conclusions, a list of references and an appendix. The list of references contains 9 titles, occupying 1 page. The paper contains 28 figures. The total volume of the work is 61 pages. The first chapter substantiates and analyzes the process of creating an online store. The main aspects related to its creation and functional requirements were investigated and highlighted. The second section describes the design and development of the software product related to the online store. The program architecture was developed, technologies and tools for implementation were selected, and the main functions and capabilities of the software product were described. In the third section, the developed software product was tested. Its functionality was evaluated, test scenarios were executed, and possible errors or shortcomings were identified. The conclusion contains the results of the work done to create the "Online Store". The results of the work were summarized, the obtained results were analyzed, and conclusions were formulated regarding the goals and objectives achieved. The appendix contains a listing of the developed software product, which includes the code base and program structure. The keywords related to the project include PHP, MYSQL, WEB-SITE, MVC, SERVER, ADMIN, AUTHORIZATION, SHOP

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 4    |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

## ЗМІСТ

|                                                                                                          |    |
|----------------------------------------------------------------------------------------------------------|----|
| РЕФЕРАТ .....                                                                                            | 3  |
| РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ<br>ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ ІНТЕРНЕТ-МАГАЗИНУ ..... | 9  |
| 1.1. Перелік задач, що підлягають автоматизації у предметній області .....                               | 9  |
| 1.2. Існуючі аналоги програмного продукту .....                                                          | 11 |
| 1.3. Архітектура та узагальнена структура, майбутнього програмного комплексу<br>.....                    | 14 |
| 1.4. Інструментальні засоби реалізації програмного продукту та вимоги до<br>апаратних засобів. ....      | 17 |
| Висновки до першого розділу.....                                                                         | 23 |
| РОЗДІЛ 2. ПРОЕКТУВАННЯ ТА РОЗРОБКА ІНТЕРНЕТ-МАГАЗИНУ.....                                                | 24 |
| 2.1. Опис майбутнього програмного комплексу.....                                                         | 24 |
| 2.2. Варіанти використання системи.....                                                                  | 25 |
| 2.3. Структура системи збереження даних та алгоритми роботи системи .....                                | 26 |
| 2.4. Структуру модулів та класів системи. ....                                                           | 30 |
| 2.5. Контролери системи. ....                                                                            | 40 |
| Висновки до другого розділу .....                                                                        | 42 |
| РОЗДІЛ 3. ОПИС РОБОТИ З ПРОГРАМНИМ ДОДАТКОМ ТА ЙОГО<br>ТЕСТУВАННЯ МОДУЛІВ ТА КЛАСІВ СИСТЕМИ.....         | 43 |
| 3.1. Опис роботи з програмним додатком .....                                                             | 43 |
| 3.2. Тестування роботи програмного забезпечення.....                                                     | 56 |
| Висновки до третього розділу.....                                                                        | 59 |
| ВИСНОВКИ.....                                                                                            | 61 |
| СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....                                                                          | 62 |

## ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД – База даних – сукупність даних, що зберігаються відповідно до схеми даних, маніпулювання якими виконують відповідно до правил засобів моделювання даних.

MVC – Model-View-Controller – архітектурний шаблон, який використовується під час проектування та розробки програмного забезпечення.

CMS – Content Management System – система керування вмістом сайту.

HTML – HyperText Markup Language – стандартизована мова розмітки документів для перегляду вебсторінок у браузері.

CSS – Cascading Style Sheets – формальна мова опису зовнішнього вигляду документа, написаного з використанням мови розмітки.

PHP – Hypertext Preprocessor – скриптова мова загального призначення, інтенсивно застосовується для розробки веб-додатків.

JS – JavaScript – динамічна, об'єктно-орієнтована прототипна мова програмування.

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 6    |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

## ВСТУП

Актуальність теми. Використання веб-технологій постійно зростає у всіх сферах людського життя. Веб-сайт є незамінним інтерфейсом між підприємством та його оточенням, включаючи партнерів, постачальників та клієнтів. Створення веб-сайту має вирішальне значення для підприємницької діяльності, особливо в Інтернеті.

Зараз все більше компаній використовують Інтернет для обміну інформацією та оптимізації співпраці з клієнтами. Створення сайту є найбільш поширеною формою присутності в мережі.

Структурована інформація на веб-сайті надає користувачам повний огляд про підприємство його діяльності. Користувачі можуть отримати цінну інформацію про товари та послуги, а також безпосередньо замовити їх, використовуючи сайт. Веб-сайт є ефективним засобом комунікації з потенційними клієнтами, що набагато переважає звичайні методи, такі як телефон або факс. Завдяки інтуїтивно зрозумілому інтерфейсу та навігації сайту, клієнти можуть самостійно обирати необхідну інформацію, яка їх цікавить. Сайт доповнює рекламну кампанію підприємства, надаючи можливість взаємодії з клієнтами та забезпечуючи зворотний зв'язок.

Метою даної роботи є розробка інтернет-магазину з використанням HTML, CSS, JavaScript, а також розробка системи управління засобами мови програмування PHP та бази даних SQL. Веб-сайт повинен забезпечувати функціонал для повної й безперебійної роботи та взаємодіяти як з комп'ютерними платформами, так і з мобільними. Також метою є поглиблення теоретичних та практичних знань і сформування необхідних навичок і вмінь для розробки веб-сайту.

Поставлене завдання передбачає виконання наступних пунктів:

- провести аналіз функціональних потреб інтернет-магазину;
- визначити технології реалізації веб-сервісу;

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 7    |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

- обрати оптимальну архітектуру програмного комплексу, враховуючи сучасні умови;
- створити базу даних для збереження інформації інтернет-магазину;
- визначити об'єктну структуру веб-сервісу інтернет-магазину;
- реалізувати систему управління контентом.

Об'єктом дослідження є методи та засоби розробки веб-сайту відповідно до визначених предметних областей. Предметом дослідження є моделі, технології, структури, засоби та методи розробки та взаємодії веб-сайтів та модулів управління сайтом, а також необхідні веб-орієнтовані технології. При розробці були використані методи об'єктно-орієнтованого проектування, веб-програмування та дизайну користувацьких інтерфейсів.

Розроблена програмна система може бути використана як практичний веб-сайт з можливістю подальшого розширення функціоналу. Випускна робота складається зі вступу, трьох розділів, висновків, списку використаних джерел (9 найменувань), трьох додатків та 28 ілюстрації в тексті. Загальний обсяг роботи становить 61 сторінку.

# РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ ІНТЕРНЕТ-МАГАЗИНУ

## 1.1. Перелік задач, що підлягають автоматизації у предметній області

Автоматизація - це використання технологій та спеціального програмного забезпечення з метою заміщення або сприяння людській праці у виконанні різноманітних завдань. У сфері електронної комерції автоматизація виявляється надзвичайно корисною для полегшення і оптимізації численних процесів. Завдяки автоматизації забезпечується автоматичне виконання рутинних операцій, що раніше вимагали великої кількості часу та зусиль. Одним із основних аспектів автоматизації в електронній комерції є автоматизація процесу збору та обробки даних. Завдяки спеціальним програмним рішенням можна автоматично збирати, категоризувати та аналізувати великі обсяги даних, що надходять з різних джерел. Це дозволяє швидше та ефективніше реагувати на зміни в ринкових умовах, а також приймати обґрунтовані рішення щодо стратегії продажів та маркетингових активностей. В контексті електронної комерції, автоматизація може бути використана для полегшення і оптимізації різних процесів, таких як:

1. Автоматизація процесу замовлення товару - використання інтернет-форм або мобільних додатків для зручного і швидкого замовлення товарів.
2. Автоматизація процесу оплати - використання систем онлайн-оплати, таких як PayPal чи Stripe, для автоматизації процесу оплати товарів.
3. Автоматизація системи розсилки листів/sms для підтвердження/сповіщення про замовлення є важливим аспектом ефективної комунікації з клієнтами. Існує кілька кроків, які можна виконати для автоматизації цього процесу.
4. Автоматизація системи рекомендації товарів також є важливою функцією для залучення і залишення клієнтів. Ця система може аналізувати покупки

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 9    |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

клієнтів і рекомендувати їм товари, які можуть їх зацікавити, надаючи їм змогу знайти товари, які є найбільш відповідними для їх потреб і інтересів.

5. Автоматизація процесу обробки відгуків і відгуків клієнтів. Це може включати використання програмного забезпечення для автоматичного аналізу відгуків клієнтів, їх класифікації та своєчасної відповіді на них
6. Автоматизація процесу створення спеціальних пропозицій і рекламних акцій. Використовуючи інструменти автоматизації, підприємства можуть швидко й легко створювати та запускати спеціальні пропозиції та рекламні акції для певних сегментів клієнтів.
7. Автоматизація систем управління запасами. Автоматизація може допомогти підприємствам відстежувати рівень своїх запасів, змінювати замовлення товарів, коли вони закінчуються, і навіть прогнозувати майбутній попит на продукти.
8. Автоматизація процесу повернення та відшкодування - автоматизація цього процесу може допомогти підприємствам швидко та ефективно обробляти запити клієнтів і зменшити ризик помилок.
9. Автоматизація систем класифікації продуктів. Автоматизація може допомогти підприємствам класифікувати свої продукти та полегшити клієнтам пошук того, що вони шукають.
10. Автоматизація системи підтвердження/відхилення замовлень адміністратором - це може включати використання автоматизації для швидкої та ефективної обробки запитів клієнтів на замовлення продукції.

Загалом автоматизація може допомогти підвищити ефективність, зменшити витрати та підвищити рівень задоволеності клієнтів шляхом оптимізації процесів і зменшення ризику помилок.

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 10   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

## 1.2. Існуючі аналоги програмного продукту

Майже в кожного великого бізнесу та людини є веб-сайт, в якому вони рекламують, продають продукцію та інше, кожна з них має свій функціонал, але він в більшості схожий. В цьому розділі будемо опишемо та проаналізуємо їх по певним критеріям. Кожен критерій буде мати свою оцінку, після чого підсумуємо і виведемо оцінку.

1. Зручний та інтуїтивний інтерфейс сайту – одна з перших речей яку помічає користувач. Інтерфейс – одна з головних речей веб-сайту, адже від цього залежить бажання користувача використовувати сайтом. (5 балів)

2. Технічна підтримка – в будь який момент у користувача може виникнути питання чи проблема, яку він не здатен розв'язати самотійно. Проблема може виникнути в будь-який час, тому зараз важливо мати технічну підтримку 24/7. (5 балів)

3. Обов'язковий функціонал – до цього критерію відносяться такі функції сайту як додавання товару в корзину, видалення товару, замовлення товару, та інші. Також будемо оцінювати якість реалізації. (5 балів)

Максимальна оцінка 15 балів.

Аналогічне програмне забезпечення:

Comfy:Зручний і інтуїтивний сайт з приємним інтерфейсом. Має швидку технічну підтримку.

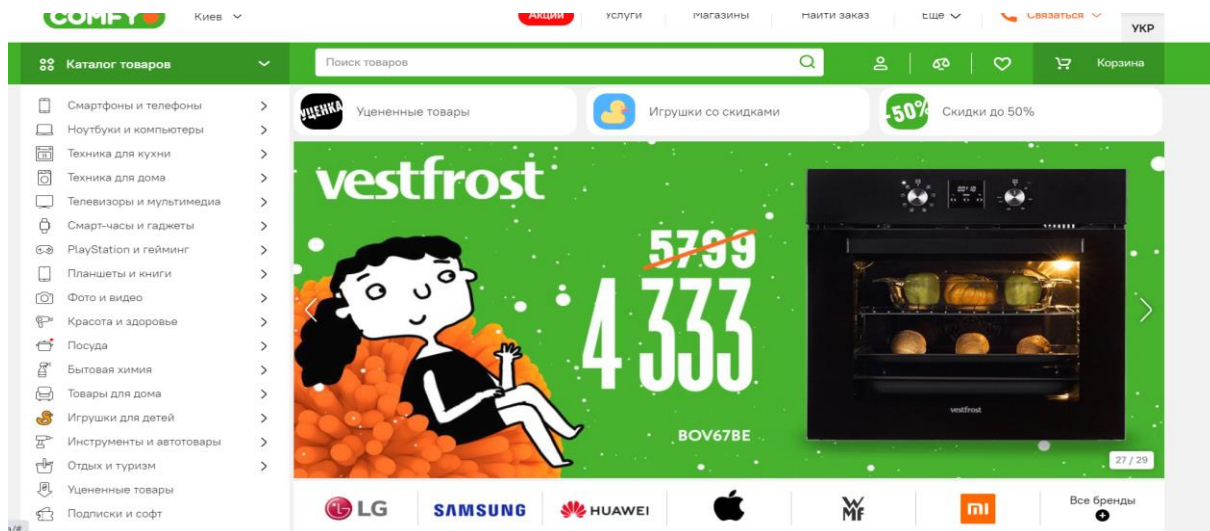


Рис. 1.2.1 – Comfy

Foxtrot: об'єднує в собі величезну кількість інструментів. Замовлення товарів та доставка, корзина та технічна підтримка - тут є все, з чим нам доводиться мати справу щодня.

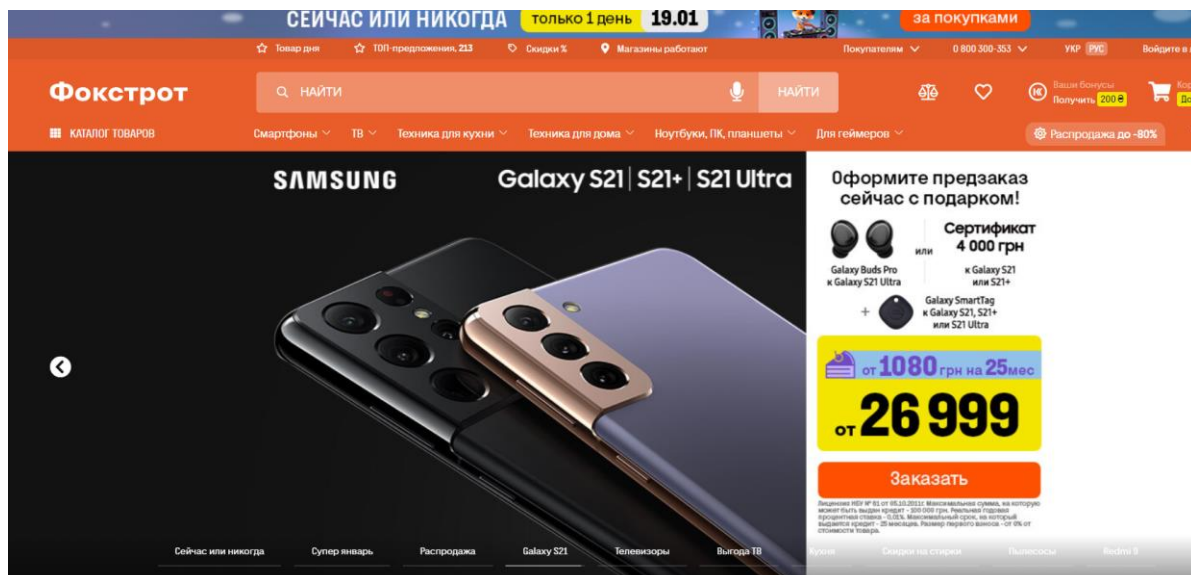


Рис. 1.2.2 – Foxtrot

Eldorado Невеликий, але зручний інструмент доступний і у вигляді мобільних додатків.

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 12   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

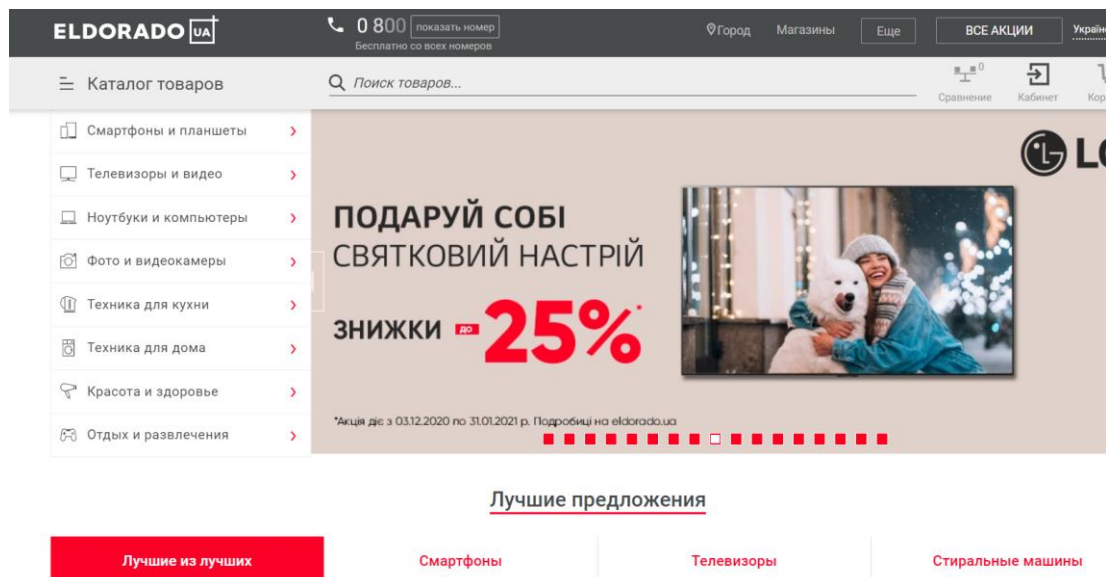


Рис. 1.2.3 – Eldorado

Таблиця 1.2.1

| Критерій                | Comfy     | Foxtrot   | Eldorado  |
|-------------------------|-----------|-----------|-----------|
| Технічна підтримка      | 5         | 3         | 4         |
| Обов'язковий функціонал | 5         | 4         | 4         |
| UI/UX                   | 4         | 3         | 3         |
| <b>Всього</b>           | <b>14</b> | <b>10</b> | <b>11</b> |

Отже, проаналізувавши інтернет-магазина ми можемо зробити висновки, що розроблюване програмне забезпечення має мати наступні критерії:

- UI/UX;
- Оперативна технічну підтримку;
- Мати обов'язковий функціонал;

### 1.3. Архітектура та узагальнена структура, майбутнього програмного комплексу

Основою системи є архітектура паттерна MVC [\[1\]](#) (Model-View-Controller). MVC (Модель-Вид-Контролер) є архітектурним паттерном, який використовується для розділення логіки програми на три складові: модель, вид і контролер.

Модель відповідає за доступ до даних та їх обробку. Вона включає логіку доступу до бази даних або інших джерел даних. Модель виконує запити до джерел даних, обробляє отримані дані і зберігає їх для подальшого використання в додатку. Крім того, модель може здійснювати обробку і валідацію даних перед збереженням.

Перегляд відповідає за візуалізацію даних і взаємодію з користувачем. Його завдання - відображення даних з моделі у зручному для користувача форматі. Перегляд генерує веб-сторінки, які користувач може бачити і з взаємодіяти.

Контролер відповідає за обробку запитів користувача та керування взаємодією між моделлю та переглядом. Він отримує запити від користувача, виконує необхідні дії в моделі, отримує результати і передає їх до перегляду для відображення. Контролер також може відповідати за маршрутизацію запитів і керування станом додатку.

Використання архітектури MVC дозволяє ефективно організувати розробку веб-додатків. Він дозволяє розділити логіку програми на окремі компоненти, що спрощує розподіл завдань між розробниками і полегшує підтримку та розширення додатку. Кожен компонент - модель, вид і контролер - може бути змінений або замінений без впливу на решту системи.

Сьогоднішнім днем фреймворки на основі паттерна MVC, такі як Laravel, CodeIgniter та Yii, стали дуже популярними в розробці веб-додатків. Вони надають розробникам готові компоненти, інструменти та різноманітні функції, що спрощують роботу з архітектурою MVC і прискорюють процес розробки. Застосування архітектури MVC в інтернет-магазині може мати численні переваги. Наприклад, модель може забезпечувати доступ до бази даних з продуктами, замовленнями та користувачами, а також виконувати операції, пов'язані з цими даними, такі як додавання, редагування та видалення. Перегляд може відповідати за відображення списку продуктів, деталей конкретного продукту, кошика покупок та інших сторінок інтернет-магазину. Контролер може обробляти запити користувача, які стосуються додавання товару до кошика, оформлення замовлення, авторизації користувача тощо.

Загалом, використання архітектури MVC у розробці інтернет-магазину дозволяє забезпечити чітку організацію коду, полегшує підтримку і розширення функціональності та забезпечує гнучкість і масштабованість додатку. Завдяки цьому підходу розробка інтернет-магазину може бути більш ефективною та продуктивною.

На даному етапі розробки розглянемо загальний алгоритм його роботи. Загальна схема роботи інтернет-магазину приведена на рисунку 1.3.1

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 15   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |



#### 1.4. Інструментальні засоби реалізації програмного продукту та вимоги до апаратних засобів.

Сайт буде розроблений з використанням мов HTML/CSS, PHP та JavaScript, і використовуватиме власну систему управління вмістом (CMS). Для зберігання інформації про користувачів буде використовуватись база даних SQL на сервері Windows. Для взаємодії з базами даних [8] будуть використовуватись SQL-запити, які дозволяють вибирати необхідні поля з бази даних інтернет-магазину. Для управління базами даних буде використовуватись технологія phpMyAdmin.

Робота з базами даних є поширеною практикою у сучасних додатках, незалежно від того, чи це настільний додаток, веб-сайт або веб-служба. В платформі .NET Framework існує багато способів взаємодії з базами даних з використанням керованого коду, які описуються на нашому веб-сайті. Однією з основних систем управління базами даних для Windows є SQL Server, а ми також надамо опис адміністрування цієї системи.

Окрім того, важливою складовою роботи інтернет-магазину є безпека. Забезпечення захисту даних користувачів та їх конфіденційності є надзвичайно важливою задачею. У цьому контексті використовуються різноманітні заходи безпеки, такі як шифрування передачі даних, застосування автентифікації та авторизації, контроль доступу до ресурсів та перевірка на вразливості.

Крім того, для забезпечення зручності користувачів та покращення їх досвіду використання, можуть бути впроваджені додаткові функціональні можливості. Наприклад, можливість додавання товарів до списку бажань, огляди та рейтинги товарів, фільтрація за різними параметрами, рекомендації товарів на основі попередніх покупок та багато іншого.

Необхідно також враховувати адаптивність інтернет-магазину до різних пристроїв і розмірів екранів, щоб користувачі могли зручно здійснювати покупки як на комп'ютері, так і на мобільних пристроях. Розробка

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 17   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

респонсивного дизайну та оптимізація швидкості завантаження сторінок є важливими аспектами цього процесу.

Усі ці функціональності і аспекти розробки інтернет-магазину потребують уважного планування, проектування та реалізації. Застосування паттерну MVC допомагає розділити логіку програми на логічно пов'язані компоненти, що полегшує розвиток, супровід та модифікацію системи в майбутньому.

У результаті вдалої реалізації інтернет-магазину з використанням MVC та відповідних технологій, користувачі зможуть зручно здійснювати покупки та замовлення необхідних товарів, отримуючи задоволення від високоякісного сервісу та безпеки своїх даних.

Вимоги до апаратних засобів для інтернет-магазину можуть включати:

1. Великий обсяг оперативної пам'яті та процесорної потужності для обробки великої кількості запитів і даних.
2. Відповідний обсяг сховища даних для збереження інформації про товари, користувачів і замовлення.
3. Надійність і захищеність системи: високоякісні компоненти та постійна моніторинг для забезпечення надійності і безпеки системи
4. Мережеві з'єднання: швидке і надійне мережеве з'єднання для забезпечення безперебійної роботи інтернет-магазину і забезпечення безпечності даних.
5. Резервне копіювання: для забезпечення можливості відновлення даних у випадку виникнення непередбачуваних обставин.

Розберемо інструментальні засоби їх по черзі:

### 1.4.1. MVC

MVC [2] (Модель-Вид-Контролер) - це архітектурний паттерн, який широко використовується при розробці програмного забезпечення, зокрема веб-додатків. Цей паттерн дозволяє розділити логіку програми на три основні компоненти: модель, вид і контролер.

Інтернет-магазин розроблений з використанням стійкої та структурованої системи даних, заснованої на паттерні MVC (Модель-Перегляд-Контролер). Основна мета сайту - забезпечити користувачам можливість здійснювати покупки та робити замовлення на потрібні їм товари.

### 1.4.2. HTML

HTML (HyperText Markup Language) - це мова розмітки гіпертексту, яка використовується для створення веб-сторінок. HTML є основою для побудови веб-документів та визначення їх структури і вигляду. За допомогою HTML можна визначати різноманітні елементи, такі як заголовки, абзаци, списки, таблиці, зображення, посилання та багато інших. Основні принципи HTML полягають у використанні тегів для обрамлення вмісту сторінки. Теги вказують браузеру, як слід інтерпретувати і відображати вміст

### 1.4.3. CSS

CSS [3] (Cascading Style Sheets) - це мова стилізації, яка використовується для визначення зовнішнього вигляду веб-сторінок, написаних з використанням HTML або інших мов розмітки. CSS дозволяє встановлювати різноманітні стилі, такі як кольори, шрифти, розміри, відступи, фонові зображення та багато інших параметрів, для елементів на веб-сторінці. Основна ідея CSS полягає в розділенні стилів від структури сторінки, що дозволяє зберігати стильові правила окремо

від розмітки. Це спрощує управління та зміну зовнішнього вигляду веб-сторінок, оскільки можна змінювати стилі на всій сторінці шляхом зміни одного CSS файлу, не змінюючи HTML-структуру. CSS працює на основі селекторів, які вказують, на які елементи сторінки будуть застосовуватися певні стилі

#### 1.4.4. PHP

PHP [\[4\]](#) (Hypertext Preprocessor) - це скриптова мова програмування загального призначення, яка використовується для розробки веб-додатків та динамічних веб-сторінок. Вона є однією з найпопулярніших мов програмування для веб-розробки та має велику спільноту розробників. Основна особливість PHP полягає в тому, що вона виконується на стороні сервера. Це означає, що PHP-код виконується на веб-сервері, а результати його виконання передаються на браузер користувача у вигляді звичайного HTML. Такий підхід дозволяє генерувати динамічний вміст на стороні сервера, включаючи звернення до бази даних, обробку форм, створення сесій та багато іншого. PHP має велику кількість вбудованих функцій і розширень, що спрощують розробку веб-додатків. Вона підтримує роботу з різними базами даних, включаючи MySQL, PostgreSQL, SQLite та інші. Також PHP підтримує роботу з різними протоколами та сервісами, такими як HTTP, FTP, XML, JSON і багато інших. PHP можна використовувати як самостійну мову програмування або в поєднанні з HTML та CSS для розробки веб-сторінок та додатків. Завдяки своїй простоті та гнучкості PHP використовується для створення різноманітних веб-проектів, від невеликих особистих сайтів до великих корпоративних веб-сайтів та електронних комерційних платформ.

#### 1.4.5. JavaScript

JavaScript - це скриптова мова програмування, яка використовується для розробки веб-додатків та динамічного взаємодії з користувачем на веб-сторінках.

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 20   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

Вона є однією з найпоширеніших мов програмування в інтернеті і виконується безпосередньо в браузері. JavaScript дає можливість створювати інтерактивні елементи на веб-сторінках, змінювати їх вміст, стиль та поведінку. Вона використовується для валідації форм, анімації, маніпуляції з даними на стороні клієнта, асинхронного завантаження даних (AJAX) та багатьох інших задач. JavaScript є мовою з подієвим драйвером, що означає, що вона реагує на події, такі як клік мишею, натискання клавіші або завантаження сторінки. За допомогою JavaScript можна створювати функції, які викликаються при виникненні певних подій, що дозволяє забезпечувати взаємодію з користувачем. JavaScript є мовою, яка підтримується всіма сучасними браузерами. Вона має велику кількість вбудованих функцій та методів для маніпуляції з HTML-елементами, роботи зі стилями, зміни DOM-структури, обробки подій та багатьох інших завдань.

#### 1.4.6. CMS

CMS (Система управління контентом) - це програмне забезпечення, яке використовується для створення, керування та публікації веб-контенту. Вона дозволяє користувачам без спеціальних навичок веб-розробки створювати та редагувати вміст своїх веб-сайтів з використанням зручного інтерфейсу. Основна перевага CMS полягає в тому, що вона розділяє веб-дизайн від вмісту, що дозволяє змінювати дизайн сайту, не впливаючи на вміст. CMS надає різноманітні функції, такі як керування структурою сайту, додавання та редагування сторінок, управління мультимедійним вмістом, керування користувачами та доступом, пошукову оптимізацію та багато іншого. CMS може бути використана для розробки різних типів веб-сайтів, від особистих блогів до корпоративних порталів та електронних магазинів. Деякі популярні CMS включають WordPress, Joomla, Drupal, TYPO3 та багато інших. Кожна з них має свої унікальні особливості та набір функцій, що задовольняють різні потреби

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          |      |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          | 21   |

користувачів. Використання CMS спрощує процес створення та управління веб-сайтом, оскільки вона надає готові рішення та інструменти для швидкого розгортання та налаштування сайту. Крім того, CMS забезпечує безпеку веб-сайту, оновлення системи та додатків, а також підтримку спільноти розробників, що дозволяє отримувати підтримку та розширювати функціональність сайту. Загалом, CMS є важливим інструментом для розробників та власників веб-сайтів, оскільки вона дозволяє ефективно керувати контентом та публікувати вміст без необхідності в глибоких знаннях програмування та веб-розробки.

#### 1.4.7. SQL

SQL [\[5\]](#) (Structured Query Language) - це мова запитів структурованої вибірки даних, яка використовується для управління та обробки реляційних баз даних. SQL дозволяє користувачам виконувати різноманітні операції з базою даних, такі як створення таблиць, вибірка даних, вставка, оновлення та видалення записів. Основна мета SQL - це взаємодія з базами даних для зберігання, організації та отримання інформації. SQL використовується в багатьох реляційних базах даних, таких як MySQL, PostgreSQL, Oracle, SQL Server та інші. Вона має стандартний набір команд і функцій, які дозволяють виконувати різні завдання з обробки даних. За допомогою SQL можна створювати таблиці зі специфікацією полів, визначати зв'язки між таблицями, виконувати складні запити для вибірки даних з однієї чи декількох таблиць, сортувати, фільтрувати та групувати дані, створювати нові таблиці на основі результатів запитів, а також здійснювати інші операції для керування базою даних. SQL є потужним інструментом для роботи з базами даних, оскільки вона дозволяє ефективно організовувати та обробляти великі обсяги даних. Вона забезпечує можливість зберігання, оновлення та отримання інформації з бази даних з використанням простого та зрозумілого синтаксису. SQL є стандартом в

галузі управління базами даних і є необхідним інструментом для багатьох розробників та аналітиків даних.

#### 1.4.8. phpMyAdmin

PhpMyAdmin [\[6\]](#) є веб-додатком з відкритим кодом, написаним на мові PHP. Він надає зручний веб-інтерфейс для адміністрування СУБД MySQL. За допомогою phpMyAdmin можна адмініструвати сервер MySQL, виконувати SQL-запити та переглядати вміст таблиць та баз даних. Цей додаток здобув велику популярність серед веб-розробників, оскільки дозволяє керувати СУБД MySQL без прямого введення SQL-команд. PhpMyAdmin поширюється під ліцензією GNU General Public License і використовується в багатьох розробках, таких як XAMPP, Denwer, AppServ, Open Server.

### Висновки до першого розділу

В першому розділі було розглянуто особливості веб-сайтів та визначено їх мету та основні характеристики. Дослідження показало, що веб-сайти є важливим інструментом для комунікації та представлення інформації в Інтернеті. Аналізуючи їх особливості, можна виокремити такі аспекти, як дизайн, навігація, контент і функціональність. Розуміння цих особливостей допомагає розробникам створювати ефективні веб-сайти, які задовольняють потреби користувачів та досягають своїх цілей. Розділ дозволяє отримати загальне уявлення про веб-сайти та становить важливу основу для подальших досліджень.

## РОЗДІЛ 2. ПРОЕКТУВАННЯ ТА РОЗРОБКА ІНТЕРНЕТ-МАГАЗИНУ.

### 2.1. Опис майбутнього програмного комплексу

Головна сторінка - представляє собою головну вітальну сторінку сайту, на якій містяться основні категорії товарів, останні новинки та акції.

Каталог товарів - містить список товарів, які продаються на сайті, з можливістю фільтрації товарів за категоріями, брендами та ціною.

Сторінка товару - представляє собою детальну інформацію про конкретний товар, включаючи зображення, опис, характеристики та відгуки покупців.

Кошик - сторінка, на якій користувач може переглядати товари, які він вибрав для покупки, редагувати кількість товарів та видаляти їх з кошика.

Сторінка оформлення замовлення - де користувач може ввести свої контактні дані, вибрати спосіб доставки.

Сторінка мої замовлення - де користувач може переглянути історію своїх замовлень та слідкувати за їх статусом.

Сторінка контактів - де користувач може знайти інформацію про контакти магазину, наприклад, адресу, номер телефону, електронну адресу та іншу контактну інформацію.

Сторінка адміністрування - де адміністратор може керувати сайтом.

Сторінка політики конфіденційності та умов надання послуг – де користувач може дізнатися про правила та умови, які діють на сайті.

## 2.2. Варіанти використання системи

Діаграма прецедентів [7] (варіантів використання) – це потужний інструмент у мові моделювання UML, який допомагає описати поведінку системи. Вона дозволяє візуалізувати взаємодію між акторами і варіантами використання, що допомагає розуміти, як система функціонує та які функції вона забезпечує. Варіанти використання визначають набір дій, послуг і функцій, які система повинна здійснювати для задоволення потреб акторів. Вони відображають специфічні сценарії, які можуть відбутися при взаємодії користувачів або зовнішніх систем з системою. У контексті діаграм прецедентів, "система" може означати будь-який об'єкт або програмне забезпечення, яке розробляється або використовується. "Актори" представляють людей, групи або організації, які виконують роль або взаємодіють з системою. У стандартні види відносин між акторами і подіями входять:

- Асоціації (association) вказують на зв'язок між акторами і варіантами використання.
- Розширення (extend) дозволяє описати додаткові можливості, які можуть бути активовані в певному варіанті використання.
- Узагальнення (generalization) використовується для показу спільних характеристик або спадкування між різними варіантами використання. Використання діаграм прецедентів спрощує аналіз, проектування та розробку системи, дозволяючи краще зрозуміти її функціональність та взаємодію з акторами. Вона допомагає забезпечити високу якість та взаєморозуміння між учасниками проекту, що сприяє успішному впровадженню системи.

Діаграма прецедентів розробленого застосунку зображена на рисунку 2.2.1.

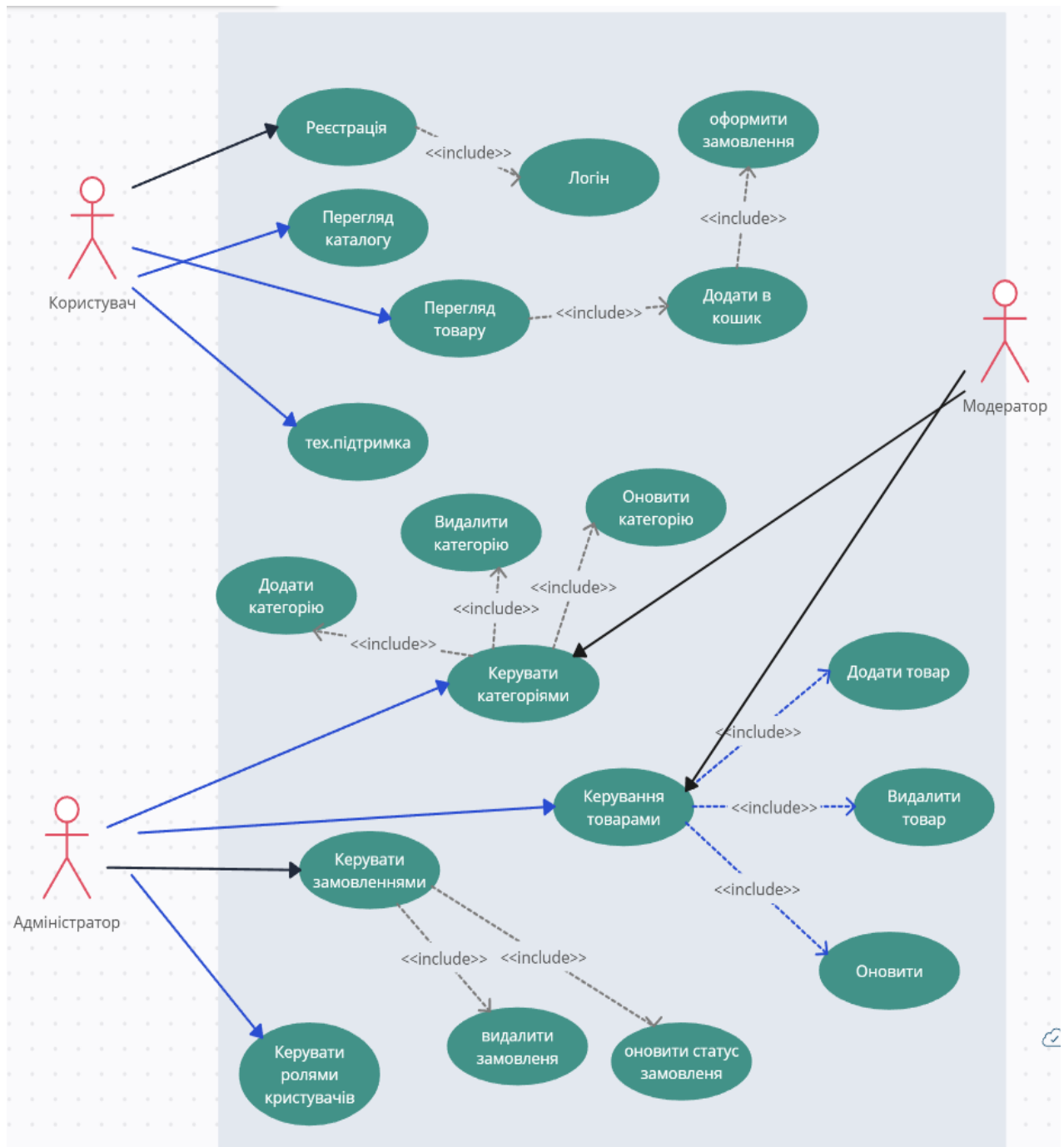


Рис.2.2.1 - Діаграма прецедентів системи

## 2.3. Структура системи збереження даних та алгоритми роботи системи

Для ефективного зберігання та керування даними в інтернет-магазині використовується потужна структурована реляційна база даних, така як MySQL. Ця база даних включає в себе набір таблиць, які представляють різні сутності

системи, такі як товари, категорії, підкатегорії, користувачі, замовлення та інші. Процес роботи інтернет-магазину включає кілька етапів, які сприяють зручному та ефективному процесу покупки для користувачів:

- Користувач проводить пошук потрібного товару в каталозі сайту за допомогою вбудованої функції пошуку або шляхом навігації по категоріях.

- Після знаходження бажаного товару, користувач додає його до кошика, вказуючи необхідну кількість.
- Користувач переходить до процесу оформлення замовлення, де він вводить дані про доставку, обирає спосіб оплати та підтверджує замовлення.
- Після цього система перевіряє наявність товару на складі і підтверджує замовлення, генеруючи унікальний номер замовлення.
- Користувач отримує підтвердження замовлення, де вказані деталі замовлення та інструкції щодо оплати.
- Після здійснення оплати замовлення користувачем, система перевіряє отримані платежі та підтверджує оплату.
- Замовлення підтверджується та надсилається на обробку та відправку.

Користувач отримує повідомлення про статус замовлення та очікуваний час доставки. Цей процес забезпечує зручну та надійну покупку для користувачів і дозволяє інтернет-магазину ефективно

База даних інтернет-магазину складається з п'яти таблиць:

1. Таблиця "Категорії" (category) - описує категорії товарів:

- id (ідентифікатор)
- name (назва)
- sort\_order (порядок сортування)
- status (статус)

2. Таблиця "Підкатегорії" (subcategory) - описує підкатегорії товарів:

- id (ідентифікатор)
- name (назва)
- sort\_order (порядок сортування)
- category\_id (ідентифікатор батьківської категорії)
- status (статус)

3. Таблиця "Замовлення" (product\_order) - містить інформацію про замовлення:

- user\_name (ім'я користувача)
- user\_email (електронна пошта користувача)
- user\_phone (телефон користувача)
- user\_comment (коментар користувача)
- user\_id (ідентифікатор користувача)
- date (дата замовлення)
- products (інформація про товари)
- status (статус замовлення)

4. Таблиця "Товари" (product) - містить інформацію про товари:

- id (ідентифікатор)
- name (назва)
- code (код товару)
- brand (бренд)
- price (ціна)
- title (заголовок)
- description (опис)
- category\_id (ідентифікатор категорії)
- sub\_category\_id (ідентифікатор підкатегорії)
- availability (наявність)

- is\_featured (виділений)
- is\_new (новий)
- image (зображення)
- status (статус)

5. Таблиця "Користувачі" (user) - описує користувачів системи:

- id (ідентифікатор)
- first\_name (ім'я)
- last\_name (прізвище)
- email (електронна пошта)
- password (пароль)
- birth (дата народження)
- company (компанія)
- address (адреса)
- city (місто)
- state (область)
- postcode (поштовий індекс)
- country (країна)
- additional\_info (додаткова інформація)
- phone (телефон)
- date (дата реєстрації)"
- status (статус)
- role (роль)

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 29   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

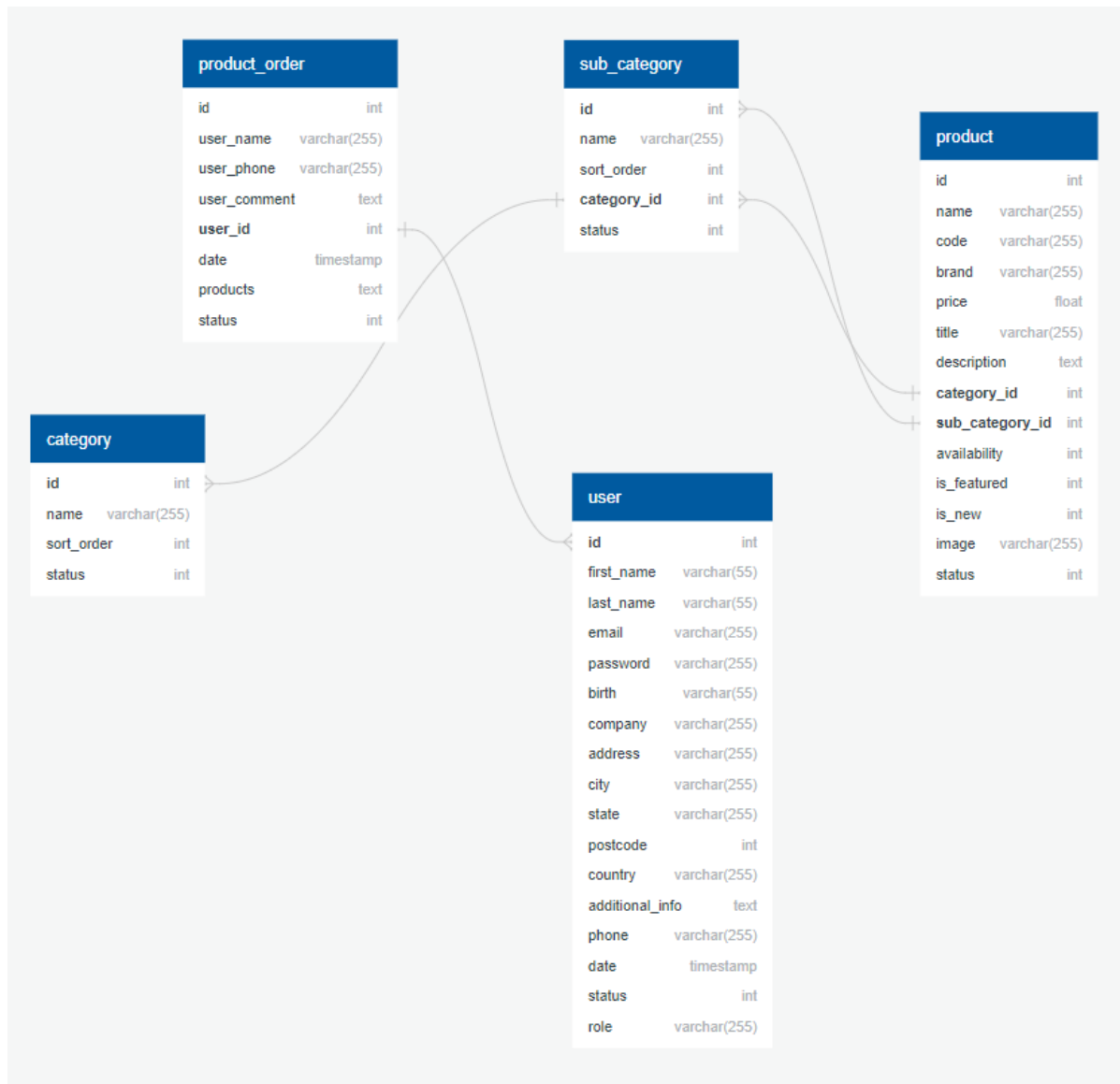


Рис. 2.3.1 – Схема бази даних системи

## 2.4. Структуру модулів та класів системи.

У шаблоні проектування MVC (Model-View-Controller), модель - це компонент, що відповідає за логіку додатку та взаємодію з базою даних або зовнішніми сервісами. Він має на увазі модель даних та бізнес-логіку, які визначають, які дані зберігаються та як вони обробляються. Модель може містити класи та функції, які виконують операції з даними, наприклад, отримання, оновлення, видалення та створення нових записів. Він також може

містити методи для проведення розрахунків, перевірки даних, валідації, обробки помилок тощо. Окрім того, модель може включати логіку для взаємодії з іншими модулями в рамках додатку, відправки та отримання повідомлень, подій або викликів функцій. Модель використовується в контролері для взаємодії з моделлю та представленням. Контролер викликає методи модуля для отримання або зміни даних, а потім передає ці дані представленню для відображення результатів.

Розберемо їх по черзі.

#### 2.4.1. Category Module

Category Module - модель, що відповідає за категорії та підкатегорії.

Даний код представляє модель "Category" для роботи з категоріями та підкатегоріями продуктів у веб-додатку. Основна функція цієї моделі - це взаємодія з базою даних для створення, зміни, видалення та отримання категорій та підкатегорій. Повний код в додатку А.

Він складається з таких методів:

Метод "createCategory": Цей метод дозволяє створити нову категорію продуктів. Він отримує параметри: \$name (назва категорії), \$sortOrder (порядковий номер) і \$status (статус, включено або виключено). Метод виконує підключення до бази даних, створює SQL-запит для додавання запису в таблицю "category" і виконує його. Якщо запит виконується успішно, повертається ідентифікатор останнього вставленого запису, в іншому випадку повертається 0.

Метод "createSubCategory": Цей метод дозволяє створити нову підкатегорію продуктів. Він отримує параметри: \$name (назва підкатегорії), \$sortOrder (порядковий номер), \$categoryId (ідентифікатор батьківської категорії) і \$status (статус, включено або виключено). Метод виконує підключення до бази даних, створює SQL-запит для додавання запису в таблицю "sub\_category" і виконує його. Якщо запит виконується успішно,

повертається ідентифікатор останнього вставленого запису, в іншому випадку повертається 0.

Метод "getCategoryById": Цей метод повертає категорію зазначеним ідентифікатором. Він отримує параметр \$id (ідентифікатор категорії), виконує підключення до бази даних, створює SQL-запит для вибірки даних з таблиці "category" за заданим ідентифікатором і виконує його. Результатом є асоціативний масив з інформацією про категорію.

Метод "getSubCategoryById(\$id)": Цей метод повертає підкатегорію з вказаним ідентифікатором. Він встановлює з'єднання з базою даних, створює SQL-запит для вибірки даних з таблиці "sub\_category" за вказаним ідентифікатором, встановлює значення параметра ":id" і виконує запит. Результатом є асоціативний масив з інформацією про підкатегорію.

Метод "getSubCategories(\$id)": Цей метод повертає список підкатегорій з вказаним ідентифікатором. Він встановлює з'єднання з базою даних, створює SQL-запит для вибірки даних з таблиці "sub\_category" за вказаним ідентифікатором, встановлює значення параметра ":id" і виконує запит. Результатом є масив асоціативних масивів з інформацією про підкатегорії.

Метод "updateCategoryById(\$id, \$name, \$sortOrder, \$status)": Цей метод оновлює категорію з вказаним ідентифікатором. Він встановлює з'єднання з базою даних, створює SQL-запит для оновлення запису в таблиці "category" з вказаним ідентифікатором, встановлює значення параметрів ":id", ":name", ":sortOrder" і ":status" і виконує запит. Результатом є значення true або false в залежності від успішності виконання запиту.

Метод "updateSubCategoryById(\$id, \$name, \$sortOrder, \$categoryId, \$status)": Цей метод оновлює підкатегорію з вказаним ідентифікатором. Він встановлює з'єднання з базою даних, створює SQL-запит для оновлення запису в таблиці "sub\_category" з вказаним ідентифікатором, встановлює значення параметрів ":id", ":name", ":sortOrder", ":categoryId" і ":status" і виконує запит.

Результатом є значення true або false в залежності від успішності виконання запити.

Метод "deleteCategoryById(\$id)": Цей метод видаляє категорію з вказаним ідентифікатором. Він встановлює з'єднання з базою даних, створює SQL-запит для видалення запису з таблиці "category" за вказаним ідентифікатором, встановлює значення параметра ":id" і виконує запит. Результатом є значення true або false в залежності від успішності виконання запити.

Метод "deleteSubCategoryById(\$id)": Цей метод видаляє підкатегорію з вказаним ідентифікатором. Він встановлює з'єднання з базою даних, створює SQL-запит для видалення запису з таблиці "sub\_category" за вказаним ідентифікатором, встановлює значення параметра ":id" і виконує запит. Результатом є значення true або false в залежності від успішності виконання запити.

Метод "getCategoriesList()": Цей метод повертає список категорій для відображення на сайті. Він встановлює з'єднання з базою даних, створює SQL-запит для вибірки даних з таблиці "category" з умовою "status = 1" (увімкнено) і впорядковує їх за значенням "sort\_order". Результатом є масив асоціативних масивів з інформацією про категорії.

Метод "getSubCategoriesList(\$category\_id)": Цей метод повертає список підкатегорій для вказаної категорії. Він встановлює з'єднання з базою даних, створює SQL-запит для вибірки даних з таблиці "sub\_category" з умовами "status = 1" (увімкнено) і "category\_id = :category\_id" (вказана категорія), встановлює значення параметрів ":status" і ":category\_id" і виконує запит. Результатом є масив асоціативних масивів з інформацією про підкатегорії.

Метод "getSubCategoryName(\$category\_id, \$id)": Цей метод повертає назву підкатегорії за вказаними ідентифікаторами категорії і підкатегорії. Він встановлює з'єднання з базою даних, створює SQL-запит для вибірки назви з таблиці "sub\_category" з умовами "status = 1" (увімкнено), "category\_id =

:category\_id" (вказана категорія) і "id = :id" (вказана підкатегорія), встановлює значення параметрів ":status", ":category\_id" і ":id" і виконує запит. Результатом є назва підкатегорії або false, якщо підкатегорія не знайдена.

Метод "getCategoriesListAdmin()": Цей метод повертає список категорій для відображення в панелі адміністратора. Він встановлює з'єднання з базою даних, створює SQL-запит для вибірки даних з таблиці "category" і впорядковує їх за значенням "sort\_order". Результатом є масив асоціативних масивів з інформацією про категорії.

Метод "getSubCategoriesListAdmin(\$category\_id = 1)": Цей метод повертає список підкатегорій для вказаної категорії в панелі адміністратора. Він встановлює з'єднання з базою даних, створює SQL-запит для вибірки даних з таблиці "sub\_category" з умовою "category\_id = :category\_id" (вказана категорія) і впорядковує їх за значенням "sort\_order". Результатом є масив асоціативних масивів з інформацією про підкатегорії.

## 2.4.2. Product Module

Product Module - модель, що відповідає за керування інформацією про товари.

Цей клас Product містить набір статичних методів для роботи з продуктами в базі даних. Давайте розглянемо кожну функцію по черзі:

Повний код в додатку Б.

Метод getLatestProducts(\$count = self::SHOW\_BY\_DEFAULT): Цей метод повертає останні продукти з бази даних. Він приймає необов'язковий параметр \$count, який визначає кількість продуктів, що повертаються. За замовчуванням він використовує константу SHOW\_BY\_DEFAULT, яка встановлена на значення 6. Метод виконує запит до бази даних і повертає масив з результатами.

Метод getFeaturedProducts(): Цей метод повертає пропозиції продуктів з бази даних. Він виконує запит до бази даних, щоб вибрати продукти, які є пропозиціями. Метод повертає масив з результатами.

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          |      |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          | 34   |

Метод `getProductsListByCategory($categoryId, $page = 1)`: Цей метод повертає список продуктів за заданою категорією. Він приймає параметр `$categoryId`, що визначає ідентифікатор категорії продуктів, і необов'язковий параметр `$page`, який визначає номер сторінки. Метод виконує запит до бази даних і повертає масив з результатами.

Метод `getProductsListBySubCategory($categoryId, $subCategoryId, $page = 1)`: Цей метод повертає список продуктів за заданою підкатегорією. Він приймає параметри `$categoryId` і `$subCategoryId`, що визначають ідентифікатори категорії і підкатегорії продуктів, і необов'язковий параметр `$page`, який визначає номер сторінки. Метод виконує запит до бази даних і повертає масив з результатами.

Метод `getProductById($id)`: Цей метод повертає інформацію про продукт за заданим ідентифікатором. Він приймає параметр `$id`, що визначає ідентифікатор продукту, і виконує запит до бази даних для отримання даних про продукт.

Метод `getTotalProductsInCategory($categoryId)`: Цей метод повертає загальну кількість.

### 2.4.3. Order Module

Order Module - модель, що відповідає за керування інформацією про замовлення. Основні функції цього класу включають збереження, оновлення, видалення та отримання замовлень з бази даних. Повний код в додатку В.

Ось опис кожного методу:

Метод `save($userName, $userEmail, $userPhone, $userComment, $userId, $products)`: Цей метод зберігає нове замовлення в базі даних. Приймає параметри, такі як `$userName` (ім'я користувача), `$userEmail` (електронна пошта користувача), `$userPhone` (телефон користувача), `$userComment` (коментар користувача), `$userId` (ідентифікатор користувача) і `$products` (список продуктів у форматі JSON). Повертає результат виконання запиту до бази даних.

Метод `updateOrderById($id, $options)`: Цей метод оновлює існуюче замовлення в базі даних. Приймає параметри, такі як `$id` (ідентифікатор замовлення) і `$options` (масив зі зміненими параметрами замовлення, наприклад, ім'я, електронна пошта, телефон тощо). Повертає нічого (`void`).

Метод `getOrdersList()`: Цей метод отримує список всіх замовлень з бази даних. Повертає масив з інформацією про кожне замовлення, таку як ідентифікатор, ім'я користувача, електронна пошта, дата і статус.

Метод `getOrderListById($userId)`: Цей метод отримує список замовлень за ідентифікатором користувача. Приймає параметр `$userId` (ідентифікатор користувача). Повертає масив з інформацією про кожне замовлення.

Метод `getOrderById($id)`: Цей метод отримує інформацію про замовлення за його ідентифікатором. Приймає параметр `$id` (ідентифікатор замовлення). Повертає масив з інформацією про замовлення.

Метод `deleteOrderById($id)`: Цей метод видаляє замовлення за його ідентифікатором. Приймає параметр `$id` (ідентифікатор замовлення). Повертає нічого (`void`).

Метод `getQuantityInOrder($id)`: Цей метод отримує кількість продуктів у замовленні за його ідентифікатором. Приймає параметр `$id` (ідентифікатор замовлення). Повертає масив з інформацією про продукти та їх кількість у замовленні.

Метод `getStatusText($status)`: Цей метод повертає текстовий опис статусу замовлення на основі числового значення статусу. Приймає параметр `$status` (числове значення статусу). Повертає текстовий опис статусу.

#### 2.4.4. User Module

Цей код представляє клас `User`, який містить різні методи для роботи з користувачами. Весь код в додатку Г.Давайте розглянемо кожен метод окремо:

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          |      |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          | 36   |

Метод register(\$firstName, \$lastName, \$email, \$password, \$birth, \$company, \$address, \$city, \$state, \$postcode, \$country, \$info, \$phone): Цей метод виконує реєстрацію нового користувача. Він отримує різні параметри, такі як ім'я, прізвище, електронна пошта, пароль, дата народження, компанія, адреса, місто, область, поштовий індекс, країна, додаткова інформація і телефон. Потім виконується запит до бази даних для вставки нового користувача.

Метод update(\$id, \$firstName, \$lastName, \$email, \$password, \$birth, \$company, \$address, \$city, \$state, \$postcode, \$country, \$info, \$phone): Цей метод оновлює існуючого користувача з вказаним ідентифікатором. Він приймає параметри, які включають ідентифікатор користувача і різні дані користувача. Потім виконується запит до бази даних для оновлення відповідного запису користувача.

Метод checkUserData(\$email, \$password): Цей метод перевіряє, чи існує користувач з вказаними електронною поштою та паролем. Він виконує запит до бази даних і перевіряє, чи є відповідний запис користувача з вказаними даними. Якщо такий запис існує, повертається ідентифікатор користувача, інакше повертається false.

Метод auth(\$userId): Цей метод виконує аутентифікацію користувача. Він записує іде checkLogged(): Цей метод перевіряє, чи існує активна сесія для користувача. Якщо сесія існує, повертається ідентифікатор користувача, в іншому випадку користувач перенаправляється на сторінку входу (/login).

Метод isGuest(): Цей метод перевіряє, чи є користувач гостем (неавторизованим). Якщо існує активна сесія, повертається false, в іншому випадку - true.

Метод getUserName(): Цей метод отримує ім'я користувача за його ідентифікатором. Якщо існує активна сесія, отримується ідентифікатор користувача, потім виконується запит до бази даних для отримання даних

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          |      |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          | 37   |

користувача (ім'я, прізвище). Ім'я користувача повертається, якщо користувач аутентифікований, в іншому випадку повертається рядок "guest".

Метод `checkFirstName($firstName)`: Цей метод перевіряє, чи має вказане ім'я користувача необхідну довжину (не менше 2 символів). Якщо довжина імені відповідає умові, повертається `true`, в іншому випадку - `false`.

Метод `checkLastName($lastName)`: Цей метод перевіряє, чи має вказане прізвище користувача необхідну довжину (не менше 2 символів). Якщо довжина прізвища відповідає умові, повертається `true`, в іншому випадку - `false`.

Метод `checkEmail($email)`: Цей метод перевіряє, чи вказана електронна пошта коректна за допомогою функції `filter_var` і фільтра `FILTER_VALIDATE_EMAIL`. Якщо електронна пошта відповідає умові, повертається `true`, в іншому випадку - `false`.

Метод `checkEmailExists($email)`: Цей метод перевіряє, чи існує вказана електронна пошта в базі даних. Він виконує запит до бази даних для підрахунку кількості записів з вказаною електронною поштою. Якщо кількість записів більше нуля, повертається `true`, в іншому випадку - `false`.

Метод `checkPassword($password)`: Цей метод перевіряє, чи має вказане паролі необхідну довжину (не менше 6 символів). Якщо довжина пароля відповідає умові, повертається `true`, в іншому випадку - `false`.

Метод `checkPhone($phone)`: Цей метод перевіряє, чи має вказаний номер телефону необхідну довжину (не менше 10 символів). Якщо довжина номеру телефону відповідає умові, повертається `true`, в іншому випадку - `false`.

Метод `getUserById($id)`: Цей метод отримує дані користувача за його ідентифікатором з бази даних. Виконується запит до бази даних для отримання даних користувача за його ідентифікатором. Результат повертається у вигляді асоціативного масиву.

Метод `getUserByEmail($email)`: Цей метод отримує дані користувача за його електронною поштою з бази даних. Виконується запит до бази даних для отримання даних користувача за його електронною поштою. Результат повертається у вигляді асоціативного масиву.

Метод `resetPassword($email)`: Цей метод скидає пароль користувача, замінюючи його на випадково згенерований. Виконується запит до бази даних для оновлення паролю користувача. Якщо оновлення пройшло успішно, повертається новий пароль.

Метод `getUserList()`: Цей метод отримує список всіх користувачів з бази даних. Виконується запит до бази даних для отримання списку користувачів. Результат повертається у вигляді масиву, де кожен елемент масиву є асоціативним масивом з даними одного користувача.

Метод `deleteUserById($id)`: Цей метод видаляє користувача з бази даних за його ідентифікатором. Виконується запит до бази даних для видалення запису користувача за його ідентифікатором. Якщо видалення пройшло успішно, повертається `true`, в іншому випадку - `false`.

Метод `saveRole($role)`: Цей метод зберігає нову роль користувача в базі даних. Виконується запит до бази даних для вставки нового запису ролі користувача. Якщо вставка пройшла успішно, повертається `true`, в іншому випадку - `false`.

Метод `updateUserRoleById($id, $options)`: Цей метод оновлює роль користувача за його ідентифікатором. Виконується запит до бази даних для оновлення ролі користувача. Якщо оновлення пройшло успішно, повертається `true`, в іншому випадку - `false`.

Метод `updateUserById($id, $options)`: Цей метод оновлює дані користувача за його ідентифікатором. Виконується запит до бази даних для оновлення даних

користувача. Якщо оновлення пройшло успішно, повертається true, в іншому випадку - false.

## 2.5. Контролери системи.

Ось опис кожного з контролерів:

- AdminCategoryController.php: Цей контролер відповідає за адміністрування категорій товарів. Він містить методи для відображення списку категорій, створення нової категорії, редагування і видалення існуючих категорій.
- AdminController.php: Цей контролер відповідає за адміністрування загальних аспектів системи. Він містить методи для управління користувачами, ролями користувачів, статистикою, налаштуваннями системи та іншими адміністративними функціями.
- AdminOrderController.php: Цей контролер відповідає за адміністрування замовлень. Він містить методи для перегляду замовлень, зміни їх статусу, відміни замовлень та інших дій, пов'язаних з управлінням замовленнями.
- AdminProductController.php: Цей контролер відповідає за адміністрування товарів. Він містить методи для відображення списку товарів, створення нового товару, редагування та видалення існуючих товарів, а також керування характеристиками товарів.
- AdminRoleController.php: Цей контролер відповідає за адміністрування ролей користувачів. Він містить методи для відображення списку ролей, створення нової ролі, редагування та видалення існуючих ролей.
- AdminSubCategoryController.php: Цей контролер відповідає за адміністрування підкатегорій товарів. Він містить методи для відображення списку підкатегорій, створення нової підкатегорії, редагування та видалення існуючих підкатегорій.

- **CabinetController.php:** Цей контролер відповідає за кабінет користувача. Він містить методи для відображення особистої інформації користувача, зміни пароля, перегляду історії замовлень та інших функцій, пов'язаних з кабінетом користувача.
- **CartController.php:** Цей контролер відповідає за роботу з кошиком покупок. Він містить методи для додавання товарів до кошика, видалення товарів з кошика, оформлення замовлення та інших функцій, пов'язаних з управлінням кошиком покупок.
- **CatalogController.php:** Цей контролер відповідає за відображення каталогу товарів. Він містить методи для відображення списку товарів, фільтрації товарів за певними параметрами, перегляду деталей товару та інших функцій, пов'язаних з каталогом товарів.
- **ModerController.php:** Цей контролер відповідає за модерацію вмісту. Він містить методи для перегляду та затвердження/відхилення вмісту, поданих користувачами, таких як коментарі, відгуки, зображення тощо.
- **ProductController.php:** Цей контролер відповідає за відображення окремого товару. Він містить методи для відображення деталей конкретного товару, додавання товару до кошика, додавання відгуку, оцінки товару та інших функцій, пов'язаних з окремим товаром.
- **SiteController.php:** Цей контролер відповідає за основні сторінки сайту, такі як головна сторінка, сторінка контактів, сторінка "Про нас" та інші статичні сторінки. Він містить методи для відображення цих сторінок та обробки форм, які можуть бути пов'язані з цими сторінками.
- **UserController.php:** Цей контролер відповідає за реєстрацію та аутентифікацію користувачів. Він містить методи для відображення форми реєстрації та входу, обробки цих форм, виходу з системи та інших функцій, пов'язаних з управлінням користувачами.

Ці контролери представляють різні аспекти функціональності системи і забезпечують взаємодію між користувачем та програмним забезпеченням. Кожен з них відповідає за виконання певних завдань та обробку відповідних запитів від користувачів.

### Висновки до другого розділу

У системі програмного забезпечення інтернет-магазину були описані різні класи та модулі, які відповідають за різні аспекти функціональності магазину. Класи контролерів відповідають за обробку різних запитів та взаємодію з іншими модулями системи. Класи бази даних забезпечують збереження та доступ до даних, а класи користувачів, товарів, кошика та замовлень реалізують відповідні функції і операції, пов'язані з цими сутностями.

Завдяки такій структурі та класифікації функціональності системи, програмне забезпечення інтернет-магазину демонструє високу організованість та простоту розширення. Кожна модель та клас має чітку відповідальність і може бути незалежно модифікований або розширений без впливу на решту системи.

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 42   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

## РОЗДІЛ 3. ОПИС РОБОТИ З ПРОГРАМНИМ ДОДАТКОМ ТА ЙОГО ТЕСТУВАННЯ МОДУЛІВ ТА КЛАСІВ СИСТЕМИ

### 3.1. Опис роботи з програмним додатком

На даному етапі розробки кваліфікаційної роботи веб-додаток інтернет-магазину вже реалізований і потребує пояснень щодо його використання та проведення тестувань. Опис роботи з додатком складається з ілюстрацій та текстових пояснень, які покривають основні аспекти інтерфейсу та функціоналу, доступного під час користування. Опис інтернет-магазину електроніки включає декілька окремих вікон, для кожного з яких надається детальне пояснення. Завдяки цьому користувачі матимуть змогу зрозуміти, як користуватися різними елементами інтерфейсу та виконувати необхідні дії.

#### 3.1.1. Реєстрація користувачів

Додаток надає можливість реєстрації нових користувачів, які отримують унікальний ідентифікатор та пароль для входу. Це забезпечує безпеку та обмежує доступ до особистої інформації.

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 43   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

Ваша особиста інформація

Ім'я \*

First Name

Прізвище \*

Last Name

Email \*

Email

Пароль \*

Password

Дата народження \*

дд. мм. рррр

Your address

Компанія

company

Адреса \*

Address

Місто \*

city

Область \*

State

Поштовий індекс \*

Zip / Postal Code

Країна \*

Country

Додаткова інформація

Додаткова інформація

Мобільний телефон \*

phone

Необхідно зареєструвати хоча б один номер телефону

\*Обов'язкове поле

Регістрація

Рис.3.1.1 Вікно реєстрації користувачів

### 3.1.2. Авторизація користувачів

Зареєстровані користувачі мають можливість авторизуватися в додатку, використовуючи свій ідентифікатор та пароль. Це дозволяє користувачам отримати доступ до свого особистого облікового запису та здійснювати покупки.

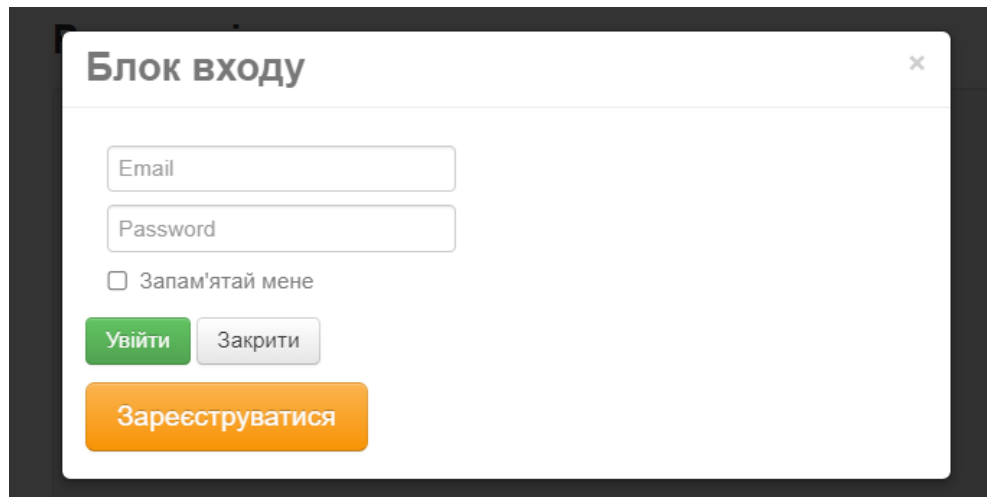


Рис.3.1.2 Вікно авторизації користувачів

### 3.1.3. Пошук та перегляд товарів

Додаток надає можливість користувачам шукати товари за назвою. Сортувати товари за ціною, назвою. Результати пошуку відображаються у зручному списку, де користувачі можуть переглядати деталі товарів, зображення, опис та ціни.

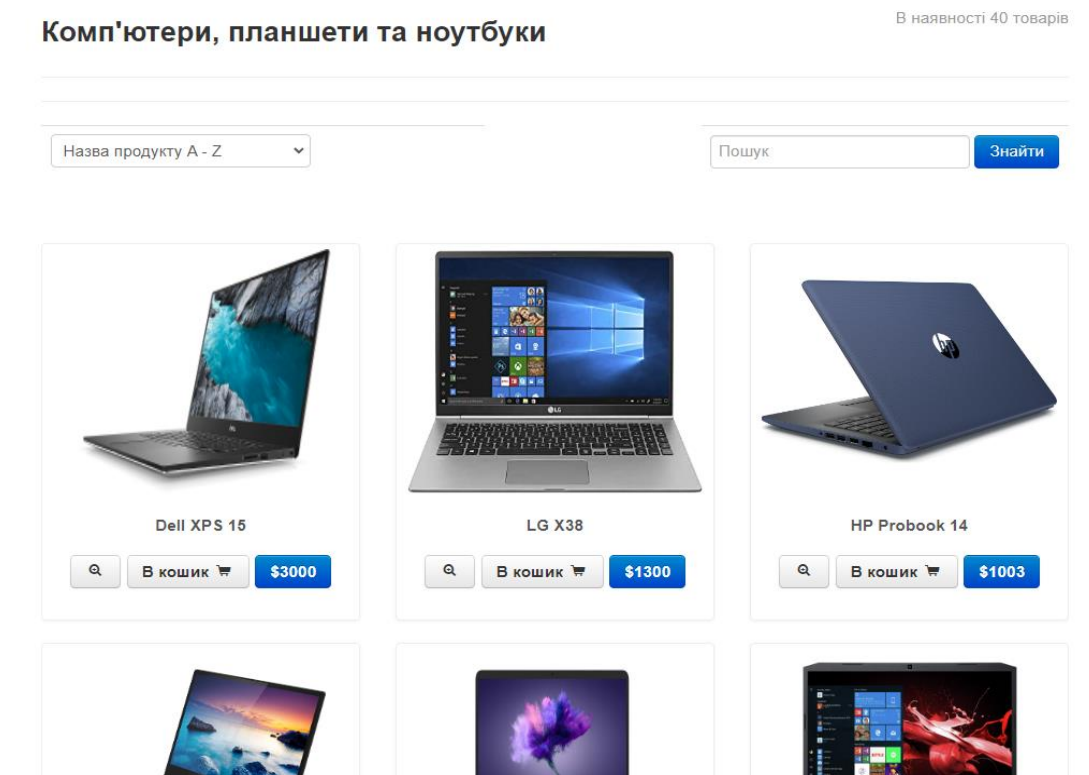


Рис.3.1.3.1 Вікно перегляду товарів

Комп'ютери, планшети та ноутбуки

В наявності 40 товарів

Назва продукту A - Z

LG

Знайти



LG X38

В кошик

\$1300

Рис.3.1.3.2 Вікно пошуку товарів

Комп'ютери, планшети та ноутбуки

В наявності 40 товарів

Назва продукту A - Z

Пошук

Знайти



Dell XPS 15

В кошик

\$3000



Honor R48

В кошик

\$3000



Acer Ri38

В кошик

\$3000

Рис.3.1.3.3 Вікно сортування товарів

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 46   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |



## Dell XPS 15

\$3000

Додати

### Інформація про Продукт

| Продукт деталі |              |
|----------------|--------------|
| Назва:         | Dell XPS 15  |
| Ціна:          | 3000\$       |
| Бренд:         | Dell         |
| Опис:          | Company Dell |

Рис.3.1.3.4 Вікно перегляду деталей товару

### 3.1.4. Додавання товарів до кошика

Користувачі можуть додавати бажані товари до свого кошика покупок. Кількість товарів можна змінювати або видаляти їх з кошика

### Кошик [ 4 Товар(ів) ]

← Продовжуйте покупки

| Продукт        | Назва         | Опис         | Кількість/Оновлення | Ціна    | Усього |
|----------------|---------------|--------------|---------------------|---------|--------|
|                | Dell XPS 15   | Company Dell | 3                   | \$3000  | \$9000 |
|                | HP Probook 14 | Company HP   | 1                   | \$1003  | \$1003 |
| ЗАГАЛЬНА СУМА: |               |              |                     | \$10003 |        |

← Продовжити покупки

→ Далі

Рис.3.1.4.1 Вікно додавання товарів в кошик

### 3.1.5. Оформлення замовлення

Користувачі можуть додавати бажані товари до свого кошика покупок. Кількість товарів можна змінювати або видаляти їх з кошика. Це дає змогу користувачам зберігати обрані товари перед оформленням замовлення

Home / Кошик

Кошик [ 4 Товар(ів) ]

ДЛЯ ОФОРМЛЕННЯ ЗАМОВЛЕННЯ ЗАПОВНІТЬ ФОРМУ НИЖЧЕ, НАШ МЕНЕДЖЕР З ВАМ ЗВ'ЯЖЕТЬСЯ

Ім'я

Name

Email

Email

Телефон

Phone

Коментар

Відправити

← Кошик

Рис.3.1.5.1 Вікно замовлення товарів

Home / Кошик

Кошик [ 0 Товар(ів) ]

Замовлення відправлено

← Кошик

Рис.3.1.5.2 Вікно замовлення товарів завершено

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 48   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

### 3.1.6. Особистий кабінет

Особистий кабінет забезпечує зручний спосіб для користувачів управляти своїми особистими даними та отримувати інформацію про їхні замовлення. Він дозволяє забезпечити актуальність і точність інформації про користувача та забезпечує зручний доступ до історії замовлень для більшої контролю та зручності.

## Cabinet

### Ваш Кабінет

 [ 0 ] Товари у вашому кошику

Ласкаво просимо, **admin**

Ваша роль, **admin**

Керуйте продуктами

Керуйте категоріями

Керуйте замовленнями

Керуйте користувача

Переглянути історію замовлень

Редагувати особисту інформацію

Рис.3.1.6.1 Вікно особистий кабінету

### 3.1.7. Переглянути історію замовлень

У особистому кабінеті користувачі можуть переглядати історію своїх замовлень. Це включає інформацію про дату замовлення, статус доставки та деталі кожного замовлення. Користувачі можуть легко відстежувати свої попередні покупки та мати доступ до докладної інформації про кожен окремий замовлення.

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 49   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

#### Список замовлень

Ласкаво просимо, *admin*

#### Ваші замовлення

| Ім'я  | Пошта           | Дата                | Статус          |             |
|-------|-----------------|---------------------|-----------------|-------------|
| admin | admin@admin.web | 2023-05-16 14:40:14 | Нове замовлення | Переглянути |
| admin | admin@admin.web | 2023-05-16 22:33:36 | Доставлено      | Переглянути |
| admin | admin@admin.web | 2023-05-28 16:24:30 | Нове замовлення | Переглянути |

Рис.3.1.7.1 Вікно списку замовлень

| Ім'я  | Пошта           | Телефон     | Коментар | Дата                | Статус     |
|-------|-----------------|-------------|----------|---------------------|------------|
| admin | admin@admin.web | 09795145693 | товари 3 | 2023-05-16 22:33:36 | Доставлено |

#### Деталі замовлення:

| Назва          | Код    | Ціна | Кількість |         |
|----------------|--------|------|-----------|---------|
| Midrange D2    | 34242  | 4000 | 1         |         |
| Lumix T13      | 115543 | 5000 | 1         |         |
| Rosa Ki 10     | 237415 | 4300 | 1         |         |
| ЗАГАЛЬНА СУМА: |        |      |           | \$13300 |

#### Товари:

| Name        | фото                                                                                | Price |
|-------------|-------------------------------------------------------------------------------------|-------|
| Midrange D2 |  | 4000  |

Рис.3.1.7.2 Вікно перегляду замовлення

### 3.1.8. Зв'язок з тех.підтримкою

Додаток надає можливість звернутися до технічної підтримки через онлайн-чат або електронну пошту. Користувачі можуть отримати допомогу, задати питання або повідомити про проблеми, які вони зустріли

---

**Email Us**

Імя

email

Прізвище

Рис.3.1.8.1 Вікно тех.підтримки

### 3.1.9. Адміністрація та керування сайтом

Як адміністратор, у вас є повний доступ до адміністративної панелі веб-додатка. Це надає вам можливість керувати категоріями товарів, замовленнями та користувачами.

Основні можливості адміністратора включають:

1. Керування категоріями: Ви можете створювати, редагувати та видаляти категорії товарів. Це дозволяє вам організовувати товари в зручні категорії для користувачів і полегшує навігацію в додатку.

Панель адміністратора

Домашня сторінка

Панель адміністратора Керування категоріями

Додати категорію

Додати підкатегорію

Список категорій

| ID категорії:               | Ім'я                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | Порядок сортування          | Статус             |               |                    |          |  |  |    |                   |   |   |   |            |          |   |                                  |   |   |   |            |          |   |             |   |   |   |            |          |   |                   |   |   |   |            |          |  |  |  |  |
|-----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------|--------------------|---------------|--------------------|----------|--|--|----|-------------------|---|---|---|------------|----------|---|----------------------------------|---|---|---|------------|----------|---|-------------|---|---|---|------------|----------|---|-------------------|---|---|---|------------|----------|--|--|--|--|
| 1                           | ЕЛЕКТРОНІКА                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    | 1                           | 1                  | Edit          | Delete             |          |  |  |    |                   |   |   |   |            |          |   |                                  |   |   |   |            |          |   |             |   |   |   |            |          |   |                   |   |   |   |            |          |  |  |  |  |
|                             | <table> <tr> <th>Ідентифікатор підкатегорії:</th> <th>Ім'я</th> <th>Категорія ID:</th> <th>Порядок сортування</th> <th>Статус</th> <th></th> <th></th> </tr> <tr> <td>1</td> <td>Мобільні телефони</td> <td>1</td> <td>1</td> <td>1</td> <td>Редагувати</td> <td>Видалити</td> </tr> <tr> <td>2</td> <td>Комп'ютери, планшети та ноутбуки</td> <td>1</td> <td>1</td> <td>1</td> <td>Редагувати</td> <td>Видалити</td> </tr> <tr> <td>3</td> <td>Фотоапарати</td> <td>1</td> <td>1</td> <td>1</td> <td>Редагувати</td> <td>Видалити</td> </tr> <tr> <td>4</td> <td>Звук і зображення</td> <td>1</td> <td>1</td> <td>1</td> <td>Редагувати</td> <td>Видалити</td> </tr> </table> | Ідентифікатор підкатегорії: | Ім'я               | Категорія ID: | Порядок сортування | Статус   |  |  | 1  | Мобільні телефони | 1 | 1 | 1 | Редагувати | Видалити | 2 | Комп'ютери, планшети та ноутбуки | 1 | 1 | 1 | Редагувати | Видалити | 3 | Фотоапарати | 1 | 1 | 1 | Редагувати | Видалити | 4 | Звук і зображення | 1 | 1 | 1 | Редагувати | Видалити |  |  |  |  |
| Ідентифікатор підкатегорії: | Ім'я                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | Категорія ID:               | Порядок сортування | Статус        |                    |          |  |  |    |                   |   |   |   |            |          |   |                                  |   |   |   |            |          |   |             |   |   |   |            |          |   |                   |   |   |   |            |          |  |  |  |  |
| 1                           | Мобільні телефони                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | 1                           | 1                  | 1             | Редагувати         | Видалити |  |  |    |                   |   |   |   |            |          |   |                                  |   |   |   |            |          |   |             |   |   |   |            |          |   |                   |   |   |   |            |          |  |  |  |  |
| 2                           | Комп'ютери, планшети та ноутбуки                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               | 1                           | 1                  | 1             | Редагувати         | Видалити |  |  |    |                   |   |   |   |            |          |   |                                  |   |   |   |            |          |   |             |   |   |   |            |          |   |                   |   |   |   |            |          |  |  |  |  |
| 3                           | Фотоапарати                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    | 1                           | 1                  | 1             | Редагувати         | Видалити |  |  |    |                   |   |   |   |            |          |   |                                  |   |   |   |            |          |   |             |   |   |   |            |          |   |                   |   |   |   |            |          |  |  |  |  |
| 4                           | Звук і зображення                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | 1                           | 1                  | 1             | Редагувати         | Видалити |  |  |    |                   |   |   |   |            |          |   |                                  |   |   |   |            |          |   |             |   |   |   |            |          |   |                   |   |   |   |            |          |  |  |  |  |
| 2                           | Картини                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | 2                           | 1                  | Edit          | Delete             |          |  |  |    |                   |   |   |   |            |          |   |                                  |   |   |   |            |          |   |             |   |   |   |            |          |   |                   |   |   |   |            |          |  |  |  |  |
|                             | <table> <tr> <th>Ідентифікатор підкатегорії:</th> <th>Ім'я</th> <th>Категорія ID:</th> <th>Порядок сортування</th> <th>Статус</th> <th></th> <th></th> </tr> <tr> <td>30</td> <td>asd</td> <td>2</td> <td>8</td> <td>1</td> <td>Редагувати</td> <td>Видалити</td> </tr> </table>                                                                                                                                                                                                                                                                                                                                                                                               | Ідентифікатор підкатегорії: | Ім'я               | Категорія ID: | Порядок сортування | Статус   |  |  | 30 | asd               | 2 | 8 | 1 | Редагувати | Видалити |   |                                  |   |   |   |            |          |   |             |   |   |   |            |          |   |                   |   |   |   |            |          |  |  |  |  |
| Ідентифікатор підкатегорії: | Ім'я                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | Категорія ID:               | Порядок сортування | Статус        |                    |          |  |  |    |                   |   |   |   |            |          |   |                                  |   |   |   |            |          |   |             |   |   |   |            |          |   |                   |   |   |   |            |          |  |  |  |  |
| 30                          | asd                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | 2                           | 8                  | 1             | Редагувати         | Видалити |  |  |    |                   |   |   |   |            |          |   |                                  |   |   |   |            |          |   |             |   |   |   |            |          |   |                   |   |   |   |            |          |  |  |  |  |

Рис.3.1.9.1 Вікно керування категоріями

Панель адміністратора Керування категоріями

Редагувати категорії

Відредагуйте категорію

Name \*

ЕЛЕКТРОНІКА

Sort Order \*

1

Status

Showed

Save

Рис.3.1.9.2 Вікно редагування категорії

2. Керування товарами: Ви можете створювати, редагувати та видаляти товари. Це дозволяє вам перевіряти якість і відповідність товарів вимогам вашого магазину перед їхнім показом для користувачів.

Панель адміністратора

Домашня сторінка

Панель адміністратора Керуйте продуктами

Додати товар

Список продуктів

| Ідентифікатор продукту: | Код    | Ім'я          | Ціна  |            |          |
|-------------------------|--------|---------------|-------|------------|----------|
| 1                       | 34561  | Iphone 7      | 500   | Редагувати | Видалити |
| 2                       | 23411  | Iphone S8     | 700   | Редагувати | Видалити |
| 3                       | 178375 | LG            | 565   | Редагувати | Видалити |
| 4                       | 28341  | Rasper        | 1000  | Редагувати | Видалити |
| 6                       | 32424  | Fly A38       | 180   | Редагувати | Видалити |
| 7                       | 23841  | Sony S1       | 494   | Редагувати | Видалити |
| 8                       | 24324  | Dell XPS 15   | 3000  | Редагувати | Видалити |
| 9                       | 23432  | LG X38        | 1300  | Редагувати | Видалити |
| 10                      | 3848   | HP Probook 14 | 1003  | Редагувати | Видалити |
| 11                      | 3844   | Lenovo SP2    | 800   | Редагувати | Видалити |
| 12                      | 37472  | Honor R48     | 3000  | Редагувати | Видалити |
| 13                      | 388421 | Acer RI38     | 3000  | Редагувати | Видалити |
| 14                      | 324235 | Stream Q5     | 10000 | Редагувати | Видалити |
| 15                      | 34234  | Canon A3      | 5000  | Редагувати | Видалити |

Рис.3.1.9.3 Вікно керування товарами

### Редагувати товар

Ім'я \*

Код \*

Ціна \*

Категорія \*

Subcategory \*

Brand

Зображення



Файл не вибрано

Опис

Рис.3.1.9.4 Вікно редагування та додавання товару

3. Керування замовленнями: Ви маєте доступ до списку всіх замовлень, зроблених користувачами. Ви можете переглядати деталі замовлення, оновлювати їх статуси та забезпечувати виконання замовлень з відповідними вимогами та термінами.

## Список замовлень

| ID замовлення: | Ім'я   | Пошта                | Дата                | Статус          |             |            |          |
|----------------|--------|----------------------|---------------------|-----------------|-------------|------------|----------|
| 10             | admin  | admin@admin.web      | 2023-05-16 14:40:14 | Нове замовлення | Переглянути | Редагувати | Видалити |
| 11             | moder  | moderator2@gmail.com | 2023-05-16 20:37:55 | Нове замовлення | Переглянути | Редагувати | Видалити |
| 12             | admin  | admin@admin.web      | 2023-05-16 22:33:36 | Доставлено      | Переглянути | Редагувати | Видалити |
| 13             | admin  | admin@admin.web      | 2023-05-28 16:24:30 | Нове замовлення | Переглянути | Редагувати | Видалити |
| 14             | Bohdan | admin@gmail.com      | 2023-05-30 00:05:11 | Нове замовлення | Переглянути | Редагувати | Видалити |

Рис.3.1.9.5 Вікно списку замовлень

## Редагувати замовлення

Ім'я користувача \*

Телефон користувача \*

Email

Коментар користувача \*

User ID:

Статус

## Products:

| ID Product: | Name    | Code | Price |
|-------------|---------|------|-------|
| 22          | Yeti R7 | 2384 | 2000  |

Save

Рис.3.1.9.6 Вікно редагування замовлення

4. Керування користувачами: Ви можете переглядати профілі користувачів, вносити зміни у їхні особисті дані та керувати їхніми правами доступу. Це дозволяє вам контролювати активність користувачів та забезпечувати безпеку вашого веб-додатка.

| Панель адміністратора Керуйте користувача |       |          |                      |            |                 |            |            |            |            |                 |         |                        |               |                     |        |          |            |          |
|-------------------------------------------|-------|----------|----------------------|------------|-----------------|------------|------------|------------|------------|-----------------|---------|------------------------|---------------|---------------------|--------|----------|------------|----------|
| Додати користувача                        |       |          |                      |            |                 |            |            |            |            |                 |         |                        |               |                     |        |          |            |          |
| Список користувачів                       |       |          |                      |            |                 |            |            |            |            |                 |         |                        |               |                     |        |          |            |          |
| Ідентифікатор                             | Ім'я  | Прізвище | Email                | Пароль     | Дата народження | Компанія   | Адреса     | Місто      | Область    | Поштовий індекс | Країна  | Додаткова інформація   | Телефон       | Дата                | Статус | Роль     |            |          |
| 1                                         | admin | Rocket   | admin@admin.web      | 123456     | 2000-01-08      | Rosy inc.  | street 23  | New York   | Arizona    | 2858285         | USA     | Additional information | 3777777777777 | 2020-05-14 12:05:42 | 1      | admin    | Редагувати | Видалити |
| 2                                         | Rop   | Loco     | loco@mga.com         | 123456     | 2008-01-12      | Cop        | street 282 | San Diego  | [*S]       | 18385           | [*I]    | Additional information | 123456789123  | 2020-05-14 15:38:02 | 1      | admin123 | Редагувати | Видалити |
| 3                                         | Test  | Test     | test@gmail.com       | 123456     | 2011-11-11      | Test       | test       | Test       | [*I]       | 1341516         | [*I]    | test                   | 12345678912   | 2020-05-14 17:49:46 | 1      | admin    | Редагувати | Видалити |
| 7                                         | admin | admin    | admin@gmail.com      | admin      | 2021-12-29      | admin      | admin      | admin      | admin      | 45455           | Україна | Додаткова інформація   | 0978514592    | 2023-01-23 19:26:11 | 1      | admin    | Редагувати | Видалити |
| 14                                        | moder | moder    | moderator2@gmail.com | moderator2 |                 | moderator2 | moderator2 | moderator2 | moderator2 | 0               | Україна | Додаткова інформація   | 0989878933    | 2023-05-13 21:08:31 | 1      | moder    | Редагувати | Видалити |

Рис.3.1.9.7 Вікно списку користувачів

Панель адміністратора

Панель адміністратора Керуйте Користувачами Редагувати замовлення

### Редагувати Користувача

Роль

admin

user

admin

moder

saveRole

Рис.3.1.9.8 Вікно редагування ролі

## 3.2. Тестування роботи програмного забезпечення

На сьогоднішній день фахівці виділяють наступні основні типи тестування для сайтів:

1. Функціональне тестування (Functionality testing)
2. Тестування зручності користування (Usability testing)
3. Тестування інтерфейсу користувача (UI testing)

4. Тестування продуктивності (Performance testing)
5. Тестування безпеки (Security testing)

Функціональне тестування (Functionality testing) - це тип тестування, який перевіряє, чи працюють всі функції програмного забезпечення згідно з вимогами та специфікаціями. Це незалежний від структури програмного забезпечення тест, який перевіряє різні аспекти, описані у вимогах та специфікаціях.

Функціональне тестування включає такі етапи:

1. Перевірка коректності роботи всіх посилань.
2. Перевірка коректності відображення користувацьких форм та повідомлень про помилки при некоректному заповненні.
3. Перевірка правильності коду HTML і CSS для полегшення індексації сайту пошуковими системами.
4. Тестування відповідності бізнес-процесів робочим процесам на сайті.
5. Перевірка реакції на негативні сценарії, такі як введення некоректних даних.

Тестування зручності користування (Usability testing) спрямоване на виявлення недоліків у взаємодії користувача з сайтом. Головні характеристики, що оцінюються, включають простоту навчання, навігацію, задоволення користувача та загальний вигляд.

Тестування інтерфейсу користувача (UI testing) включає перевірку графічного інтерфейсу, взаємодії елементів управління та відображення контенту на сайті. Метою цього тестування є переконатися, що інтерфейс користувача є зрозумілим, естетичним та функціональним.

Тестування продуктивності (Performance testing) відповідає на питання про швидкодію, масштабованість та стабільність роботи сайту при різних

навантаженнях. Воно включає тестування швидкості завантаження сторінок, відгуків на користувацькі запити та витривалості сайту при тривалій роботі.

Тестування безпеки (Security testing) перевіряє вразливості сайту на зловживання та несанкціонований доступ. Це включає перевірку захисту від атак типу SQL-ін'єкція, перехоплення сесії та інших потенційних загроз.

Всі ці види тестування важливі для забезпечення якості програмного забезпечення та покращення користувацького досвіду. Вони допомагають виявити помилки та проблеми перед розгортанням сайту та забезпечують його ефективну та безпечну роботу.

Під час проведення тестування програмного забезпечення інтернет-магазину було проведено ретельний аналіз та перевірка різних функціональних аспектів цієї системи. Метою тестування було переконатися в надійності та безперебійності роботи інтернет-магазину перед його випуском в експлуатацію.

У рамках тестування було протестовано наступні аспекти програмного забезпечення інтернет-магазину:

1. Реєстрація та авторизація користувачів: була перевірена можливість реєстрації нових користувачів та авторизації існуючих. Перевірено коректність збереження та відображення введених даних користувачів, а також забезпечення безпеки під час авторизації.
2. Пошук товарів: проведено пошук різних товарів за різними критеріями, такими як назва товару, категорія, ціна тощо. Перевірено правильність виводу результатів пошуку та відображення детальної інформації про товар.
3. Додавання товарів до кошика: перевірено можливість додавання товарів до кошика покупок, а також коректність розрахунку загальної суми замовлення.

4. Оформлення замовлення: проведено тестування процесу оформлення замовлення. Перевірено коректність обробки замовлення та його збереження у системі.
5. Оплата замовлення: протестовано процес оплати замовлення різними методами, такими як кредитна карта, електронний переказ тощо. Перевірено правильність обробки платежів та підтвердження їх успішного здійснення.

Після уважної перевірки цих функціональних аспектів програмного забезпечення інтернет-магазину, раді повідомити, що жодних помилок або недоліків не було виявлено. Весь функціонал працює коректно та відповідає вимогам, що дозволяє заявити про готовність системи до використання.

Тестування було проведено з урахуванням різних сценаріїв використання, а також на різних пристроях та веб-браузерах, що дозволило впевнитися в сумісності програмного забезпечення інтернет-магазину з різними середовищами.

Враховуючи результати тестування, можна стверджувати, що програмне забезпечення інтернет-магазину готове до експлуатації і має високий рівень надійності та безперебійності у своїй роботі.

### Висновки до третього розділу

У даному розділі було надано детальний опис роботи з програмним додатком інтернет-магазину. Розглянуто взаємодію з різними функціональними модулями та класами системи. Описано процеси реєстрації та авторизації користувачів, пошуку товарів, додавання їх до кошика покупок, оформлення замовлення та оплати.

Результати тестування були дуже задовільними. Під час виконання тестових сценаріїв не виявлено жодних помилок або недоліків у роботі системи.

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 59   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

Функціонал програмного забезпечення інтернет-магазину працює коректно та надійно, відповідаючи очікуванням користувачів.

Таким чином, на основі проведеного тестування можна зробити висновок, що програмне забезпечення інтернет-магазину готове до використання та має високий рівень якості та надійності. Розроблений інтерфейс та його функціонал були успішно випробуваний і демонструють свою працездатність.

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 60   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

## ВИСНОВКИ

В результаті виконання цієї роботи було проаналізовано аналоги інтернет магазинів, були оцінені та взято до уваги всі головні критерії, які мають бути у веб-додатку. Були визначені структурні елементи, які мають бути у додатку.

В процесі реалізації веб-додатка був розроблений головний функціонал з алгоритмами. Були сформовані основні вимоги та задачі, які стосуються самого веб-додатку. Було описано роботу з програмним додатком, включаючи взаємодію з різними функціональними модулями та класами системи. Було детально розкрито функціонал кожного модуля та його внутрішню структуру.

Завдяки чіткій структурі модулів та класів, програмне забезпечення інтернет-магазину виявилось легко розширюваним та підтримуваним. Кожна модель виконує свої завдання та взаємодіє з іншими частинами системи без зайвих проблем.

Були описана необхідність створення сайту, проведені певні аналізи, щоб дізнатися ресурсоемність ПЗ, її вимоги та описаний метод реалізації програми.

Для написання веб-додатку було використано мову HTML/CSS, JS/PHP та SQL Windows Server [\[9\]](#).

Під час тестування було перевірено роботу системи в різних сценаріях, включаючи реєстрацію та авторизацію користувачів, пошук товарів, додавання їх до кошика та оформлення замовлення. Результати тестування показали, що програмне забезпечення інтернет-магазину працює коректно та бездоганно. Помилки або недоліків у роботі системи не виявлено, що свідчить про високу якість та надійність програмного додатку.

Отже, можна зробити висновок, що програмне забезпечення інтернет-магазину готове до використання, має високу якість та надійність. Розроблений інтерфейс та його функціонал успішно пройшли тестування і відповідають вимогам і очікуванням користувачів.

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 61   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Патерн MVC [Електронний ресурс] // Highload. – 2022. – Режим доступу до ресурсу: <https://highload.today/uk/blogs/shho-take-mvc-ta-mvp-patterni/>.
2. MVC Framework [Електронний ресурс] // Tutorials Point. – 2021. – Режим доступу до ресурсу: [https://www.tutorialspoint.com/mvc\\_framework/mvc\\_framework\\_introduction.htm#:~:text=The%20Model%2DView%2DController%20\(development%20aspects%20of%20an%20application.](https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm#:~:text=The%20Model%2DView%2DController%20(development%20aspects%20of%20an%20application.)
3. CSS [Електронний ресурс] // css.in.ua. – 2020. – Режим доступу до ресурсу: <https://css.in.ua/>.
4. PHP [Електронний ресурс] // The PHP Group. – 2022. – Режим доступу до ресурсу: <https://www.php.net/manual/en/intro-what-is.php>.
5. SQL Tutorial [Електронний ресурс] // sqltutorial.org. – 2022. – Режим доступу до ресурсу: <https://www.sqltutorial.org/>.
6. phpMyAdmin [Електронний ресурс] // phpMyAdmin. – 2022. – Режим доступу до ресурсу: <https://www.phpmyadmin.net/>.
7. Діаграма прецедентів [Електронний ресурс] // ТОВ «На Урок». – 2020. – Режим доступу до ресурсу: <https://naurok.com.ua/prezentaciya-diagrami-uml-diagrami-precedentiv-238715.html>.
8. Работа с базами данных [Електронний ресурс] // ТОВ "Хостинг «Україна»". – 2022. – Режим доступу до ресурсу: <https://www.ukraine.com.ua/uk/blog/programming/sql-baza-dannih-dlya-chego-prednaznachena-baza-dannih.html>
9. Работа с базами данных [Електронний ресурс] // ТОВ "Хостинг «Україна»". – 2022. – Режим доступу до ресурсу: <https://www.ukraine.com.ua/uk/blog/>. [Електронний ресурс] // Microsoft. – 2023. – Режим доступу до ресурсу: <https://learn.microsoft.com/en-us/sql/relational-databases/sql-server-guides?view=sql-server-ver16>.

Додатки

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 63   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

```

class Category
{

    public static function createCategory($name, $sortOrder, $status)
    {
        // DB connection
        $db = Db::getConnection();

        // DB request
        $sql = 'INSERT INTO category (name, sort_order, status) '
            . 'VALUES (:name, :sort_order, :status)';

        // Getting and returning results. Prepared Request Used
        $result = $db->prepare($sql);
        $result->bindParam(":name", $name, PDO::PARAM_STR);
        $result->bindParam(":sort_order", $sortOrder, PDO::PARAM_STR);
        $result->bindParam(":status", $status, PDO::PARAM_STR);
        if ($result->execute()) {
            return $db->lastInsertId();
        }
        return 0;
    }

    public static function createSubCategory($name, $sortOrder, $categoryId, $status)
    {
        // DB connection
        $db = Db::getConnection();

        // DB request
        $sql = 'INSERT INTO sub_category (name, sort_order, category_id, status) '
            . 'VALUES (:name, :sort_order, :category_id, :status)';

        // Getting and returning results. Prepared Request Used
        $result = $db->prepare($sql);
        $result->bindParam(":name", $name, PDO::PARAM_STR);
        $result->bindParam(":sort_order", $sortOrder, PDO::PARAM_INT);
        $result->bindParam(":category_id", $categoryId, PDO::PARAM_INT);
        $result->bindParam(":status", $status, PDO::PARAM_INT);
        if ($result->execute()) {
            return $db->lastInsertId();
        }
        return 0;
    }
}

```

```

public static function getCategoryById($id)
{
    // DB connection
    $db = Db::getConnection();

    // DB request
    $sql = "SELECT * FROM category WHERE id = :id";

    // Prepared Request Used
    $result = $db->prepare($sql);
    $result->bindParam(":id", $id, PDO::PARAM_INT);

    // Indicate that we want to get data in the form of an array
    $result->setFetchMode(PDO::FETCH_ASSOC);

    // Execute the request
    $result->execute();

    // Return data
    return $result->fetch();
}

public static function getSubCategoryById($id)
{
    // DB connection
    $db = Db::getConnection();

    // DB request
    $sql = "SELECT * FROM sub_category WHERE id = :id";

    // Prepared Request Used
    $result = $db->prepare($sql);
    $result->bindParam(":id", $id, PDO::PARAM_INT);

    // Indicate that we want to get data in the form of an array
    $result->setFetchMode(PDO::FETCH_ASSOC);

    // Execute the request
    $result->execute();

    // Return data
    return $result->fetch();
}

public static function getSubCategories($id) {
    // DB connection
    $db = Db::getConnection();

    // DB request
    $sql = "SELECT * FROM sub_category WHERE id = :id";

```

```

// Prepared Request Used
$result = $db->prepare($sql);
$result->bindParam(":id", $id, PDO::PARAM_INT);

// Indicate that we want to get data in the form of an array
$result->setFetchMode(PDO::FETCH_ASSOC);

// Execute the request
$result->execute();

// Return data
return $result->fetchAll();
}

public static function updateCategoryById($id, $name, $sortOrder, $status)
{
    // DB connection
    $db = Db::getConnection();

    // DB request
    $sql = "UPDATE category SET name = :name, sort_order = :sortOrder, status = :status
"
        . "WHERE id = :id";

    // Getting and returning results. Prepared Request Used
    $result = $db->prepare($sql);
    $result->bindParam(":id", $id, PDO::PARAM_INT);
    $result->bindParam(":name", $name, PDO::PARAM_STR);
    $result->bindParam(":sortOrder", $sortOrder, PDO::PARAM_STR);
    $result->bindParam(":status", $status, PDO::PARAM_INT);
    return $result->execute();
}

public static function updateSubCategoryById($id, $name, $sortOrder, $categoryId,
$status)
{
    // DB connection
    $db = Db::getConnection();

    // DB request
    $sql = "UPDATE sub_category SET name = :name, sort_order = :sortOrder,
category_id = :category_id,"
        . " status = :status WHERE id = :id";

    // Getting and returning results. Prepared Request Used
    $result = $db->prepare($sql);
    $result->bindParam(":id", $id, PDO::PARAM_INT);
    $result->bindParam(":name", $name, PDO::PARAM_STR);
    $result->bindParam(":sortOrder", $sortOrder, PDO::PARAM_INT);
    $result->bindParam(":category_id", $categoryId, PDO::PARAM_INT);

```

```

        $result->bindParam(":status", $status, PDO::PARAM_INT);
        return $result->execute();
    }

    public static function deleteCategoryById($id)
    {
        // DB connection
        $db = Db::getConnection();

        // DB request
        $sql = 'DELETE FROM category WHERE id = :id';

        // Getting and returning results. Prepared Request Used
        $result = $db->prepare($sql);
        $result->bindParam(":id", $id, PDO::PARAM_INT);
        return $result->execute();
    }

    public static function deleteSubCategoryById($id)
    {
        // DB connection
        $db = Db::getConnection();

        // DB request
        $sql = 'DELETE FROM sub_category WHERE id = :id';

        // Getting and returning results. Prepared Request Used
        $result = $db->prepare($sql);
        $result->bindParam(":id", $id, PDO::PARAM_INT);
        return $result->execute();
    }

    public static function getCategoriesList()
    {
        // DB connection
        $db = Db::getConnection();

        // DB request
        $sql = 'SELECT id, name FROM category '
            . 'WHERE status="1" '
            . 'ORDER BY sort_order ASC';
        $result = $db->query($sql);

        // Getting and returning results
        $i = 0;
        $categoryList = array();
        while ($row = $result->fetch()) {
            $categoryList[$i]['id'] = $row['id'];
            $categoryList[$i]['name'] = $row['name'];
            $i++;
        }
    }

```

```

    }
    return $categoryList;
}

public static function getSubCategoriesList($category_id)
{
    $status = 1;
    // Create empty array for categories list
    $subCategoryList = array();
    // Connect to DB
    $db = Db::getConnection();

    // Make request string
    $sql = 'SELECT id, name, category_id FROM sub_category'
        . ' WHERE status=:status AND category_id=:category_id'
        . ' ORDER BY sort_order ASC';

    // Getting and returning results. Prepared Request Used
    $result = $db->prepare($sql);
    $result->bindParam(":status", $status, PDO::PARAM_INT);
    $result->bindParam(":category_id", $category_id, PDO::PARAM_INT);
    $result->setFetchMode(PDO::FETCH_ASSOC);
    $result->execute();

    // Get and return array
    $i = 0;
    while ($row = $result->fetch()) {
        $subCategoryList[$i]['id'] = $row['id'];
        $subCategoryList[$i]['name'] = $row['name'];
        $subCategoryList[$i]['category_id'] = $row['category_id'];
        $i++;
    }

    return $subCategoryList;
}

public static function getSubCategoryName($category_id, $id)
{
    $status = 1;
    // Connect to DB
    $db = Db::getConnection();

    // Make request string
    $sql = 'SELECT name FROM sub_category'
        . ' WHERE status=:status AND category_id=:category_id AND id=:id'
        . ' ORDER BY sort_order ASC';
    $result = $db->prepare($sql);
    $result->bindParam(":status", $status, PDO::PARAM_INT);
    $result->bindParam(":category_id", $category_id, PDO::PARAM_INT);
    $result->bindParam(":id", $id, PDO::PARAM_INT);

```

```

$result->setFetchMode(PDO::FETCH_ASSOC);
$result->execute();

// Get and return array
if ($row = $result->fetch()) {
    return $name = $row['name'];
}
return false;
}

public static function getCategoriesListAdmin()
{
    // DB connection
    $db = Db::getConnection();

    // DB request
    $sql = 'SELECT id, name, sort_order, status FROM category ORDER BY sort_order
ASC';
    $result = $db->query($sql);

    // Getting and returning results
    $categoryList = array();
    $i = 0;
    while ($row = $result->fetch()) {
        $categoryList[$i]['id'] = $row['id'];
        $categoryList[$i]['name'] = $row['name'];
        $categoryList[$i]['sort_order'] = $row['sort_order'];
        $categoryList[$i]['status'] = $row['status'];
        $i++;
    }
    return $categoryList;
}

public static function getSubCategoriesListAdmin($category_id = 1)
{
    $category_id = intval($category_id);

    if ($category_id) {
        // Create empty array for categories list
        $subCategoryList = array();
        // Connect to DB
        $db = Db::getConnection();

        // Make request string
        $sql = 'SELECT id, name, category_id, sort_order, status FROM sub_category '
            . 'WHERE category_id="'.$category_id.'"
            . 'ORDER BY sort_order ASC';
        $result = $db->query($sql);

        // Get and return array
        $i = 0;

```

```

while ($row = $result->fetch()) {
    $subCategoryList[$i]['id'] = $row['id'];
    $subCategoryList[$i]['name'] = $row['name'];
    $subCategoryList[$i]['category_id'] = $row['category_id'];
    $subCategoryList[$i]['sort_order'] = $row['sort_order'];
    $subCategoryList[$i]['status'] = $row['status'];
    $i++;
}
return $subCategoryList;
}
return false;
}

```

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 70   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

```

class Order
{

    public static function save($userName, $userEmail, $userPhone, $userComment, $userId,
$products)
    {
        // DB Connection
        $db = Db::getConnection();

        // DB query text
        $sql = "INSERT INTO product_order (user_name,user_email, user_phone,
user_comment, user_id, products) "
            . "VALUES (:user_name, :user_email, :user_phone, :user_comment, :user_id,
:products)";

        $products = json_encode($products);

        $result = $db->prepare($sql);
        $result->bindParam(":user_name", $userName, PDO::PARAM_STR);
        $result->bindParam(":user_email", $userEmail, PDO::PARAM_STR);
        $result->bindParam(":user_phone", $userPhone, PDO::PARAM_STR);
        $result->bindParam(":user_comment", $userComment, PDO::PARAM_STR);
        $result->bindParam(":user_id", $userId, PDO::PARAM_INT);
        $result->bindParam(":products", $products, PDO::PARAM_STR);

        return $result->execute();
    }

    public static function updateOrderById($id, $options)
    {
        // DB connection
        $db = Db::getConnection();

        // DB query text
        $sql = "UPDATE product_order SET "
            . "user_name = :userName,"
            . "user_email = :userEmail,"
            . "user_phone = :userPhone,"
            . "user_comment = :userComment,"
            . "user_id = :userId,"
            . "status = :status "
            . "WHERE id = :id";

        // Getting and returning the results. Prepare Request Used.
        $result = $db->prepare($sql);
        $result->bindParam(":id", $id, PDO::PARAM_INT);
        $result->bindParam(":userName", $options['userName'], PDO::PARAM_STR);

```

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 71   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

```

$result->bindParam(":userEmail", $options['userEmail'], PDO::PARAM_STR);
$result->bindParam(":userPhone", $options['userPhone'], PDO::PARAM_STR);
$result->bindParam(":userComment", $options['userComment'], PDO::PARAM_STR);
$result->bindParam(":userId", $options['userId'], PDO::PARAM_INT);
$result->bindParam(":status", $options['status'], PDO::PARAM_INT);
$result->execute();
}

public static function getOrdersList()
{
    // DB Connection
    $db = Db::getConnection();

    // DB query
    $sql = "SELECT id, user_email, user_name, date, status FROM product_order";

    // Getting and returning results
    $result = $db->query($sql);
    $ordersList = array();
    $i = 0;
    while ($row = $result->fetch()) {
        $ordersList[$i]['id'] = $row['id'];
        $ordersList[$i]['user_name'] = $row['user_name'];
        $ordersList[$i]['user_email'] = $row['user_email'];
        $ordersList[$i]['date'] = $row['date'];
        $ordersList[$i]['status'] = $row['status'];
        $i++;
    }

    return $ordersList;
}

public static function getOrderListById($userId)
{
    // Підключення до бази даних
    $db = Db::getConnection();

    // Запит до бази даних
    $sql = "SELECT id, user_email, user_name, date, status FROM product_order
    WHERE user_id = :userId";

    // Отримання результатів запиту
    $result = $db->prepare($sql);
    $result->bindParam(":userId", $userId, PDO::PARAM_INT);
    $result->execute();

    // Створення масиву з даними про замовлення
    $orders = array();
    $i = 0;
    while ($row = $result->fetch()) {
        $orders[$i]['id'] = $row['id'];

```

```

        $orders[$i]['user_name'] = $row['user_name'];
        $orders[$i]['user_email'] = $row['user_email'];
        $orders[$i]['date'] = $row['date'];
        $orders[$i]['status'] = $row['status'];
        $i++;
    }

    return $orders;
}

public static function getOrderById($id)
{
    // DB connection
    $db = Db::getConnection();

    // DB query
    $sql = "SELECT * FROM product_order WHERE id = :id";

    // Get and returns results (use prepare request)
    $result = $db->prepare($sql);
    $result->bindParam(":id", $id, PDO::PARAM_INT);

    // Indicate that we want to get data in the form of an array
    $result->setFetchMode(PDO::FETCH_ASSOC);

    // Execute the request
    $result->execute();

    // Return data
    return $result->fetch();
}

public static function deleteOrderById($id)
{
    // DB connection
    $db = Db::getConnection();

    // DB query
    $sql = 'DELETE FROM product_order WHERE id = :id';

    // Getting and returning the results
    $result = $db->prepare($sql);
    $result->bindParam(":id", $id, PDO::PARAM_INT);
    $result->execute();
}

public static function getQuantityInOrder($id)
{
    // Get information about order
    $order = self::getOrderById($id);

```

```

$productsJson = $order['products'];
$productsObject = json_decode($productsJson);
$productsArray = [];

foreach ($productsObject as $id => $quantity) {
    $productsArray[$id] = $quantity;
}

return $productsArray;
}

public static function getStatusText($status)
{
    switch ($status) {
        case '1':
            return 'Нове замовлення';
            break;
        case '2':
            return 'В обробці';
            break;
        case '3':
            return 'Доставлено';
            break;
        case '4':
            return 'Закрито';
            break;
    }
}
}

```

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.Л. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 74   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

```

class Product
{
    // Number of items displayed by default
    const SHOW_BY_DEFAULT = 6;

    public static function getLatestProducts($count = self::SHOW_BY_DEFAULT)
    {
        // DB connection
        $db = Db::getConnection();

        // DB query text
        $sql = 'SELECT id, name, price, is_new FROM product '
            . 'WHERE status="1" ORDER BY id DESC '
            . 'LIMIT :count';

        // Prepare Request Used
        $result = $db->prepare($sql);
        $result->bindParam(':count', $count, PDO::PARAM_INT);

        // Indicate that we want to get data in the form of an array
        $result->setFetchMode(PDO::FETCH_ASSOC);

        // Command execution
        $result->execute();

        // Getting and returning results
        $i = 0;
        $productsList = array();
        while ($row = $result->fetch()) {
            $productsList[$i]['id'] = $row['id'];
            $productsList[$i]['name'] = $row['name'];
            $productsList[$i]['price'] = $row['price'];
            $productsList[$i]['is_new'] = $row['is_new'];
            $i++;
        }
        return $productsList;
    }
    public static function getFeaturedProducts()
    {
        // DB connection
        $db = Db::getConnection();

        // DB query text
        $sql = 'SELECT id, name, price, is_new FROM product '
            . 'WHERE status="1" AND is_featured="1" '
            . 'ORDER BY id DESC';
    }
}

```

```

$result = $db->query($sql);

// Getting and returning results
$i = 0;
$productsList = array();
while ($row = $result->fetch()) {
    $productsList[$i]['id'] = $row['id'];
    $productsList[$i]['name'] = $row['name'];
    $productsList[$i]['price'] = $row['price'];
    $productsList[$i]['is_new'] = $row['is_new'];
    $i++;
}
return $productsList;
}

public static function getProductsListByCategory($categoryId, $page = 1)
{
    $limit = Product::SHOW_BY_DEFAULT;
    // Offset(for request)
    $offset = ($page - 1) * self::SHOW_BY_DEFAULT;

    // DB connection
    $db = Db::getConnection();

    // DB query text
    $sql = "SELECT id, name, price, is_new, availability, title FROM product "
        . "WHERE status = '1' AND category_id = :category_id "
        . "ORDER BY id ASC LIMIT :limit OFFSET :offset";

    // Prepare Request Used
    $result = $db->prepare($sql);
    $result->bindParam(':category_id', $categoryId, PDO::PARAM_INT);
    $result->bindParam(':limit', $limit, PDO::PARAM_INT);
    $result->bindParam(':offset', $offset, PDO::PARAM_INT);

    // Command execution
    $result->execute();

    // Getting and returning results
    $i = 0;
    $products = array();
    while ($row = $result->fetch()) {
        $products[$i]['id'] = $row['id'];
        $products[$i]['name'] = $row['name'];
        $products[$i]['price'] = $row['price'];
        $products[$i]['is_new'] = $row['is_new'];
        $products[$i]['title'] = $row['title'];
        $products[$i]['availability'] = $row['availability'];
        $i++;
    }
}

```

```

        return $products;
    }

    public static function getProductsListBySubCategory($categoryId,
    $subCategoryId, $page = 1)
    {
        $limit = Product::SHOW_BY_DEFAULT;
        // Offset(for request)
        $offset = ($page - 1) * self::SHOW_BY_DEFAULT;

        // DB connection
        $db = Db::getConnection();

        // DB query text
        $sql = "SELECT id, name, price, is_new FROM product "
            . "WHERE status='1' AND category_id = :category_id AND sub_category_id
= :sub_category_id "
            . "ORDER BY id ASC LIMIT :limit OFFSET :offset";

        // Prepare Request Used
        $result = $db->prepare($sql);
        $result->bindParam(':category_id', $categoryId, PDO::PARAM_INT);
        $result->bindParam(':sub_category_id', $subCategoryId, PDO::PARAM_INT);
        $result->bindParam(':limit', $limit, PDO::PARAM_INT);
        $result->bindParam(':offset', $offset, PDO::PARAM_INT);

        // Command execution
        $result->execute();

        // Getting and returning results
        $i = 0;
        $products = array();
        while ($row = $result->fetch()) {
            $products[$i]['id'] = $row['id'];
            $products[$i]['name'] = $row['name'];
            $products[$i]['price'] = $row['price'];
            $products[$i]['is_new'] = $row['is_new'];
            $i++;
        }
        return $products;
    }

    public static function getProductById($id)
    {
        // DB connection
        $db = Db::getConnection();

        // DB query text
        $sql = "SELECT * FROM product WHERE id = :id";

        // Prepare Request Used

```

```

$result = $db->prepare($sql);
$result->bindParam(":id", $id, PDO::PARAM_INT);

// Command execution
$result->execute();

// Getting and returning results
return $result->fetch();
}
public static function getTotalProductsInCategory($categoryId)
{
    // DB connection
    $db = Db::getConnection();

    // DB request text
    $sql = 'SELECT count(id) AS count FROM product WHERE status="1" AND
category_id = :category_id';

    // Prepare Request Used
    $result = $db->prepare($sql);
    $result->bindParam(":category_id", $categoryId, PDO::PARAM_INT);

    // Command execution
    $result->execute();

    // Return value count - quantity
    $row = $result->fetch();
    return $row['count'];
}
public static function getTotalProductsInSubCategory($subcategoryId)
{
    // DB connection
    $db = Db::getConnection();

    // DB request text
    $sql = 'SELECT count(id) AS count FROM product WHERE status="1" AND
sub_category_id = :sub_category_id';

    // Prepare Request Used
    $result = $db->prepare($sql);
    $result->bindParam(":sub_category_id", $subcategoryId, PDO::PARAM_INT);

    // Command execution
    $result->execute();

    // Return value count - quantity
    $row = $result->fetch();
    return $row['count'];
}
public static function getTotalProductsInFeaturedProducts()

```

```

{
    // DB connection
    $db = Db::getConnection();

    // DB query text
    $sql = 'SELECT count(id) AS count FROM product WHERE status="1" AND
is_featured="1"';

    // Query execution
    $result = $db->query($sql);

    // Return value count - quantity
    $row = $result->fetch();
    return $row['count'];
}
public static function getProductsByIds($idsArray)
{
    // DB connection
    $db = Db::getConnection();

    // Turn the array into a string to form a condition in the request
    $idsString = implode(',', $idsArray);

    // DB query text
    $sql = "SELECT * FROM product WHERE status = '1' AND id IN
($idsString)";

    $result = $db->query($sql);

    // Indicate, that we want to get data in the form of an array
    $result->setFetchMode(PDO::FETCH_ASSOC);

    // Getting and returning resultss
    $i = 0;
    $products = array();
    while($row = $result->fetch()) {
        $products[$i]['id'] = $row['id'];
        $products[$i]['name'] = $row['name'];
        $products[$i]['description'] = $row['description'];
        $products[$i]['code'] = $row['code'];
        $products[$i]['price'] = $row['price'];
        $i++;
    }
    return $products;
}
public static function getProductsList()
{
    // DB connection
    $db = Db::getConnection();

```

```

        // Getting and returning results
        $result = $db->query('SELECT id, name, price, code FROM product ORDER
BY id ASC');
        $productsList = array();
        $i = 0;
        while ($row = $result->fetch()) {
            $productsList[$i]['id'] = $row['id'];
            $productsList[$i]['name'] = $row['name'];
            $productsList[$i]['price'] = $row['price'];
            $productsList[$i]['code'] = $row['code'];
            $i++;
        }
        return $productsList;
    }
    public static function deleteProductById($id)
    {
        // DB connection
        $db = Db::getConnection();

        // DB query text
        $sql = 'DELETE FROM product WHERE id = :id';

        // Getting and returning results. Prepare Request Used
        $result = $db->prepare($sql);
        $result->bindParam(":id", $id, PDO::PARAM_INT);
        return $result->execute();
    }

    public static function createProduct($options)
    {
        // DB connection
        $db = Db::getConnection();

        // DB query text
        $sql = 'INSERT INTO product '
            . '(name, code, price, category_id, sub_category_id, brand, availability, '
            . 'description, is_new, is_featured, status) '
            . 'VALUES '
            . '(:name, :code, :price, :category_id, :subcategory_id, :brand, :availability, '
            . ':description, :is_new, :is_featured, :status)';

        // Getting and returning results. Prepare Request Used
        $result = $db->prepare($sql);
        $result->bindParam(":name", $options['name'], PDO::PARAM_STR);
        $result->bindParam(":code", $options['code'], PDO::PARAM_STR);
        $result->bindParam(":price", $options['price'], PDO::PARAM_STR);
        $result->bindParam(":category_id", $options['category_id'],
        PDO::PARAM_INT);
        $result->bindParam(":subcategory_id", $options['subcategory_id'],
        PDO::PARAM_INT);
    }

```

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.П. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 80   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

```

        $result->bindParam(":brand", $options['brand'], PDO::PARAM_STR);
        $result->bindParam(":availability", $options['availability'],
PDO::PARAM_INT);
        $result->bindParam(":description", $options['description'],
PDO::PARAM_STR);
        $result->bindParam(":is_new", $options['is_new'], PDO::PARAM_INT);
        $result->bindParam(":is_featured", $options['is_featured'],
PDO::PARAM_INT);
        $result->bindParam(":status", $options['status'], PDO::PARAM_INT);
        if ($result->execute()) {
            // If the request is completed successfully, return the id of the added record
            return $db->lastInsertId();
        }
        // Otherwise return 0
        return 0;
    }

    public static function updateProductById($id, $options)
    {
        // DB connection
        $db = Db::getConnection();

        // DB query text
        $sql = "UPDATE product
        SET
            name          = :name,
            code          = :code,
            price         = :price,
            category_id   = :category_id,
            sub_category_id = :subcategory_id,
            brand         = :brand,
            availability   = :availability,
            description    = :description,
            is_new        = :is_new,
            is_featured   = :is_featured,
            status        = :status
        WHERE id = :id";

        // Getting and returning results. Prepare Request Used
        $result = $db->prepare($sql);
        $result->bindParam(":id", $id, PDO::PARAM_INT);
        $result->bindParam(":name", $options['name'], PDO::PARAM_STR);
        $result->bindParam(":code", $options['code'], PDO::PARAM_STR);
        $result->bindParam(":price", $options['price'], PDO::PARAM_STR);
        $result->bindParam(":category_id", $options['category_id'],
PDO::PARAM_INT);
        $result->bindParam(":subcategory_id", $options['subcategory_id'],
PDO::PARAM_INT);
        $result->bindParam(":brand", $options['brand'], PDO::PARAM_STR);
        $result->bindParam(":availability", $options['availability'],

```

```

PDO::PARAM_INT);
    $result->bindParam(":description", $options['description'],
PDO::PARAM_STR);
    $result->bindParam(":is_new", $options['is_new'], PDO::PARAM_INT);
    $result->bindParam(":is_featured", $options['is_featured'],
PDO::PARAM_INT);
    $result->bindParam(":status", $options['status'], PDO::PARAM_INT);
    return $result->execute();
}
public static function getAvailabilityText($availability)
{
    switch ($availability) {
        case '1':
            return 'In Stock';
            break;
        case '0':
            return 'Under the order';
            break;
    }
}

public static function getProductsListInOrder($id)
{
    // Get information about order
    $order = Order::getOrderById($id);

    $productsJson = $order['products'];
    $productsObject = json_decode($productsJson);
    $productsArray = [];

    foreach ($productsObject as $id => $quantity) {
        $productsArray[$id] = $quantity;
    }

    $productsIds = array_keys($productsArray);
    $products = Product::getProductsByIds($productsIds);

    return $products;
}

public static function getImage($id)
{
    // Dummy Image Name
    $noImage = 'no-image.jpg';

    // The path to the product folder
    $path = '/upload/images/products/';

    // The path to the product image
    $pathToProductImage = $path . $id . '.jpg';

```

```

        if (file_exists($_SERVER['DOCUMENT_ROOT'].$pathToProductImage)) {
            // If image for product exists
            // Return the product image path
            return $pathToProductImage;
        }

        // Return the dummy image path
        return $path . $noImage;
    }

    public static function getProductsBySearch($searchTerm)
    {
        // Підключення до бази даних
        $db = Db::getConnection();

        // Підготовка запити
        $searchTerm = '%' . $searchTerm . '%';
        $sql = "SELECT * FROM product WHERE name LIKE :searchTerm";

        // Виконання запити з використанням підготовленого запити
        $stmt = $db->prepare($sql);
        $stmt->bindParam(':searchTerm', $searchTerm, PDO::PARAM_STR);
        $stmt->execute();

        // Отримання результатів запити
        $products = $stmt->fetchAll(PDO::FETCH_ASSOC);

        return $products;
    }

    public static function getProductsByCategory($categoryId, $subCategoryId,
    $searchTerm)
    {
        // Підключення до бази даних
        $db = Db::getConnection();

        // Підготовка запити
        $searchTerm = '%' . $searchTerm . '%';
        $sql = "SELECT * FROM product WHERE category_id = :categoryId AND
        sub_category_id = :subCategoryId AND name LIKE :searchTerm";

        // Виконання запити з використанням підготовленого запити
        $stmt = $db->prepare($sql);
        $stmt->bindParam(':categoryId', $categoryId, PDO::PARAM_INT);
        $stmt->bindParam(':subCategoryId', $subCategoryId, PDO::PARAM_INT);
        $stmt->bindParam(':searchTerm', $searchTerm, PDO::PARAM_STR);
        $stmt->execute();

        // Отримання результатів запити
        $products = $stmt->fetchAll(PDO::FETCH_ASSOC);
    }

```

```

        return $products;
    }

    public static function getProductsBySubCategory($categoryId, $subCategoryId,
$searchTerm, $page = 1)
    {
        $limit = Product::SHOW_BY_DEFAULT;
        $offset = ($page - 1) * self::SHOW_BY_DEFAULT;

        $db = Db::getConnection();

        $searchTerm = '%' . $searchTerm . '%';
        $sql = "SELECT id, name, price, is_new FROM product "
            . "WHERE status='1' AND category_id = :categoryId AND sub_category_id
= :subCategoryId "
            . "ORDER BY id ASC LIMIT :limit OFFSET :offset";

        $stmt = $db->prepare($sql);
        $stmt->bindParam(':categoryId', $categoryId, PDO::PARAM_INT);
        $stmt->bindParam(':subCategoryId', $subCategoryId, PDO::PARAM_INT);
        $stmt->bindParam(':limit', $limit, PDO::PARAM_INT);
        $stmt->bindParam(':offset', $offset, PDO::PARAM_INT);
        $stmt->execute();

        $i = 0;
        $products = array();
        while ($row = $stmt->fetch()) {
            $products[$i]['id'] = $row['id'];
            $products[$i]['name'] = $row['name'];
            $products[$i]['price'] = $row['price'];
            $products[$i]['is_new'] = $row['is_new'];
            $i++;
        }

        return $products;
    }

```

```

class User
{
    /**
     * User registration
     * @param string $firstName
     * @param string $lastName
     * @param string $email
     * @param string $password
     * @param string $birth
     * @param string $company
     * @param string $address
     * @param string $city
     * @param string $state
     * @param integer $postcode
     * @param string $country
     * @param string $info
     * @param string $phone
     * @param string $role
     * @return boolean
     */

    public static function GetRoleAdm($email)
    {
        $db = Db::getConnection();

        // Запит до бази даних для отримання ролі користувача
        $sql = "SELECT role FROM users WHERE email = :email";
        $result = $db->prepare($sql);
        $result->bindParam(':email', $email, PDO::PARAM_STR);
        $result->execute();

        // Отримання результатів запиту та повернення ролі користувача
        $row = $result->fetch();
        if ($row) {
            return $row['role'];
        }
        return false;
    }

    public static function GetRoleModr($email)
    {
        $db = Db::getConnection();

        // Запит до бази даних для отримання ролі користувача
        $sql = "SELECT role FROM users WHERE email = :email";
    }

```

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.П. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 85   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

```

$result = $db->prepare($sql);
$result->bindParam(':email', $email, PDO::PARAM_STR);
$result->execute();

// Отримання результатів запиту та повернення ролі користувача
$row = $result->fetch();
if ($row) {
    return $row['role'];
}
return false;
}

public static function register($firstName, $lastName, $email, $password,
$birth, $company, $address, $city, $state, $postcode, $country, $info,
$phone)
{
    // DB connection
    $db = Db::getConnection();

    // DB query text
    $sql = "INSERT INTO user (first_name, last_name, email, password,
birth, company, "
        . "address, city, state, postcode, country, additional_info, phone) "
        . "VALUES (:firstName, :lastName, :email, :password, :birth,
:company, "
        . ":address, :city, :state, :postcode, :country, :info, :phone)";

    // Getting and returning results. Prepare Request Used.
    $result = $db->prepare($sql);
    $result->bindParam(":firstName", $firstName, PDO::PARAM_STR);
    $result->bindParam(":lastName", $lastName, PDO::PARAM_STR);
    $result->bindParam(":email", $email, PDO::PARAM_STR);
    $result->bindParam(":password", $password, PDO::PARAM_STR);
    $result->bindParam(":birth", $birth, PDO::PARAM_STR);
    $result->bindParam(":company", $company, PDO::PARAM_STR);
    $result->bindParam(":address", $address, PDO::PARAM_STR);
    $result->bindParam(":city", $city, PDO::PARAM_STR);
    $result->bindParam(":state", $state, PDO::PARAM_STR);
    $result->bindParam(":postcode", $postcode, PDO::PARAM_INT);
    $result->bindParam(":country", $country, PDO::PARAM_STR);
    $result->bindParam(":info", $info, PDO::PARAM_STR);
    $result->bindParam(":phone", $phone, PDO::PARAM_STR);
    return $result->execute();
}

public static function update($id, $firstName, $lastName, $email,
$password, $birth, $company, $address, $city, $state, $postcode, $country,
$info, $phone)
{
    // DB connection

```

```

$db = Db::getConnection();

// DB query text
$sql = 'UPDATE user SET
    first_name    = :firstName,
    last_name     = :lastName,
    email         = :email,
    password      = :password,
    birth         = :birth,
    company       = :company,
    address       = :address,
    city          = :city,
    state         = :state,
    postcode      = :postcode,
    country       = :country,
    additional_info = :info,
    phone = :phone WHERE id = :id;';

// Getting and returning results. Prepare Request Used.
$result = $db->prepare($sql);
$result->bindParam(":id", $id, PDO::PARAM_INT);
$result->bindParam(":firstName", $firstName, PDO::PARAM_STR);
$result->bindParam(":lastName", $lastName, PDO::PARAM_STR);
$result->bindParam(":email", $email, PDO::PARAM_STR);
$result->bindParam(":password", $password, PDO::PARAM_STR);
$result->bindParam(":birth", $birth, PDO::PARAM_STR);
$result->bindParam(":company", $company, PDO::PARAM_STR);
$result->bindParam(":address", $address, PDO::PARAM_STR);
$result->bindParam(":city", $city, PDO::PARAM_STR);
$result->bindParam(":state", $state, PDO::PARAM_STR);
$result->bindParam(":postcode", $postcode, PDO::PARAM_INT);
$result->bindParam(":country", $country, PDO::PARAM_STR);
$result->bindParam(":info", $info, PDO::PARAM_STR);
$result->bindParam(":phone", $phone, PDO::PARAM_STR);

return $result->execute();
}

public static function checkUserData($email, $password)
{
    // DB connection
    $db = Db::getConnection();

    // DB query text
    $sql = 'SELECT * FROM user WHERE email = :email AND password
= :password';

    // Getting and returning results. Prepare Request Used.
    $result = $db->prepare($sql);
    $result->bindParam(":email", $email, PDO::PARAM_STR);

```

```

$result->bindParam(":password", $password, PDO::PARAM_STR);
$result->execute();

// Turn to the record
$user = $result->fetch();

if ($user) {
    // If the record exists, return user ID
    return $user['id'];
}
return false;
}
function getRoleByUsername($userName)
{
    $db = Db::getConnection();
    // Підключення до бази даних

    // Виконання запиту до бази даних
    $query = "SELECT role FROM users WHERE username =
'$userName'";
    $result = mysqli_query($db, $query);

    // Обробка результатів запиту
    if ($result && mysqli_fetch_assoc($result)) {
        mysqli_data_seek($result, 0);
        $row = mysqli_fetch_assoc($result);
        $role = $row['role'];
        return $role;
    } else {
        return false;
    }
}

public static function auth($userId)
{
    // write the user ID in the session
    $_SESSION['user'] = $userId;
}

public static function checkLogged()
{
    // If there is a session, return the user ID
    if (isset($_SESSION['user'])) {
        return $_SESSION['user'];
    }

    header("Location: /login");
}

public static function isGuest()

```

```

{
    if (isset($_SESSION['user'])) {
        return false;
    }
    return true;
}

public static function getUsername()
{
    // Get username by id
    if (isset($_SESSION['user'])) {
        $userId = $_SESSION['user'];
        $user = self::getUserById($userId);
        $userName = $user['first_name'];

        return $userName;
    }

    return 'guest';
}

public static function checkFirstName($firstName)
{
    if (strlen($firstName) >= 2) {
        return true;
    }
    return false;
}

public static function checkLastName($lastName)
{
    if (strlen($lastName) >= 2) {
        return true;
    }
    return false;
}

public static function checkEmail($email)
{
    if (filter_var($email, FILTER_VALIDATE_EMAIL)) {
        return true;
    }
    return false;
}

public static function checkEmailExists($email)
{
    // DB connection
    $db = Db::getConnection();

```

```

// DB query text
$sql = 'SELECT COUNT(*) FROM user WHERE email = :email';

// Getting and returning results. Prepare Request Used.
$result = $db->prepare($sql);
$result->bindParam(":email", $email, PDO::PARAM_STR);
$result->execute();

if ($result->fetchColumn())
    return true;
return false;
}

public static function checkPassword($password)
{
    if (strlen($password) >= 6) {
        return true;
    }
    return false;
}

public static function checkPhone($phone)
{
    if (strlen($phone) >= 10) {
        return true;
    }
    return false;
}

public static function getUserById($id)
{
    // DB connection
    $db = Db::getConnection();

    // DB query text
    $sql = "SELECT * FROM user WHERE id = :id";

    // Getting and returning results. Prepare Request Used.
    $result = $db->prepare($sql);
    $result->bindParam(":id", $id, PDO::PARAM_INT);

    // Indicate, that want to get data in the form of an array
    $result->setFetchMode(PDO::FETCH_ASSOC);
    $result->execute();

    return $result->fetch();
}

public static function getUserByEmail($email)
{
    // DB connection

```

```

$db = Db::getConnection();

// DB query text
$sql = "SELECT * FROM user WHERE email = :email";

// Getting and returning results. Prepare Request Used.
$result = $db->prepare($sql);
$result->bindParam(":email", $email, PDO::PARAM_STR);

// Indicate, that want to get data in the form of an array
$result->setFetchMode(PDO::FETCH_ASSOC);
$result->execute();

return $result->fetch();
}

public static function resetPassword($email)
{
    // DB connection
    $db = Db::getConnection();

    // New password
    $password = rand(123456, 987654);

    // DB query text
    $sql = "UPDATE user SET password = :password WHERE email = :email";

    // Getting and returning results. Prepare Request Used.
    $result = $db->prepare($sql);
    $result->bindParam(":email", $email, PDO::PARAM_STR);
    $result->bindParam(":password", $password, PDO::PARAM_INT);

    if ($result->execute()) {
        return $password;
    }
}

public static function getUserList()
{
    // Підключення до БД
    $db = Db::getConnection();

    // Отримання та повернення результатів
    $result = $db->query("SELECT id, first_name, last_name, email, password, birth, company, address, city, state, postcode, country, additional_info, phone, date, status, role FROM user ORDER BY id ASC");
    $userList = array();
    $i = 0;
    while ($row = $result->fetch()) {
        $userList[$i]['id'] = $row['id'];

```

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.П. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 91   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

```

        $userList[$i]['first_name'] = $row['first_name'];
        $userList[$i]['last_name'] = $row['last_name'];
        $userList[$i]['email'] = $row['email'];
        $userList[$i]['password'] = $row['password'];
        $userList[$i]['birth'] = $row['birth'];
        $userList[$i]['company'] = $row['company'];
        $userList[$i]['address'] = $row['address'];
        $userList[$i]['city'] = $row['city'];
        $userList[$i]['state'] = $row['state'];
        $userList[$i]['postcode'] = $row['postcode'];
        $userList[$i]['country'] = $row['country'];
        $userList[$i]['additional_info'] = $row['additional_info'];
        $userList[$i]['phone'] = $row['phone'];
        $userList[$i]['date'] = $row['date'];
        $userList[$i]['status'] = $row['status'];
        $userList[$i]['role'] = $row['role'];
        $i++;
    }
    return $userList;
}

public static function deleteUserById($id)
{
    // Підключення до БД
    $db = Db::getConnection();

    // Текст запиту до БД
    $sql = 'DELETE FROM user WHERE id = :id';

    // Отримання та повернення результату. Використовується
    // підготовлений запит
    $result = $db->prepare($sql);
    $result->bindParam(":id", $id, PDO::PARAM_INT);
    return $result->execute();
}

public static function saveRole($role)
{
    // DB Connection
    $db = Db::getConnection();

    // DB query text
    $sql = "INSERT INTO role) "
        . "VALUES (:role)";

    $result = $db->prepare($sql);
    $result->bindParam(":role", $role, PDO::PARAM_STR);

    return $result->execute();
}

public static function updateUserRoleById($id, $options)
{

```

```

// Підключення до БД
$db = Db::getConnection();

// Формування рядка запиту до БД
$sql = "UPDATE user SET
    role = :role
    WHERE id = :id";

// Підготовка запиту до БД
$result = $db->prepare($sql);

// Прив'язка значень параметрів до запиту
$result->bindParam(":role", $options['role'], PDO::PARAM_STR);
$result->bindParam(":id", $id, PDO::PARAM_INT);

// Виконання запиту до БД
return $result->execute();
}
public static function updateUserById($id, $options)
{
    // Підключення до БД
    $db = Db::getConnection();

    // Формування рядка запиту до БД
    $sql = "UPDATE user SET
        first_name = :first_name,
        last_name = :last_name,
        email = :email,
        password = :password,
        birth = :birth,
        company = :company,
        address = :address,
        city = :city,
        state = :state,
        postcode = :postcode,
        country = :country,
        additional_info = :additional_info,
        phone = :phone,
        date = :date,
        status = :status,
        role = :role
        WHERE id = :id";

    // Підготовка запиту до БД
    $result = $db->prepare($sql);

    // Прив'язка значень параметрів до запиту
    $result->bindParam(":first_name", $options['first_name'],
PDO::PARAM_STR);
    $result->bindParam(":last_name", $options['last_name'],

```

|      |      |              |        |      |                          |      |
|------|------|--------------|--------|------|--------------------------|------|
|      |      | Мельник Б.П. |        |      | ІПЗ.КР.Б – 121 – 23 – ПЗ | Арк. |
|      |      | Морозов А.В. |        |      |                          | 93   |
| Змн. | Арк. | № докум.     | Підпис | Дата |                          |      |

```

PDO::PARAM_STR);
    $result->bindParam(":email", $options['email'], PDO::PARAM_STR);
    $result->bindParam(":password", $options['password'],
PDO::PARAM_STR);
    $result->bindParam(":birth", $options['birth'], PDO::PARAM_STR);
    $result->bindParam(":company", $options['company'],
PDO::PARAM_STR);
    $result->bindParam(":address", $options['address'],
PDO::PARAM_STR);
    $result->bindParam(":city", $options['city'], PDO::PARAM_STR);
    $result->bindParam(":state", $options['state'], PDO::PARAM_STR);
    $result->bindParam(":postcode", $options['postcode'],
PDO::PARAM_STR);
    $result->bindParam(":country", $options['country'],
PDO::PARAM_STR);
    $result->bindParam(":additional_info", $options['additional_info'],
PDO::PARAM_STR);
    $result->bindParam(":phone", $options['phone'], PDO::PARAM_STR);
    $result->bindParam(":date", $options['date'], PDO::PARAM_STR);
    $result->bindParam(":status", $options['status'], PDO::PARAM_STR);
    $result->bindParam(":role", $options['role'], PDO::PARAM_STR);
    $result->bindParam(":id", $id, PDO::PARAM_INT);

// Виконання запиту до БД
return $result->execute();

```

|      |      |              |        |      |                                            |      |
|------|------|--------------|--------|------|--------------------------------------------|------|
|      |      | Мельник Б.Л. |        |      | Житомирська політехніка.20.121.13.000 – ПЗ | Арк. |
|      |      | Єфремов Ю.М. |        |      |                                            |      |
| Змн. | Арк. | № докум.     | Підпис | Дата |                                            | 95   |

