**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА**

**ФАКУЛЬТЕТ УПРАВЛІННЯ ФІНАНСАМИ ТА БІЗНЕСУ**

**Кафедра цифрової економіки та бізнес-аналітики**

**КУРСОВА РОБОТА**

з проектування та адміністрування БД і СД

на тему:

**Інформаційна система для магазину меблів**

**Галузь знань:** 05 «Соціальні та поведінкові науки»

**Спеціальність:** 051 «Економіка»

**Спеціалізація:** «Інформаційні технології в бізнесі»

**Освітній ступінь:**  бакалавр

|  |  |  |
| --- | --- | --- |
| **Науковий керівник:**  Депутат Б.Я. доц., к.ф.-м.н.  \_\_\_\_\_\_\_\_\_\_\_“\_\_\_” травня 2024 р.  (підпис) |  | **Виконавець:**  студент(ка) групи УФЕ-31 с  Тимчишин С. О.  \_\_\_\_\_\_\_\_\_\_ “\_\_\_”травня 2024 р.  (підпис) |

**Загальна кількість балів** \_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис, ПІП членів комісії)

**ЛЬВІВ 2024**

**ЗМІСТ**

[ВСТУП 2](#_Toc167269270)

[РОЗДІЛ 1. АНАЛІЗ ВИМОГ ІНФОРМАІЙНОЇ СИСТЕМИ ДЛЯ МАГАЗИНУ МЕБЛІВ 4](#_Toc167269271)

[1. 1. Постановка завдання 4](#_Toc167269272)

[1.2. Розробка моделі варіантів використання веб-сайту 5](#_Toc167269273)

[1.3. Аналіз засобів реалізації (техніко-економічне обґрунтування вибору) 7](#_Toc167269274)

[РОЗДІЛ 2. РОЗРОБКА БАЗИ ДАНИХ ДЛЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ МЕБЛЕВОГО МАГАЗИНУ 10](#_Toc167269275)

[2.1. Опис моделі даних 10](#_Toc167269276)

[2.2. Нормалізація відношень 11](#_Toc167269277)

[2.3. Визначення типів даних 12](#_Toc167269278)

[2.4. Обмеження цілісності даних 16](#_Toc167269279)

[2.5. Реалізація SQL-скрипту 18](#_Toc167269280)

[РОЗДІЛ 3. РОЗРОБКА ВЕБ-ДОДАТКУ ДЛЯ МАГАЗИНУ МЕБЛІВ 24](#_Toc167269281)

[3.1. Структура веб-сайту 24](#_Toc167269282)

[3.2. Макети сторінок веб-сайту 26](#_Toc167269283)

[3.3. Програмування серверної частини 29](#_Toc167269284)

[3.4. Програмування клієнтської частини 30](#_Toc167269285)

[3.5. Розміщення веб-сайту на локальному віртуальному середовищі або в Інтернеті 32](#_Toc167269286)

[ВИСНОВКИ 33](#_Toc167269287)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 34](#_Toc167269288)

[ДОДАТКИ 35](#_Toc167269289)

ВСТУП

У сучасному світі розвиток технологій та автоматизація процесів є невід'ємною складовою успішної діяльності практично в усіх галузях. Однією з найважливіших сфер, яка вимагає ефективного використання інформаційних технологій, є торгівля. Зокрема, магазини меблів, що представляють значний сегмент ринку, активно застосовують інформаційні системи для оптимізації бізнес-процесів, збільшення ефективності та поліпшення обслуговування клієнтів. Дуже багато житла, офісів, навчальних закладів в Україні зруйновано, і після перемоги всі відновлені споруди вимагатимуть меблювання. Це означатиме для виробників меблів більше роботи, яку необхідно буде виконати в стислі терміни та максимально дешево, в чому допоможе автоматизована підтримка бізнес-процесів таких підприємств. Саме тому розробка інформаційної системи для магазину меблів є **актуальною.**

До того ж, сучасний ринок меблів надзвичайно динамічний та конкурентний. За останні роки споживачі стали все більше вимагати не лише високої якості продукції, але й ефективного та зручного сервісу. У зв'язку з цим створення та використання інформаційних систем для магазинів меблів стає не лише актуальним, але й стратегічно важливим кроком для забезпечення конкурентоспроможності підприємств в цій сфері.

**Метою** дослідження є розробка та обґрунтування інформаційної системи для магазину меблів, яка б забезпечувала автоматизацію основних бізнес-процесів та підвищення ефективності його роботи, зокрема управління асортиментом, клієнтами та замовленнями.

**Завдання** курсової роботи полягають в наступному:

1) Проаналізувати вимоги до інформаційної системи магазину меблів з точки зору різних користувачів (адміністратори, менеджери, клієнти).;

2) Спроектувати модель Entity-Relationship (ER-model), яка буде відображати взаємозв'язки між сутностями в системі (товари, категорії, замовлення, клієнти, залишки на складі тощо). Визначити атрибути кожної сутності (назва товару, ціна, опис, характеристики, ім'я клієнта тощо).;

3) Розробити SQL-скрипт для створення таблиць бази даних, визначення атрибутів, типів даних та обмежень цілісності;

4) Розробити детальну карту сайту (site map), яка буде відображати ієрархію сторінок та їх взаємозв'язок. Провести програмування серверної та клієнтської частини веб-сайту для магазину меблів;

5) Провести тестування розробленої системи для перевірки її працездатності, безпеки та відповідності вимогам. Виправити помилки та недоліки, які можуть бути виявлені під час тестування;

6) Оцінити ефективність розробленої інформаційної системи та зробити висновки щодо її практичної значущості.

**Об'єктом** дослідження є діяльність магазину меблів, його бізнес-процеси, а **предметом** – розробка та впровадження інформаційної системи, спрямованої на автоматизацію процесів управління та обслуговування клієнтів.

**Практичне значення отриманих результатів**: Розроблена інформаційна система може бути використана для автоматизації роботи магазинів меблів, що дозволить їм підвищити свою конкурентоспроможність.

***Використане програмне забезпечення:*** *Навести мови програмування та програмне забезпечення, які були використані при написанні курсової роботи.*

**Структура** даної курсової роботи включає три розділи («Аналіз вимог інформаційної системи для магазину меблів», «Розробка бази даних для інформаційної системи меблевого магазину», «Розробка веб-додатку для магазину меблів»), висновки, список використаних джерел та х додатки.

РОЗДІЛ 1. АНАЛІЗ ВИМОГ ІНФОРМАІЙНОЇ СИСТЕМИ ДЛЯ МАГАЗИНУ МЕБЛІВ

**1. 1. Постановка завдання**

Сфера меблів, як одна з ключових галузей промисловості, відіграє важливу роль у формуванні та зміні нашого оточення. Від давніх часів меблі виступали не лише як предмети побуту, але й як символи статусу, смаку та культурного стилю. Сьогодні ця сфера постійно розвивається та змінюється, відображаючи потреби та вподобання сучасного споживача, тож і меблевим магазинам не можна стояти на місці, не тільки вдосконалюючи асортимент, а й оптимізовувати та автоматизовувати свою роботу.

Метою даної роботи є створення інформаційної системи для онлайн-магазину меблів (у магазину є лише сайт для продажів та склад, де зберігаються меблі та працюють працівники), яка сприятиме ефективному управлінню та покращенню обслуговування клієнтів. Інформаційна система включатиме базу даних та веб-сайт для продажу товарів через Інтернет, логічно зв’язані між собою.

Функціональність веб-сайту зі сторони користувача: можливість переглянути каталог товарів та фільтрувати їх за категоріями та виробником, можливість зробити замовлення, ознайомитися з контактною інформацією та інформацією про магазин. Функціональність веб-сайту зі сторони адміністрації: управління товарами, управління замовленнями, аналітика. Необхідно забезпечити зв’язок між сторінками, зручний та інтуїтивно зрозумілий інтерфейс, а також логічний та функціональний зв’язок сайту з базою даних.

База даних буде зберігати інформацію про товари, клієнтів, замовлення та інші дані необхідні для функціонування магазину. Створена база даних повинна дозволити реалізувати наступні технічні завдання (**ТЗ**):

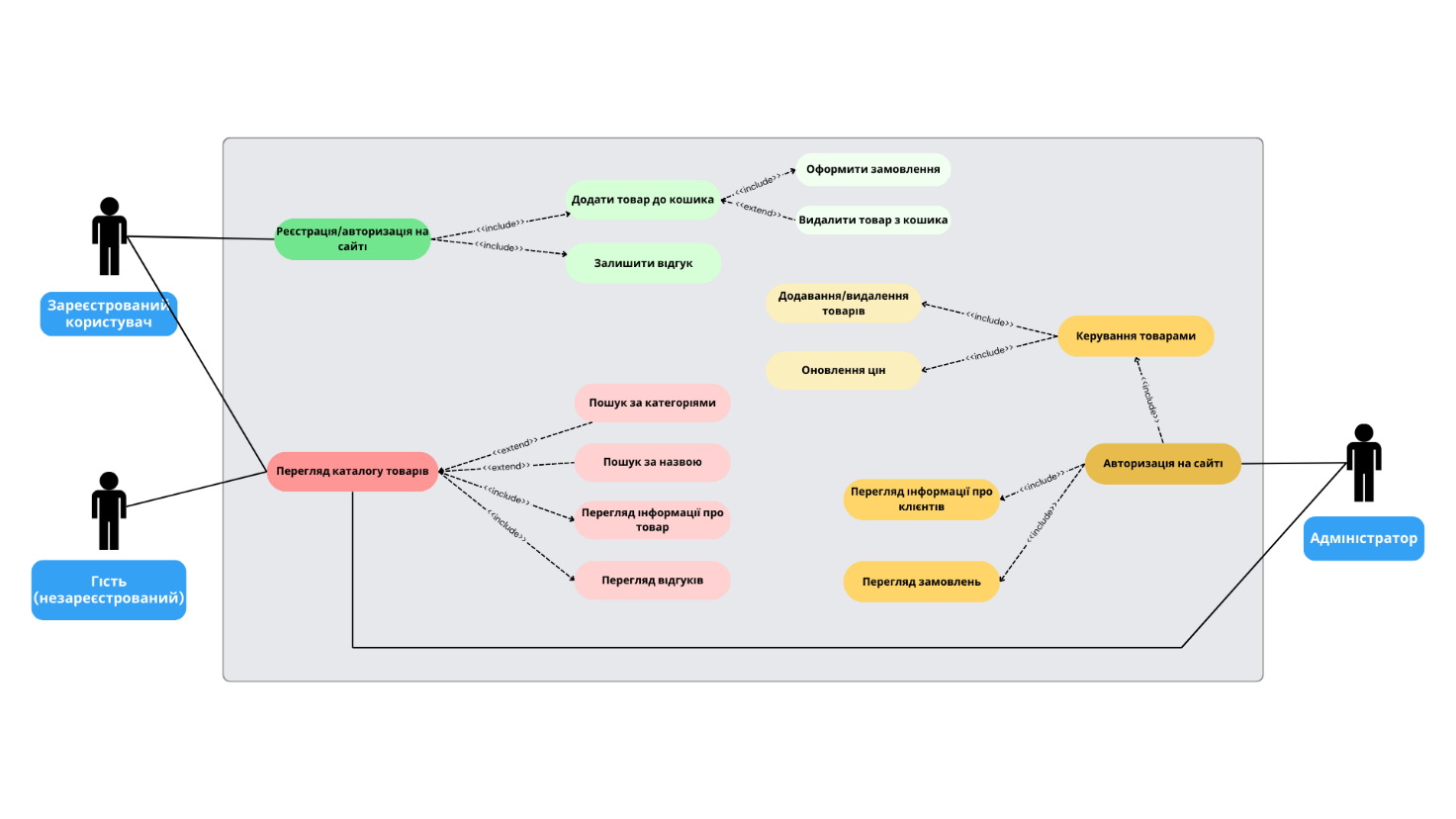
* Авторизація користувача
* Перегляд інформації про товар: назва, постачальник, ціна, розміри (висота, ширина, довжина).
* Перегляд товарів, що виготовлені з певного матеріалу.
* Пошук товару за назвою.
* Перегляд усіх товарів певної категорії з відображенням їх ціни та кількості замовлень кожної одиниці
* Список найпопулярніших товарів за певний період часу
* Список всіх клієнтів, загальна вартість їх замовлень та їх знижка
* Рейтинг категорій товарів за кількістю продажів.
* Список клієнтів, які здійснили певну кількість замовлень.
* Список клієнтів з загальною вартістю замовлень більше певної суми.
* Кількість проданих одиниць товару в певний період часу.
* Список товарів, що були куплені більше певної кількості разів.
* Статистика замовлень товарів у різні міста за певний період часу
* Показати виробників та кількість різних категорій продуктів, які вони постачають з вказанням цих категорій.
* Відобразити неоплачені замовлення.
* Відобразити всі товари та скільки разів вони були замовлені за певний період часу, а також загальну кількість замовлень в категорії, до якої відноситься товар.
* Хто більше купує чоловіки (ч) чи жінки (ж).
* Обрати 5 найпопулярніших товарів та підняти їх ціну на 5%.
* Прийняти нового працівника на роботу.
* Звільнити працівника.

**1.2. Розробка моделі варіантів використання веб-сайту**

Інформаційна система інтернет-магазину меблів має три рівні доступу:

1. Незареєстрований користувач (гість)
2. Зареєстрований користувач
3. Адміністратор

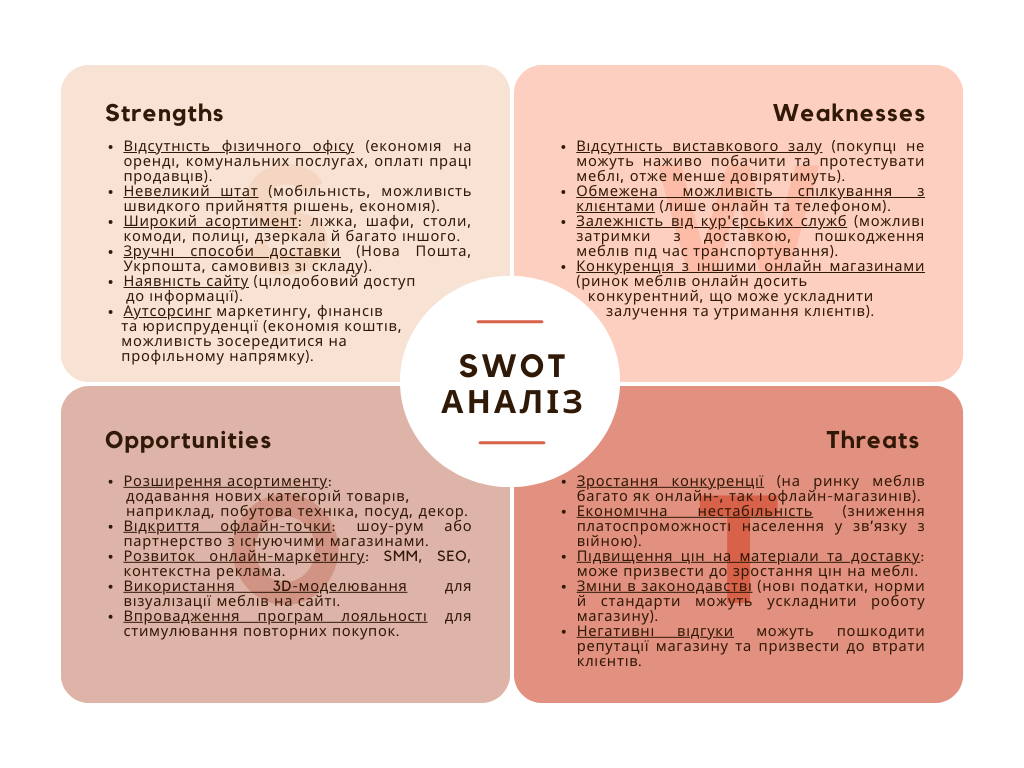
Кожен рівень доступу забезпечує певні можливості. Гість, себто незареєстрований користувач, має доступ до перегляду каталогу товарів, а відповідно і до таких функцій сайту як: пошук за категоріями, пошук за назвою, перегляд інформації про товар та перегляд відгуків, також Гість має можливість зареєструватися. Після реєстрації Гість стає Зареєстрованим користувачем, тож до вже доступних йому функцій сайту добавляється можливість додати товар до кошика (а, відповідно, і доступ до самого кошика: видалення товарів у кошику) та можливість залишити відгук. Тільки Зареєстрований користувач може оформити замовлення. Адміністратор знаходиться на найвищому рівні доступу – він може керувати товарами (додавати та видаляти їх, змінювати ціни), переглядати особисту інформацію клієнтів та їх замовлення. Всі рівні доступу сайту відображає Use Case діаграма, або ж діаграма прецедентів, що використовується для більш точного розуміння як повинна працювати система (рис. 1.1.).



**Рис. 1.1. Use Case діаграма сайту меблевого магазину**

*Джерело: Розроблено автором*

Одним з ключових інструментів стратегічного планування та управління є SWOT-аналіз. Проведення такого аналізу не лише допомагає зрозуміти внутрішні та зовнішні фактори, які впливають на бізнес, але й надає можливість ідентифікувати конкурентні переваги, виявити слабкі сторони, визначити потенційні можливості для розвитку та зрозуміти можливі загрози, що стоять перед компанією, зокрема перед магазином меблів (рис. 1.2.).



**Рис. 1.2. SWOT аналіз меблевого магазину**

*Джерело: Розроблено автором*

**1.3. Аналіз засобів реалізації (техніко-економічне обґрунтування вибору)**

Для створення інформаційної системи онлайн-магазину меблів було обрано набір технологій, що включає MySQL Workbench для розробки та управління базою даних, Visual Studio Code для написання та редагування програмного коду, та такі мови як HTML, CSS та JS для реалізації front-end частини системи. Вибір цих інструментів та технологій був обґрунтований їхньою популярністю, можливістю взаємодії та сумісності, а також швидкістю та ефективністю розробки. У майбутньому планується також використання мови програмування PHP для реалізації back-end частини системи, що дозволить забезпечити більшу гнучкість та функціональність інтернет-магазину.

HTML – скорочення від "HyperText Mark-up Language" – перекладається як "Мова розмітки гіпертексту". Іншими словами HTML – це мова розмітки, або ще один спосіб зберігання інформації. HTML використовувався для створення основної структури веб-сторінок магазину меблів. Під час розробки інформаційної системи магазину меблів він використовувався для розміщення тексту, зображень, посилань та інших елементів на сторінках [1]. Існують аналогічні мови розмітки, такі як XML (eXtensible Markup Language) або Markdown, але HTML залишається найбільш поширеною мовою для розробки веб-сторінок, саме через це її було вибрано для реалізації даної роботи.

CSS – це набір команд, які відповідають за візуалізацію сторінки. Наприклад, додати колір тексту, розмістити параграфи в рядок або колонками, змінити розмір картинок. Завдання CSS – прикрасити сайт, надати проєкту завершеного вигляду [2]. У даній роботі CSS використовувався для надання стилів і вигляду елементам, що були розміщені за допомогою HTML, зокрема встановлення кольорів, шрифтів, розмірів, відступів та інших характеристик.

JavaScript є динамічною мовою програмування, яка використовується для створення інтерактивності та динамічного контенту на веб-сторінках. Сьогодні JavaScript – одна з найпопулярніших мов програмування [3]. Під час розробки використовувалася для додавання інтерактивності та функціональності до веб-сторінок магазину меблів (валідація форм, анімація, обробка подій, взаємодія з користувачем та інше).

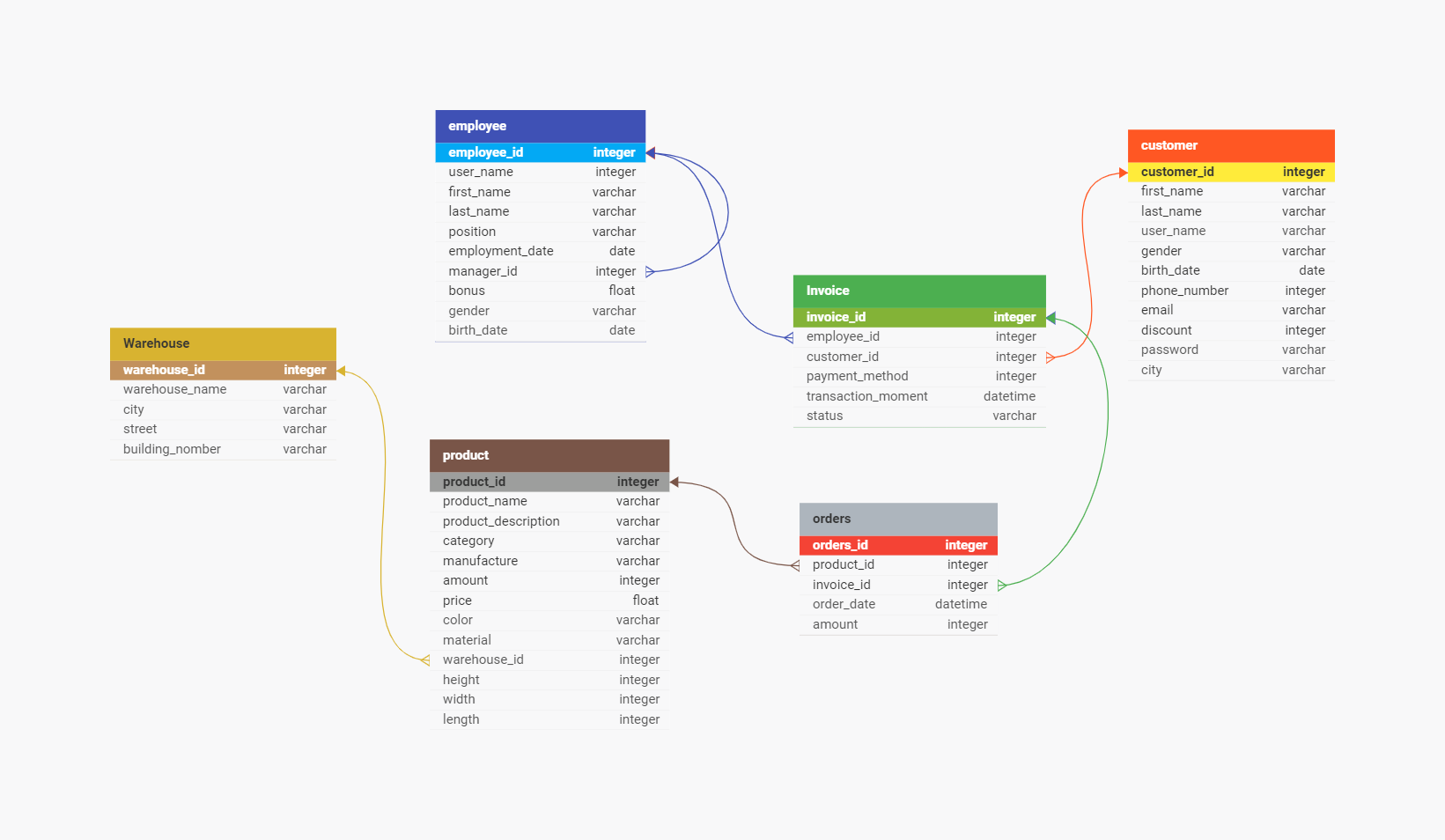
MySQL – це система керування базами даних. У реляційній базі даних дані зберігаються не всі скопом, а в окремих таблицях, завдяки чому досягається виграш в швидкості і гнучкості. Таблиці зв'язуються між собою за допомогою відносин, завдяки чому забезпечується можливість об'єднувати при виконанні запиту дані з декількох таблиць. SQL як частина системи MySQL можна охарактеризувати як мову структурованих запитів плюс найбільш поширений стандартний мова, яка використовується для доступу до баз даних [4]. MySQL було використано для створення бази даних, в якій зберігаються дані про товари, замовлення, користувачів та іншу інформацію, необхідну для роботи магазину меблів.

PHP (англ. PHP: Hypertext Preprocessor) – скриптова мова програмування, застосовується для розробки додатків та сайтів. На даний час він підтримується майже всіма хостинг-компаніями і є одним із лідерів серед мов програмування, які застосовуються для написання динамічних веб-сторінок. Це мова програмування з відкритим вихідним кодом, розроблена для вебу і може бути впроваджена в HTML код [5]. PHP буде використано для розробки back-end частини онлайн-магазину меблів. Це включатиме обробку запитів користувачів, генерацію веб-сторінок на основі даних з бази даних, управління користувачами та замовленнями та інше.

РОЗДІЛ 2. РОЗРОБКА БАЗИ ДАНИХ ДЛЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ МЕБЛЕВОГО МАГАЗИНУ

**2.1. Опис моделі даних**

Модель даних інформаційної системи онлайн-магазину меблів визначає структуру та взаємозв'язки між різними елементами даних, що використовуються для забезпечення функціональності та операцій магазину. В основі моделі лежить ER-діаграма, яка відображає сутності (об'єкти предметної області) та їхні взаємозв'язки (рис. 2.1.).



**Рис. 2.1. ER-діаграма**

*Джерело: Розроблено автором*

Модель включає наступні сутності:

* **Співробітник (employee)**. Включає дані про співробітників магазину, такі як ім'я, прізвище, нікнейм в системі, посада, дата прийняття на роботу, бонус, стать, дата народження. Кожен співробітник може мати керівника, що відображається через зовнішній ключ manager\_id.
* **Товар (product)**. Представляє дані про товари, які доступні для продажу в магазині. Це включає назву товару, опис, категорію, виробника, ціну, кількість на складі, колір, матеріал, висоту виробу, його ширину та довжину. Кожен товар пов'язаний з конкретним складом через зовнішній ключ warehouse\_id.
* **Клієнт (customer)**. Містить дані про клієнтів магазину, включаючи ім'я, прізвище, нікнейм, адресу електронної пошти, телефонний номер, стать, дату народження, місто проживання, знижку та пароль.
* **Замовлення (orders)**: Представляє інформацію про замовлення товарів: ідентифікатори товару та рахунку (product\_id, invoice \_id), кількість товару та дату замовлення.
* **Рахунок (invoice)**: Містить дані про операції продажу, себто ідентифікатори співробітника та клієнта (employee\_id, customer\_id), метод оплати, дату транзакції та статус рахунку.
* **Склад (warehouse)**: Відображає інформацію про склади, на яких зберігаються товари. Таблиця включає назву та адресу складу (місто, вулиця, будинок).

Кожна таблиця має свої унікальні поля та взаємозв'язки з іншими таблицями, що дозволяє забезпечити цілісність даних та ефективне управління інформацією в межах системи магазину меблів.

**2.2. Нормалізація відношень**

Нормалізація відношень є важливим кроком у проектуванні бази даних, що спрямований на усунення аномалій та забезпечення цілісності даних. Вихідна модель даних інтернет-магазину меблів вже пройшла процес нормалізації та відповідає всім нормальним формам.

**Перша нормальна форма (1НФ)**. Кожне поле у всіх таблицях має атомарні значення, тобто дані не розбиваються на більш прості компоненти. Усі таблиці в базі даних вже мають одноатрибутні ключі та не містять повторюваних груп атрибутів.

**Друга нормальна форма (2НФ)**. Кожен неключовий атрибут повністю залежить від первинного ключа, а не від підрядної частини ключа. Усі таблиці вже перебувають у другій нормальній формі, оскільки всі атрибути залежать від унікального ідентифікатора таблиці.

**Третя нормальна форма (3НФ)**. Усі неключові атрибути не залежать від інших неключових атрибутів, але лише від первинного ключа. Усі зв'язки між атрибутами вже були ретельно перевірені та відповідають третій нормальній формі.

Отже, структура бази даних інтернет-магазину меблів є ефективною, оптимізованою та готовою до використання, забезпечуючи належну цілісність і узгодженість даних.

**2.3. Визначення типів даних**

Для гарантії того, що в таблицях зберігаються лише ті дані, які їм підходять, а також задля ефективного використання пам'яті пристрою використовуються типи даних. Тип даних в SQL – це атрибут, який визначає вид інформації, що може зберігатися в стовпці таблиці бази даних. Типи даних гарантують, що у стовпцях зберігаються лише ті дані, які їм підходять. Наприклад, стовпець з типом даних INT може містити лише цілі числа, а стовпець з типом даних VARCHAR – лише рядки. До того ж різні типи даних займають різну кількість місця в пам'яті. Вибір правильного типу даних для кожного стовпця може допомогти зменшити розмір вашої бази даних та покращити її продуктивність. SQL використовує типи даних для визначення того, як можна порівнювати та обробляти значення в різних стовпцях. Це робить ваші запити більш чіткими та ефективними [5].

Типи даних SQL є важливою частиною мови SQL. Вони гарантують точність, ефективність та простоту використання даних. Є три основні групи типів даних: текстові дані (табл. 2.1.), числові дані (табл. 2.2.) та дані дати й часу (табл. 2.3.). В залежності від вказаного типу даних зрозмуміло який вид інформації може зберігатися, які операції можна виконувати над цими даними, та як комп'ютер повинен обробляти і зберігати їх.

*Таблиця 2.1.*

**Текстові типи даних**

|  |  |
| --- | --- |
| **Назва** | **Опис** |
| CHAR() | Містить рядок фіксованої довжини. Довжина задається в дужках |
| VARCHAR | Містить рядок змінної довжини. В дужках задається максимальна довжина |
| TINYTEXT | Рядок з найбільшою довжиною 255 символів |
| TEXT | Рядок з найбільшою довжиною 65,535 символів |
| BLOB | Великий двійковий об’єкт |
| MEDIUMTEXT | Рядок з максимальною довжиною в 16,777,215 символів. |
| MEDIUMBLOB | Рядок, який зберігає 16 Мегабайтів даних |
| LONGTEXT | Рядок, який має максимальну довжину 4,294,967,295 символів. |
| LONGBLOB | Великий двійковий об’єкт, який зберігає 4 Гігабайти даних |
| ENUM | Дозволяє вести список значень |
| SET | Містить 64 значень списку, і не зберігає більше одного вибору |

*Джерело: побудовано автором на основі [5]*

*Таблиця 2.2.*

**Числові типи даних**

|  |  |
| --- | --- |
| **Назва** | **Опис** |
| INT | Цілі числа без десяткової частини. |
| BIGINT | Великі цілі числа без десяткової частини. |
| FLOAT | Числа з плаваючою комою (одинарна точність). |
| DOUBLE | Числа з плаваючою комою подвійної точності. |
| DECIMAL | Числа з фіксованою точністю та масштабом. |
| TINYINT | Малий цілий тип з діапазоном від -128 до 127. |
| SMALLINT | Малий цілий тип з діапазоном від -32768 до 32767. |
| REAL | Числа з плаваючою комою (одинарна точність). |
| BOOLEAN | Логічний тип, який може приймати значення TRUE або FALSE. |

*Джерело: побудовано автором на основі [5]*

*Таблиця 2.3.*

**Типи даних дати та часу**

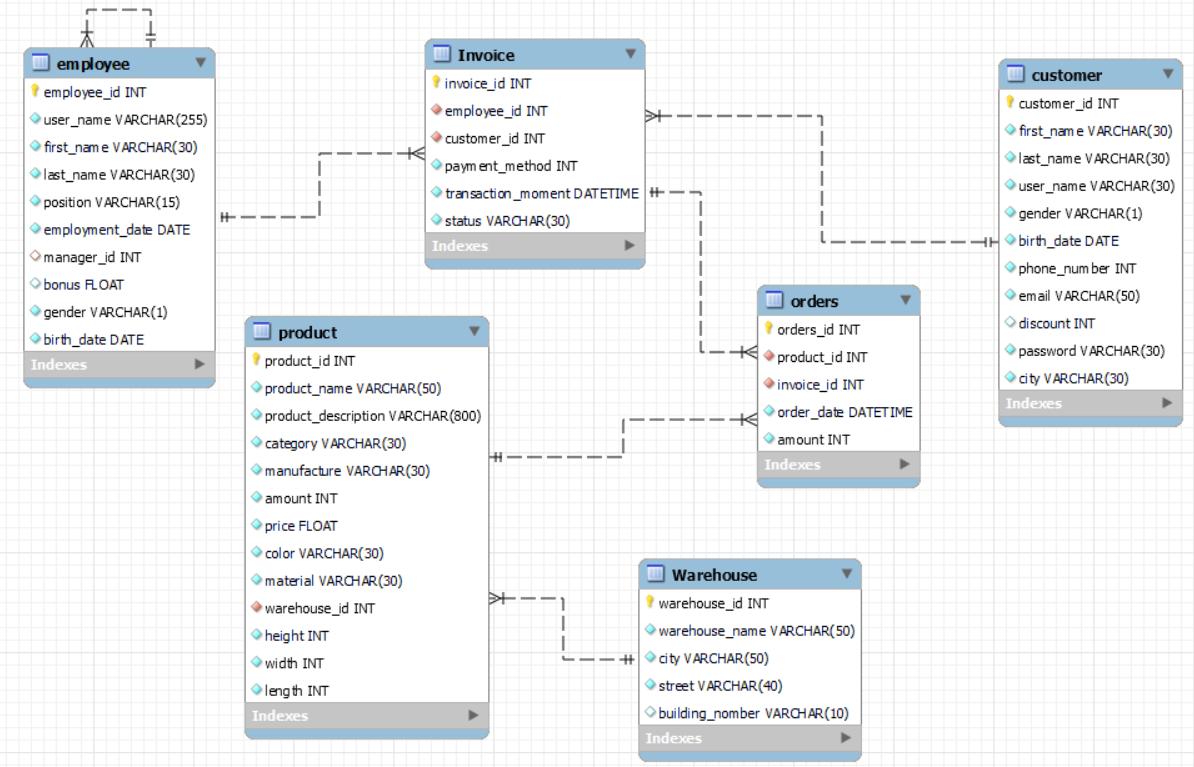
|  |  |
| --- | --- |
| **Назва** | **Опис** |
| DATE | Зберігає дату в форматі 'РРРР-ММ-ДД'. |
| TIME | Зберігає час в форматі 'ГГ:ХХ:СС'. |
| DATETIME | Зберігає дату та час в форматі 'РРРР-ММ-ДД ГГ:ХХ:СС'. |
| TIMESTAMP | Зберігає дату та час у форматі TIMESTAMP (часто використовується для відстеження змін у записах). |
| YEAR | Зберігає рік у форматі 'РРРР'. |

*Джерело: побудовано автором на основі [5]*

В базі даних магазину меблів було використано такі типи даних:

* **INT** (цілі числа). Використовується для унікальних ідентифікаторів (ключів) записів, таких як ідентифікатори співробітників (employee\_id), товарів (product\_id), клієнтів (customer\_id) та інших сутностей. Ці дані мають цілочисельне значення та є унікальними для кожного запису.
* **INT** (цілі числа). Також використано для зберігання інформації про кількість товарів на складі (amount), що дозволяє ефективно керувати запасами та відстежувати наявність товарів.
* **FLOAT** (числа з плаваючою комою). Використовується для зберігання числових даних з плаваючою комою, таких як ціна товару (price), бонуси для співробітників (bonus). Цей тип даних дозволяє точно відображати дробові числа, що є важливим для фінансових обчислень.
* **VARCHAR** (рядки змінної довжини). Було використано для зберігання текстових даних, таких як імена (first\_name, last\_name), описи (product\_description), категорії (category), назви меблів (product\_name) тощо. Розмір поля VARCHAR може бути різним в залежності від конкретних потреб, що дозволяє ефективно використовувати пам'ять бази даних.
* **DATE** (дата). Використовується для зберігання дати, таких як дата народження клієнтів (birth\_date), дата прийняття на роботу співробітників (employment\_date), дата оформлення замовлення (order\_date) тощо. Цей тип даних дозволяє ефективно виконувати операції порівняння та сортування за датами.

Детальніше розглянути типи даних та їх обмеження можна на ER-діаграмі з програми Workbench (рис. 2.2.).



**Рис. 2.2. ER-діаграма**

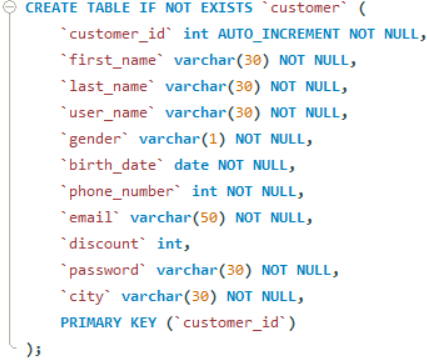
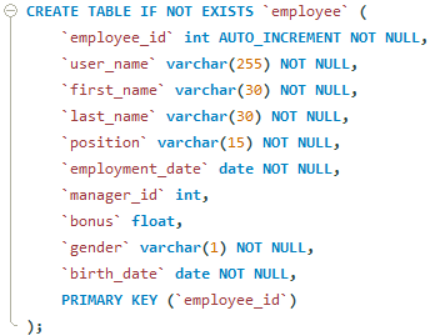
*Джерело: Розроблено автором*

**2.4. Обмеження цілісності даних**

Обмеження цілісності — це правила, які обмежують усі можливі стани бази даних, а також переходи з одного стану в інший. Таким чином, обмеження цілісності визначають множину «допустимих» станів і переходів між ними. База даних перебуває в цілісному стані, якщо вона відповідає всім визначеним для неї вимогам цілісності [6].

У таблиці `**employee**` (рис. 2.3) обмеження цілісності включають первинний ключ (PRIMARY KEY), який гарантує унікальність значень employee\_id. Поля, такі як user\_name, first\_name, last\_name, position, employment\_date, gender, та birth\_date мають обмеження NOT NULL, що забезпечує наявність даних у кожному з цих полів.

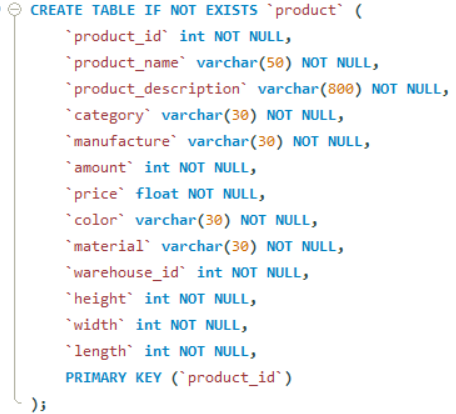
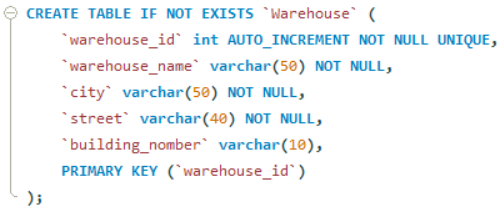
Для таблиці **`customer`** (рис. 2.3.), первинний ключ customer\_id гарантує унікальність кожного запису клієнта. Поля first\_name, last\_name, user\_name, gender, birth\_date, phone\_number, email, password, та city мають обмеження NOT NULL, що забезпечує їх заповнення.



**Рис. 2.3. Таблиці `employee` та ` customer `**

Для таблиці `**product**`(рис. 2.4), первинний ключ product\_id гарантує унікальність кожного запису товару. Поля product\_name, product\_description, category, manufacture, amount, price, color, material, warehouse\_id, height, width, та length мають обмеження NOT NULL, що забезпечує їх заповнення.

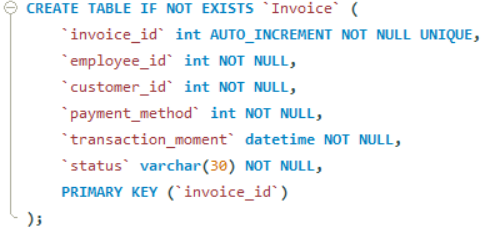
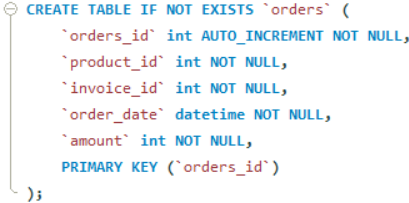
Для таблиці **`Warehouse`** (рис. 2.4.), первинний ключ warehouse\_id гарантує унікальність кожного запису складу. Поля warehouse\_name, city, street, та building\_nomber мають обмеження NOT NULL, що забезпечує їх заповнення.



**Рис. 2.4. Таблиці ` product ` та `Warehouse`**

Для таблиці **`orders`** (рис. 2.5.), первинний ключ orders\_id гарантує унікальність кожного запису замовлення. Поля product\_id, invoice\_id, order\_date, та amount мають обмеження NOT NULL, що забезпечує їх заповнення.

Для таблиці **`** **Invoice `** (рис. 2.5.), первинний ключ invoice\_id гарантує унікальність кожного запису накладної. Поля employee\_id, customer\_id, payment\_method, transaction\_moment, та status мають обмеження NOT NULL, що забезпечує їх заповнення.



**Рис. 2.5. Таблиці ` orders ` та `** **Invoice `**

Вторинні ключі (FOREIGN KEY) (рис. 2.6.) встановлюють зв'язки між таблицями, забезпечуючи цілісність даних і правильні взаємозв'язки. Наприклад, зв'язок між таблицею orders і product гарантує, що кожне замовлення відноситься до існуючого товару, а зв'язок між Invoice і customer гарантує, що кожен чек відноситься до існуючого клієнта.



**Рис. 2.6. Встановлення вторинних ключів**

Обмеження цілісності даних є важливою частиною будь-якої бази даних. Вони забезпечують точність, узгодженість та надійність даних. Використання відповідних обмежень цілісності даних може допомогти уникнути помилок, полегшити роботу з даними та покращити загальну продуктивність бази даних.

**2.5. Реалізація SQL-скрипту**

SQL-скрипт – це файл, який містить набір інструкцій SQL, що виконуються послідовно для керування даними в базі даних. Ці інструкції можуть використовуватися для створення, читання, оновлення та видалення даних, а також для керування об'єктами бази даних, такими як таблиці, індекси та користувачі [7]. Для виконання скрипту використовується MySQL Workbench.

SQL-скрипт, що відповідає за створення таблиць приведено у Додатку А. Він побудований на основі проведеного вище аналізу. Для кожної таблиці у скрипті створюються відповідні структури з визначенням типів даних для кожного атрибута, визначено первинні та воринні ключі.

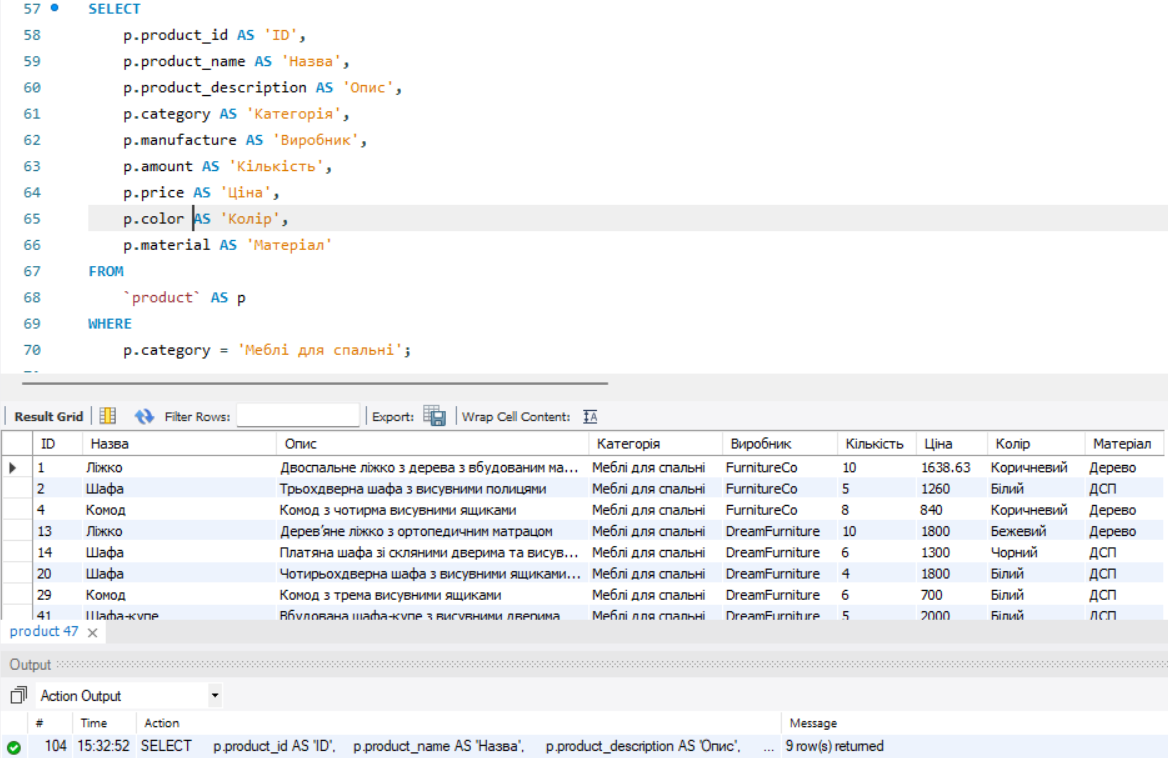
Для отримання даних з бази даних також можуть бути використані різноманітні SQL-запити. Для реалізації інформаційної системи книжкового магазину було використано такі види запитів: «SELECT», «DELETE», «UPDATE», «INSERT». Кожен запит покриває певну бізнес-вимогу інформаційної системи магазину меблів, зокрема потреби як зареєстрованих користувачів, так і адміністратора. Ось кілька запитів, актуальних для всіх рівнів доступу:

Переглянути всю інформацію про товар за ID (назва, постачальник, ціна, розміри (висота, ширина, довжина), колір та матеріал) можна за допомогою наступного запиту:



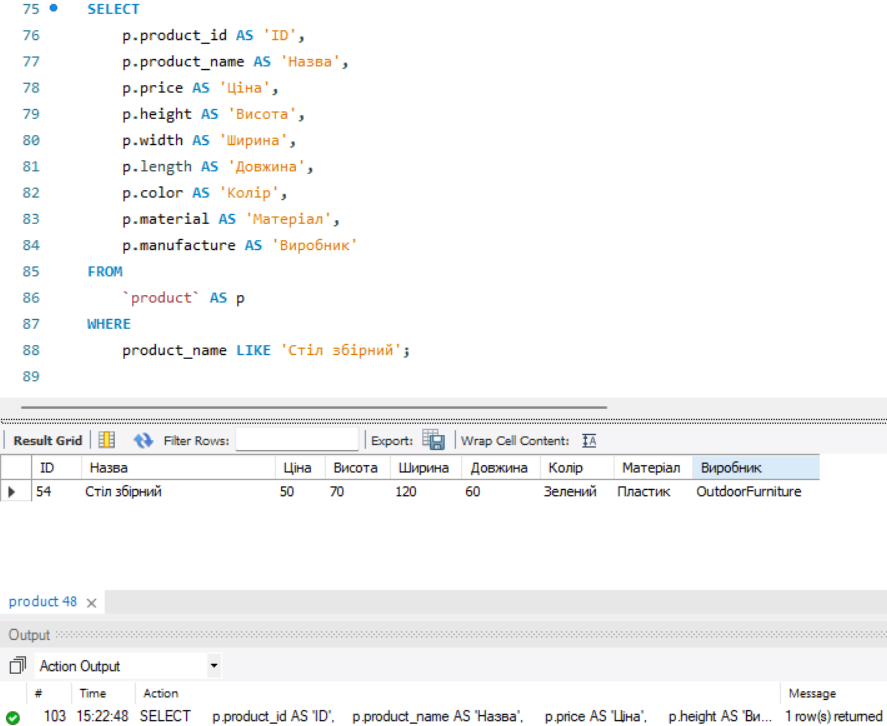
**Рис. 2.7. Вся інформація про товар**

Переглянути всі товари, що належать до певної категорії (уданому випадку, наведеному на рисунку, це «Меблі для спальні») теж можна за допомогою запиту (рис. 2.8.). Цей селект реалізовує вимогу можливості посортувати і переглядати товари за категоріями: Меблі для спальні, офісу, вітальні, їдальні, кухні, дитячої, прихожої, ванної, саду, кемпінгу, салону краси, Декоративні аксесуари.



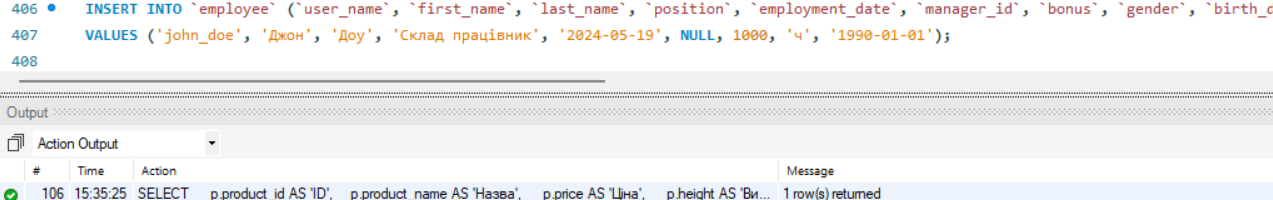
**Рис. 2.8. Сортування за категорією**

Окрім пошуку за категоріями, серед вимог є й забезпечення пошуку за іншими категоріями (матеріалом та назвою). Пошук товару за назвою (в даному випадку «Стіл збірний») реалізовує такий запит:

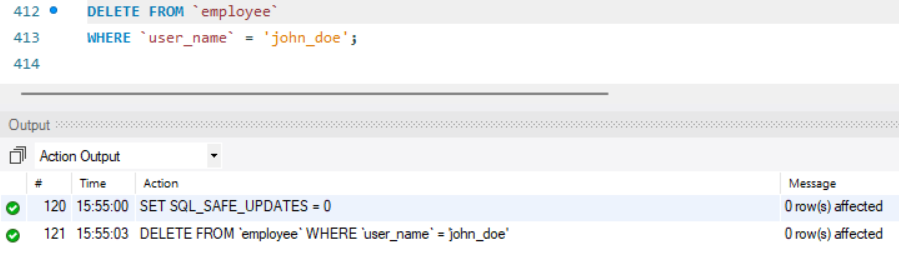


**Рис. 2.9. Пошук за назвою**

Є також запити, необхідні лише для адміністраторської частини інформаційної системи. Сюди можна віднести: Прийняття працівника на роботу (рис. 2.10) чи Звільнення працівника (рис. 2.11.).

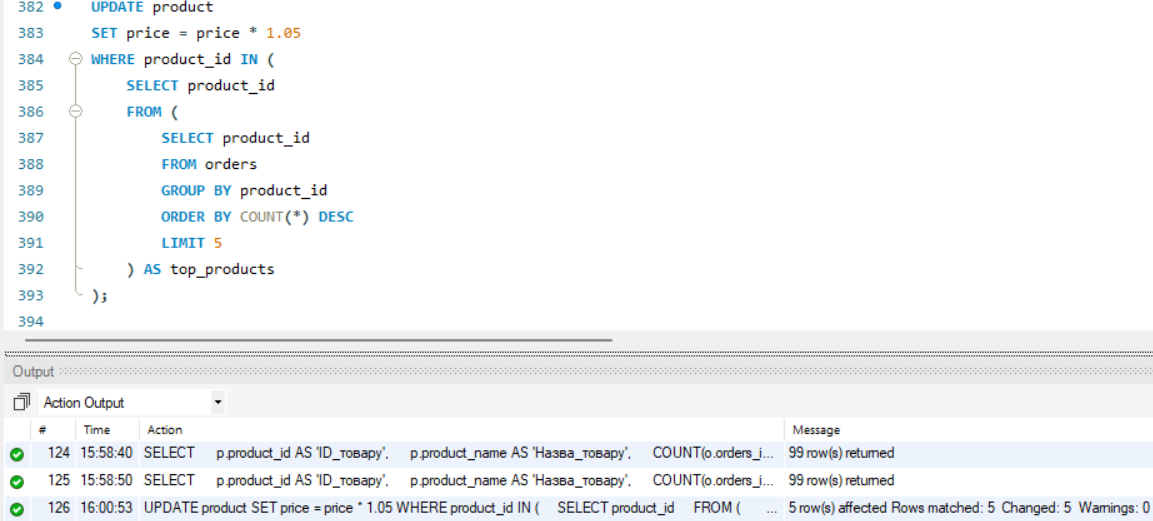


**Рис. 2.10. Працевлаштування**



**Рис. 2.11. Звільнення працівника**

Обрати 5 найпопулярніших товарів та підняти їх ціну на 5% можна за допомогою такого запиту:

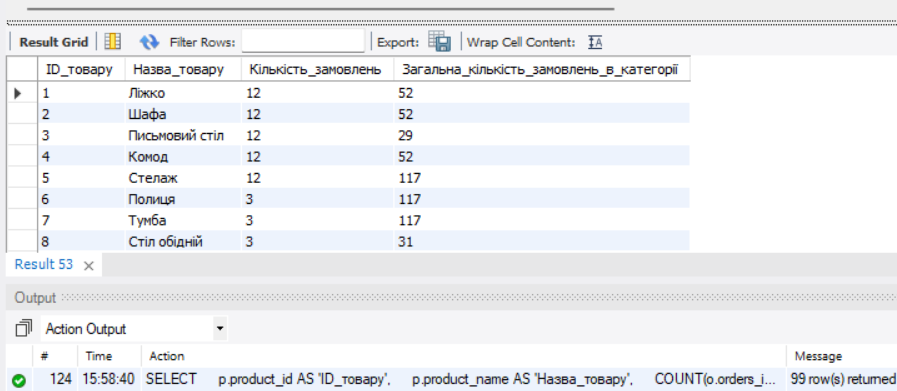


**Рис. 2.12. Підвищення ціни на 5%**

Також є запити аналітичного характери. На основі їх результатів можна аналізувати роботу магазину та на основі цього аналізу вподальшому вносити певні зміни за допомогою інструменталу адміністратора. До аналітичних запитів відносяться:

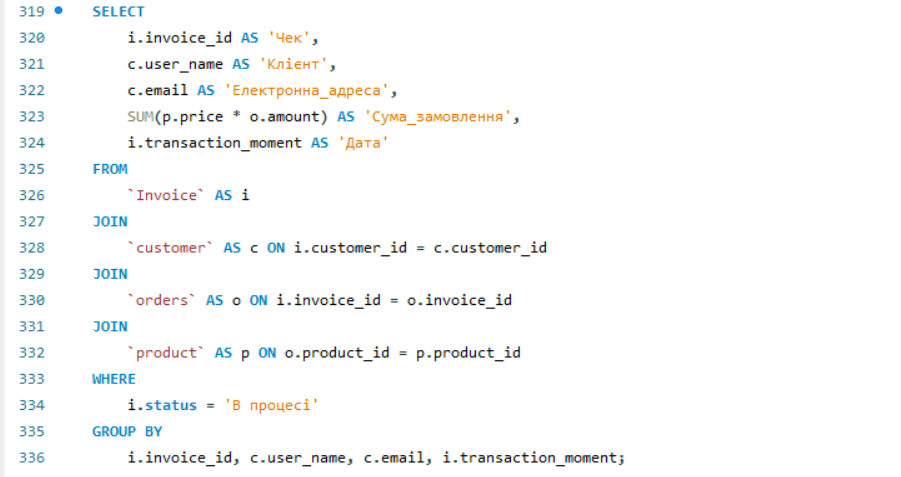
Відобразити всі товари, скільки разів вони були замовлені за певний період часу, а також загальну кількість замовлень в категорії, до якої відноситься товар:

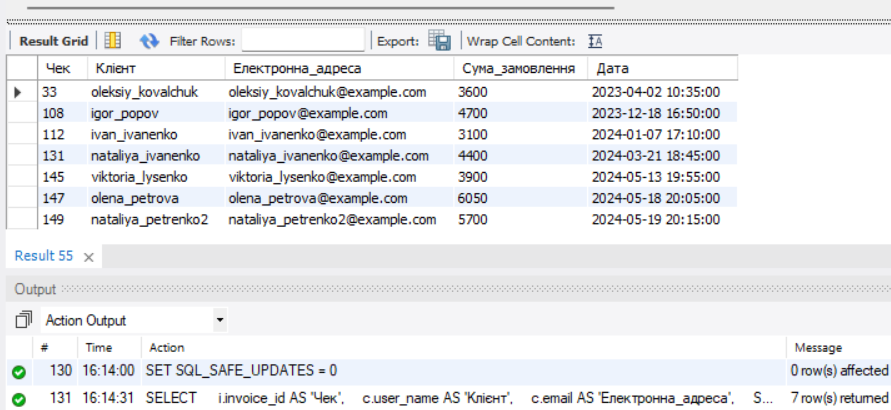




**Рис. 2.13. Всі товари (кількість замовлень товару та товарів тієї ж категорії)**

Відобразити неоплачені замовлення можна за допомогою запиту на рисунку 2.14. Окрім ID чеку, нікнейму клієнта, дати та суми замовлення відображається також електронна пошта клієнта, для забезпечення можливості зв’язатися з ним стосовно проблеми, що виникла.





**Рис. 2.14. Неоплачені замовлення**

Також за допомогою розроблених запитів можна переглянути список товарів, що ні разу не були куплені, чи список найпопулярніших позицій, рейтинг категорій, список клієнтів, що здійснили певну кількість покупок, чи товарів, що були продані певну кількість разів і так далі. Ці та інші запити, що реалізують решту технічних завдань, побудованих на основі діаграми прецедентів, розміщені у Додатку Б.

РОЗДІЛ 3. РОЗРОБКА ВЕБ-ДОДАТКУ ДЛЯ МАГАЗИНУ МЕБЛІВ

**3.1. Структура веб-сайту**

Для магазину без фізичної точки збуту функціональність його веб-сайту – ключова запорука успіху. Веб-сайт має бути візуально привабливим, щоб потенційному клієнту хотілося його листати, інтуїтивно зрозумілим, а також покривати всі функціональні вимоги, щоб задовільнити потреби клієнта. Для цього дуже важливо заздалегідь продумати й змоделювати структуру веб-сайту.

Мапа сайту – це візуальне представлення структури веб-сайту. Вона показує, як сторінки сайту пов'язані між собою, а також містить інформацію про ієрархію сторінок, їх зміст та метадані [8]. Цей графічний або текстовий образ структури веб-сайту дозволяє систематизувати та візуалізувати всі його розділи та сторінки.

Для схематичного зображення структури веб-сайту (мапи сайту) було використано програму draw.io. Основою сайту виступає Головна сторінка, вона ж Каталог, від неї розгалужуються три сторінки: Про Нас, Контакти та Кошик.

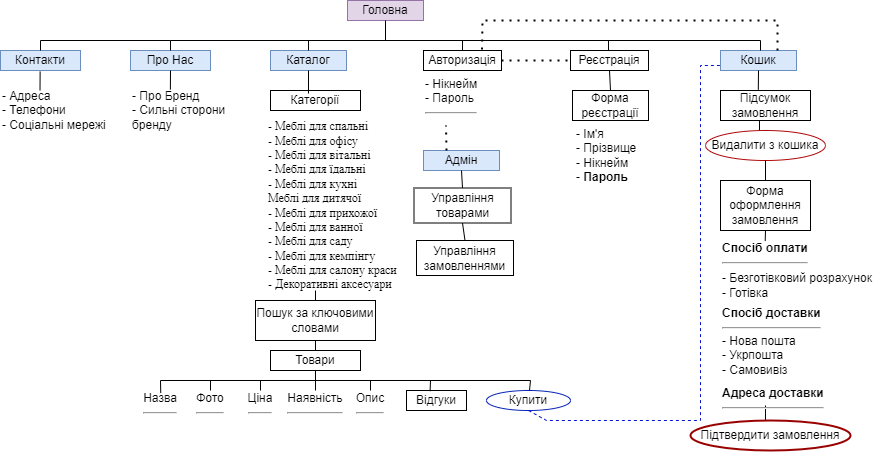
Сторінка Каталог, вона ж Головна, містить: Пошук за назвою, Пошук за Категоріями, а також власне Товари. Картка кожного товару містить: назву, фото, ціну товару, його опис, розміри відгуки на цей товар (з можливістю залишати їх самому) та кнопку Купити.

На сторінці Про Нас вказується інформація Про Компанію, сильні сторони бренду. На сторінці Контакти – контактна інформація магазину.

З шапки будь-якої сторінки можна потрапити на всі інші сторінки. Зокрема, поля «Авторизація/Реєстрація» ведуть на відповідні сторінки, після авторизації з’являється можливість здійснювати покупки та відкривається доступ до сторінки Кошик.

Кошик є окремою сторінкою сайту і, в свою чергу, складається з Підсумку замовлень та Форми оформлення замовлення, яка містить в собі: спосіб оплати (безготівковий розрахунок, готівка), спосіб доставки (Нова пошта, Укрпошта, сомовивіз зі складу), Адреси доставки. Завершується сторінка кнопкою «Підтвердити замовлення». Вся структура мапи спрямована на те, щоб користувач добрався до цієї кнопки та здійснив покупку.

Також веб-сайт містить сторінку Адміністратора, де забезпечується можливість керувати товарами (додавати, видаляти їх, змінювати ціни), а також перегляд інформації про клієнтів та їх замовлення. Доступ до сторінки забезпечується через Авторизацію.



**Рис. 3.1. Мапа веб-сайту магазину меблів**

*Джерело: Розроблено автором*

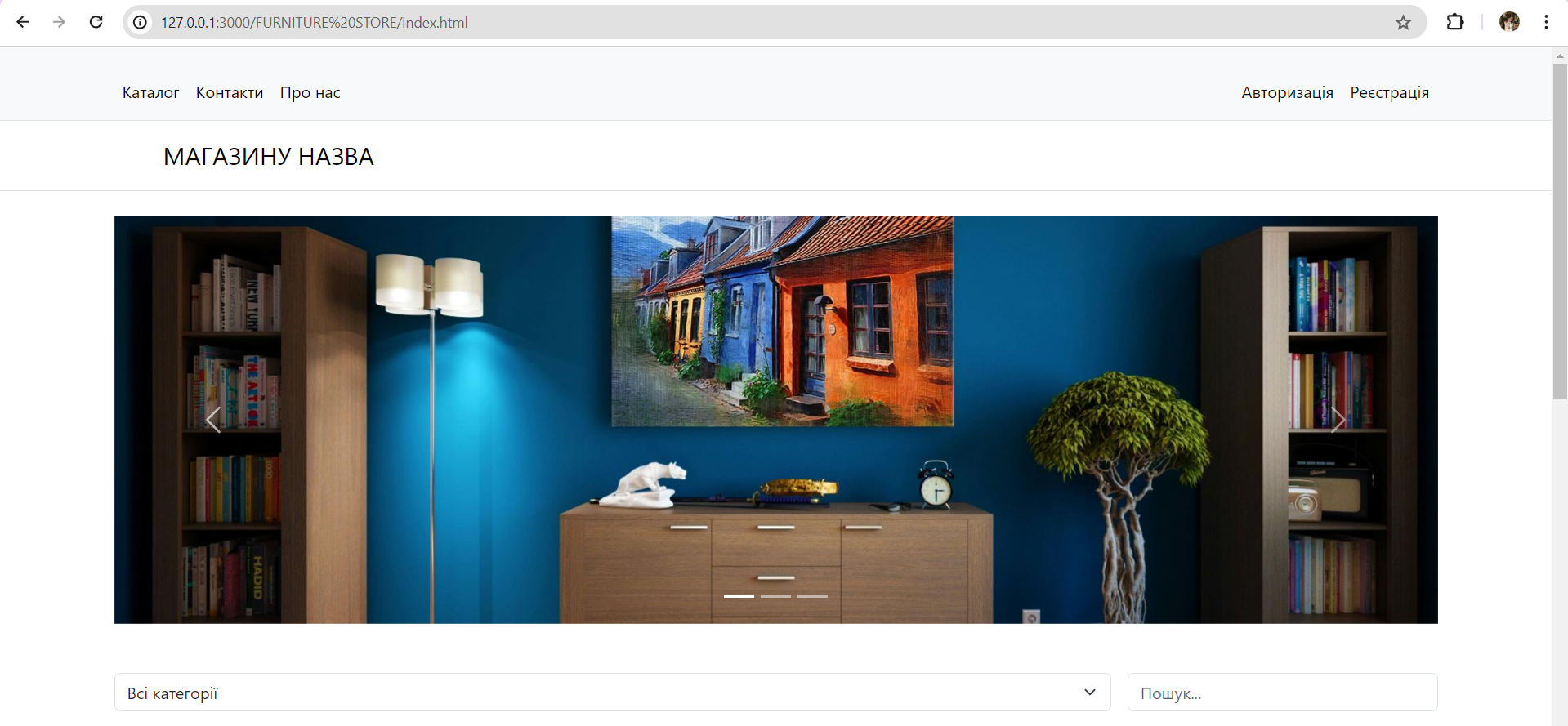
Розроблена мапа сайту для магазину меблів спрямована на те, щоб забезпечити користувачам якомога зручніший доступ до товарів (два види пошуку) та цим спрямувати покупців на здійснення покупки онлайн (натиснути кнопку «Підтвердити замовлення»). Головна мета сайту магазину – продавати, на ній і було зроблено акцент при розробці. Забезпечено три рівні доступу: незареєстрований користувач (гість), зареєстрований користувач (клієнт) та адміністратор, з різним функціоналом для кожного.

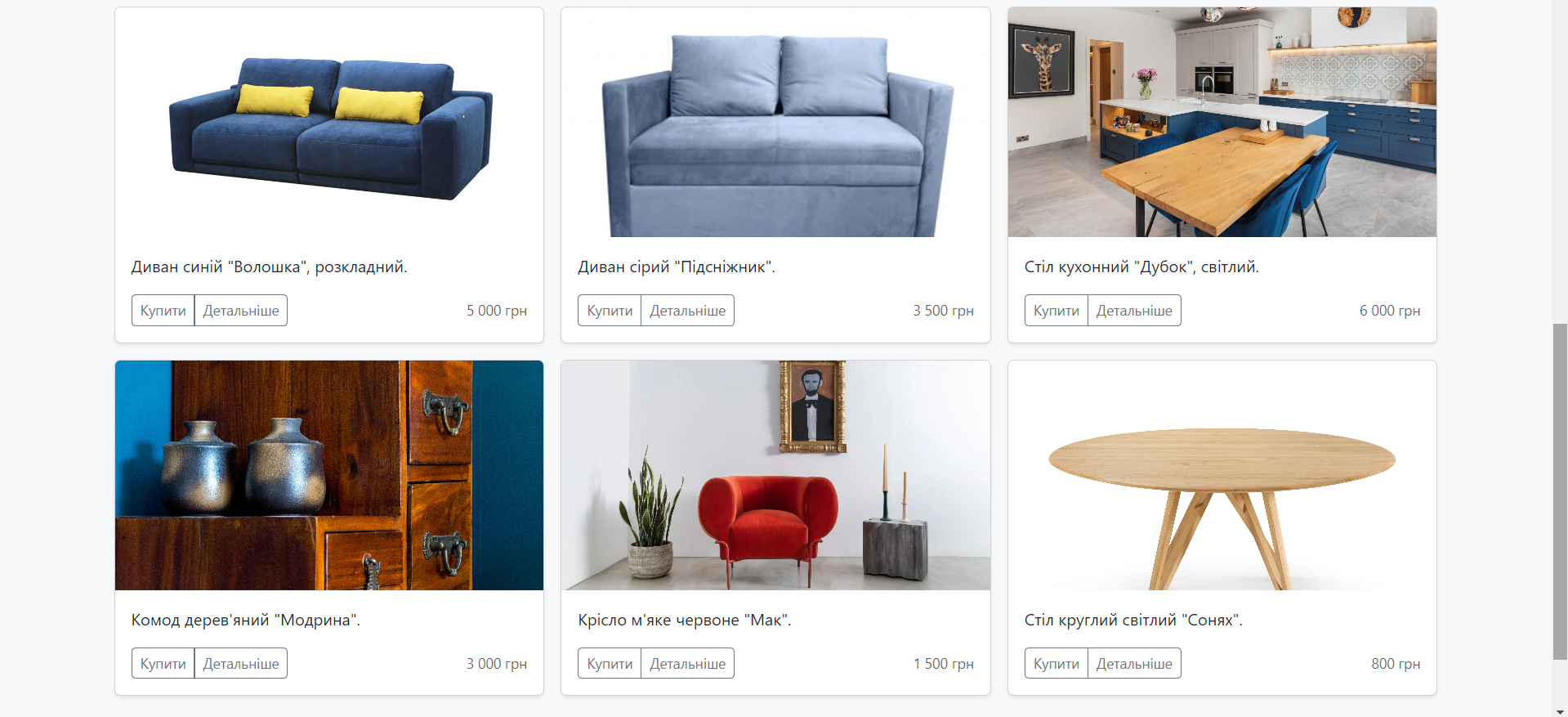
**3.2. Макети сторінок веб-сайту**

Згідно з попередньо-створеною мапою веб-сайту було розроблено макети всіх сторінок сайту, зі збереженням зв’язків та запланованої функціональності. Забезпечено три рівні доступу: гість, зареєстрований користувач, адміністратор. Гість має доступ до Каталогу, Карток товарів, Про нас та Контактів, а також можливість увійти чи зареєструватися. Після входу/реєстрації користувач додатково отримує доступ до Кошика, а, відповідно, і до покупок. При авторизації як адміністратор відкривається адмінпанель з різними можливостями для редагування вмісту сайту та доступу до даних користувачів та їх замовлень.

Головними кольорами обрано синій, сірий та білий з вкрапленнями оранжевого та жовтого – ці кольори добре поєднуються, адже знаходяться з протилежних сторін кола Іттена (синій та оранжевий, жовтий), або ж є нейтральними (сірий та білий) [9]. Гармонійно поєднані кольори забезпечують красиве зображення, до того ж, кожен колір має свій посил, який людина підсвідомо сприймає. Так, темно-синій колір асоціюється у нас зі стабільністю, довірою, спокоєм; червоний – інтенсивність, енергія; жовтий – увага, радість; блакитний – безтурботність. Людина радше натисне на блакитну кнопку, ніж на червону, оскільки червоний це небезпека, а щось блакитне небезпечним бути не може. Відповідно до цих правил було обрано кольори всіх кнопок та інших елементів на сайті.

Головна сторінка (каталог) (рис. 3.2.) однакова як для гостя, так і для зареєстрованого користувача. Її структура: одна колонка та п’ять рядків. Перший рядок – верхній колонтитул з посиланнями на всі інші сторінки та назвою магазину, другий – карусель з фото, третій – два види пошуку (за категорією та за назвою), четвертий – каталог товарів (він структурований окремо в три стовпці, де розміщені картки товарів), п’ятий – нижній колонтитул.

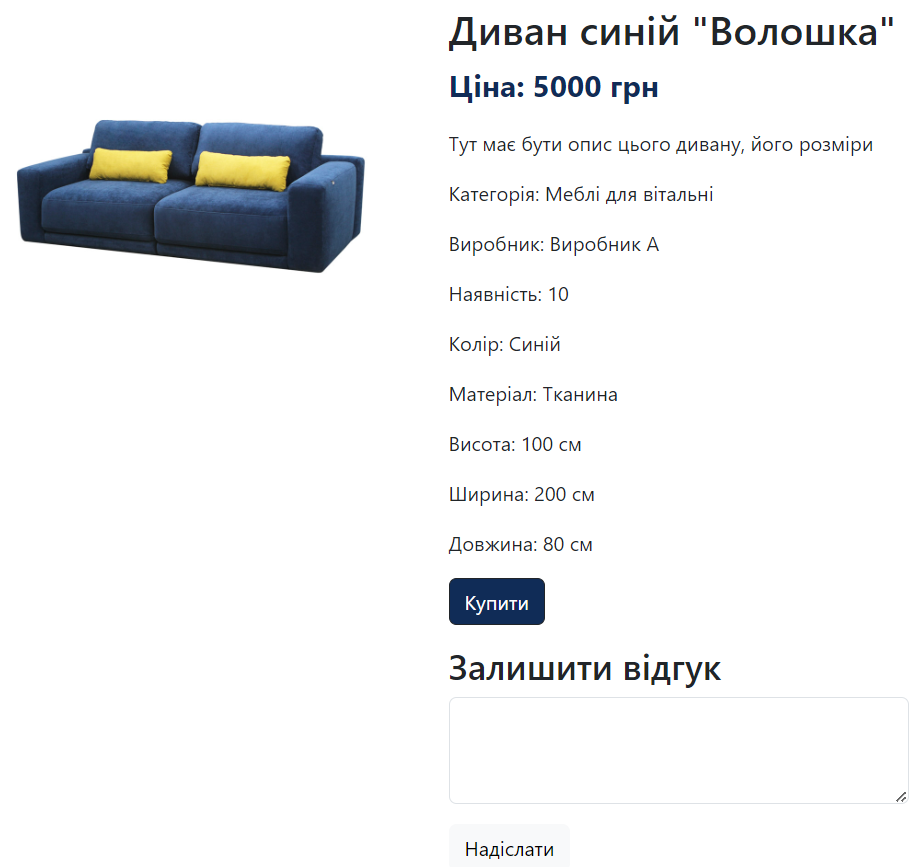






**Рис. 3.2. Головна сторінка**

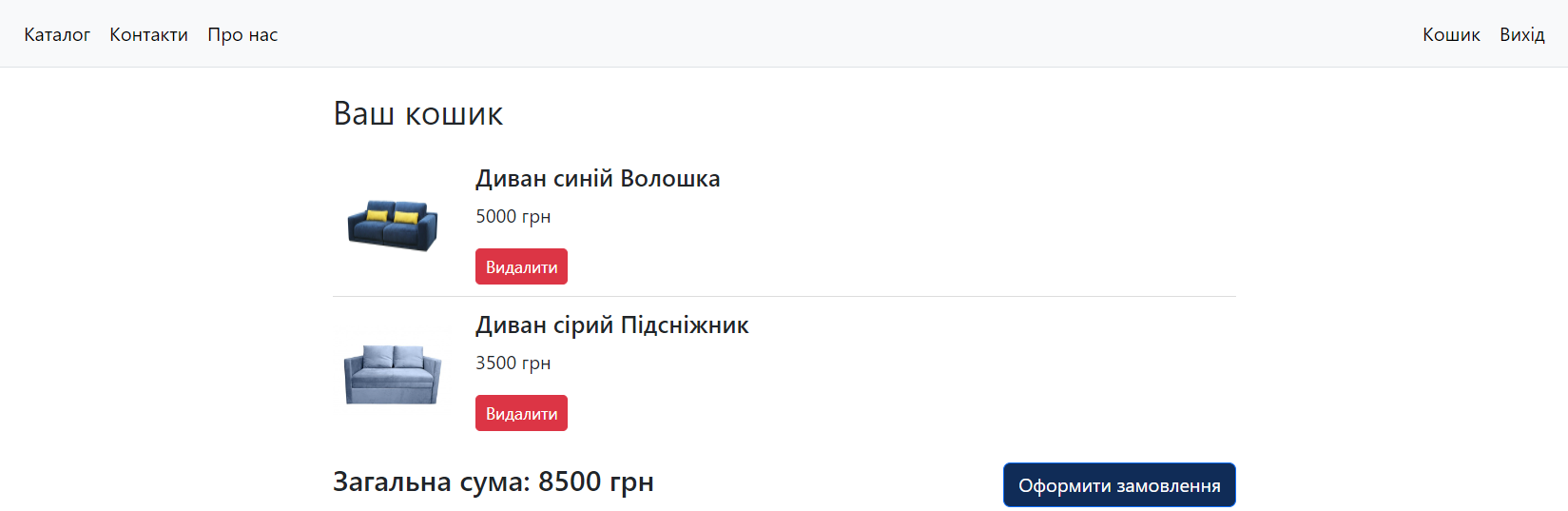
Картки товарів (рис. 3.3.) містять фото товару, його назву, ціну, опис та інші характеристики, такі як наявність, виробник, колір, матеріал та розміри (висота, ширина, довжина). Акцент зроблено на ціні товару, фото, його назві та кнопці «Купити», адже це те, на що клієнт має в першу чергу звернути увагу. Також є форма для можливості залишити відгук. Незареєстрований користувач не має кнопки «Купити» і не може залишати відгуки, всі ці можливості з’являться в нього після реєстрації.



**Рис. 3.3. Картка товару**

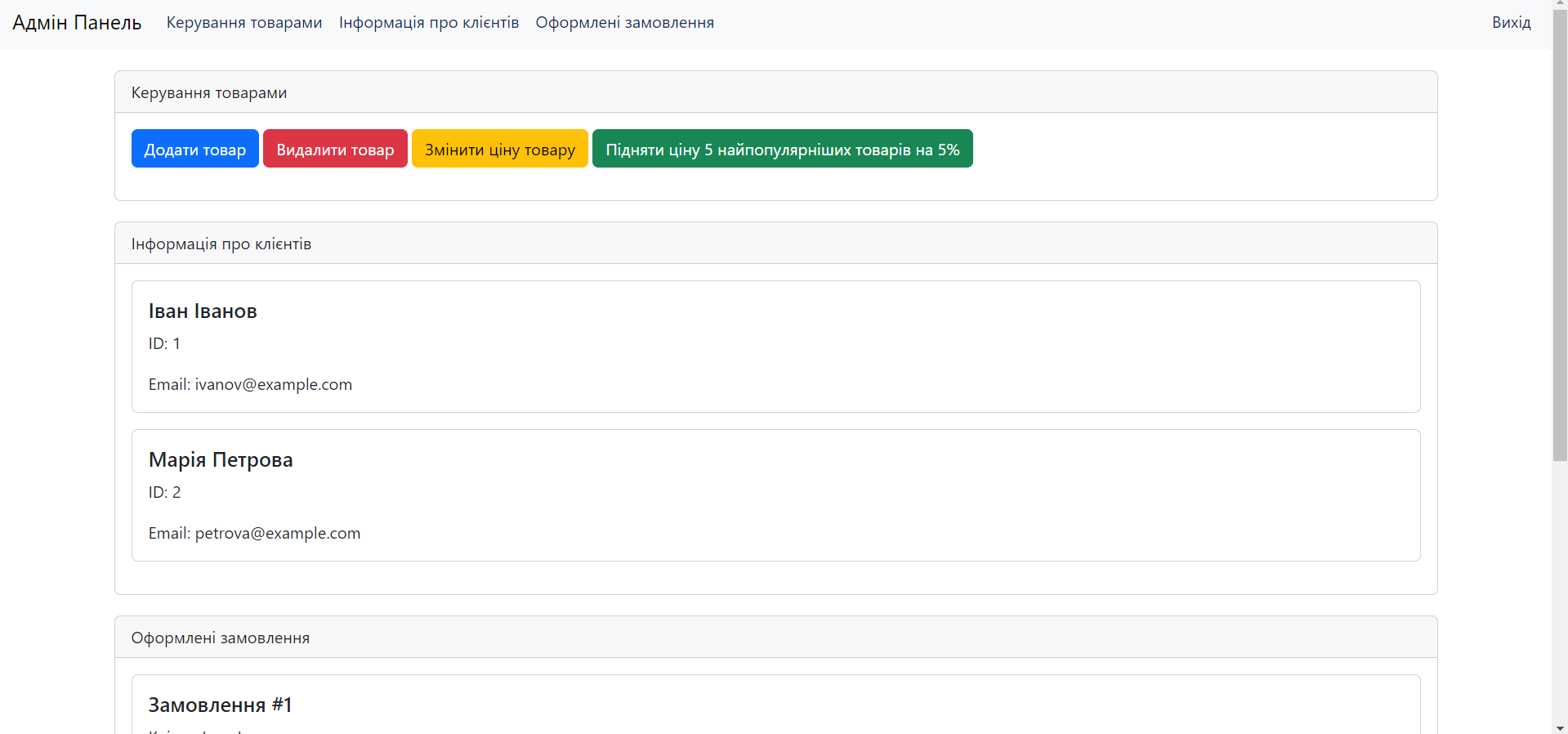
Сторінки Контакти (рис. В.1) та Про нас (рис. В.2) містять інформацію про бренд, його сильні сторони, а також контактну інформації: адресу складу, номери телефонів та соціальні мережі магазину. Поля для авторизації (рис. В.3) та реєстрації (рис. В.4) мають стандартний, звичний для користувача інтерфейс та виконують свою функцію.

В Кошику можна переглянути товари, додані в нього користувачем, за потреби видалити їх з Кошика, також відображається підсумкова сума замовлення. Кнопка «Оформити замовлення» викликає меню де користувач має зазначати необхідну інформацію та завершити покупку.



**Рис. 3.4. Кошик**

Після авторизації як адміністратор відкривається Адмінпанель. На ній можна: Додати товар, Видалити товар, Змінити ціну товару, Підняти ціну 5 найпопулярніших товарів на 5%, переглядати інформацію про клієнтів, переглядати оформлені замовлення.



**Рис. 3.5. Адмінпанель**

Загалом сайт складається з 16 сторінок, наповнених інформацією та оформлених в одному стилі, що повністю відповідають мапі веб-сайту, описаній в підрозділі 3.1.

**3.3. Програмування серверної частини**

У даному проєкті серверна частина ще не була розроблена, але веб-сайт налаштований для її можливого підключення. Специфічні модулі серверної частини можуть бути створені для обробки запитів, взаємодії з базою даних, обробки даних форм та відправлення відповідей клієнту. Всі поля на сайті, характеристики товарів, категорії й матеріали, а також сама структура сайту повністю відповідають базі даних, описаній у розділі 2. Тож, відповідно до потреб проєкту, до веб-сайту можна дописати серверну частину (backend).

**3.4. Програмування клієнтської частини**

Для розробки клієнтської частини веб-сайту (frontend), було використано стандарти розмітки сторінок HTML5, стилі СSS3 та JavaScript. Для збереження запланованої функціональності було реалізовано низку скиптів в поєднанні з HTML та СSS. До прикладу, пошук за категоріями реалізується за допомогою js скрипту (лістинг 3.1.), для його коректної роботи кожній картці товару присвоєно data-category=" ". Скрипт шукає ці категорії і фільтрує картки за ними. Якщо категорія збіглась, то елемент залишається видимим (стиль display установлюється на block), в іншому випадку, елемент приховується (стиль display установлюється на none).

Лістинг 3.1. Пошук за категоріями

document.addEventListener('DOMContentLoaded', function () {

const searchInput = document.getElementById('search-input');

const categorySelect = document.getElementById('category-select');

const itemsContainer = document.getElementById('items-container');

const items = Array.from(itemsContainer.getElementsByClassName('item'));

function filterItems() {

const query = searchInput.value.toLowerCase();

const selectedCategory = categorySelect.value;

items.forEach(item => {

const itemName = item.querySelector('.card-text').textContent.toLowerCase();

const itemCategory = item.getAttribute('data-category');

const matchesSearch = itemName.includes(query);

const matchesCategory = selectedCategory === 'all' || itemCategory === selectedCategory;

if (matchesSearch && matchesCategory) {

item.style.display = 'block';

} else {

item.style.display = 'none';

}

});

}

searchInput.addEventListener('input', filterItems);

categorySelect.addEventListener('change', filterItems);});

Пошук за назвою працює за схожим принципом (лістинг 3.2.). Всі елементи з класом "item" всередині контейнера itemsContainer отримуються за допомогою getElementsByClassName, та перетворюються в масив за допомогою Array.from. Додається подія "input" до поля введення пошуку. Кожного разу, коли користувач вводить текст, викликається функція-обробник. У функції-обробнику, введений текст отримується за допомогою searchInput.value.toLowerCase(). Потім для кожного елемента "item" у списку items виконується перевірка, чи містить текстовий контент елемента (назва предмету) введений запит. Елемент залишається видимим або приховується як і в попередньому випадку.

Лістинг 3.2. Пошук за назвою

document.addEventListener('DOMContentLoaded', function () {

const searchInput = document.getElementById('search-input');

const itemsContainer = document.getElementById('items-container');

const items = Array.from(itemsContainer.getElementsByClassName('item'));

searchInput.addEventListener('input', function () {

const query = searchInput.value.toLowerCase();

items.forEach(item => {

const itemName = item.querySelector('.card-text').textContent.toLowerCase();

if (itemName.includes(query)) {

item.style.display = 'block';

} else {

item.style.display = 'none';

}

});

});

});

Для коректної роботи Кошика теж використовується спеціальний код (лістинг 3.3). Користувач додає товар у кошик, натискаючи на кнопку "Купити" на картці товару. Після цього, за допомогою JavaScript, обраний товар додається до локального сховища браузера, де він зберігається у форматі JSON. На сторінці кошика JavaScript витягує збережені дані з локального сховища та генерує HTML-код для кожного товару у кошику, відображаючи його зображення, назву, ціну та поле для введення кількості.

Лістинг 3.3. Обробник подій для кнопки «Купити»

document.addEventListener('DOMContentLoaded', function() {

const buyButtons = document.querySelectorAll('.btn-dark[data-id]');

buyButtons.forEach(button => {

button.addEventListener('click', function() {

const productId = button.getAttribute('data-id');

const productName = document.getElementById('productName').innerText;

const productPrice = document.getElementById('productPrice').innerText;

let cart = JSON.parse(localStorage.getItem('cart')) || [];

cart.push({ id: productId, name: productName, price: productPrice });

localStorage.setItem('cart', JSON.stringify(cart));

displayCartItems();

alert('Товар додано до кошика');

});

});

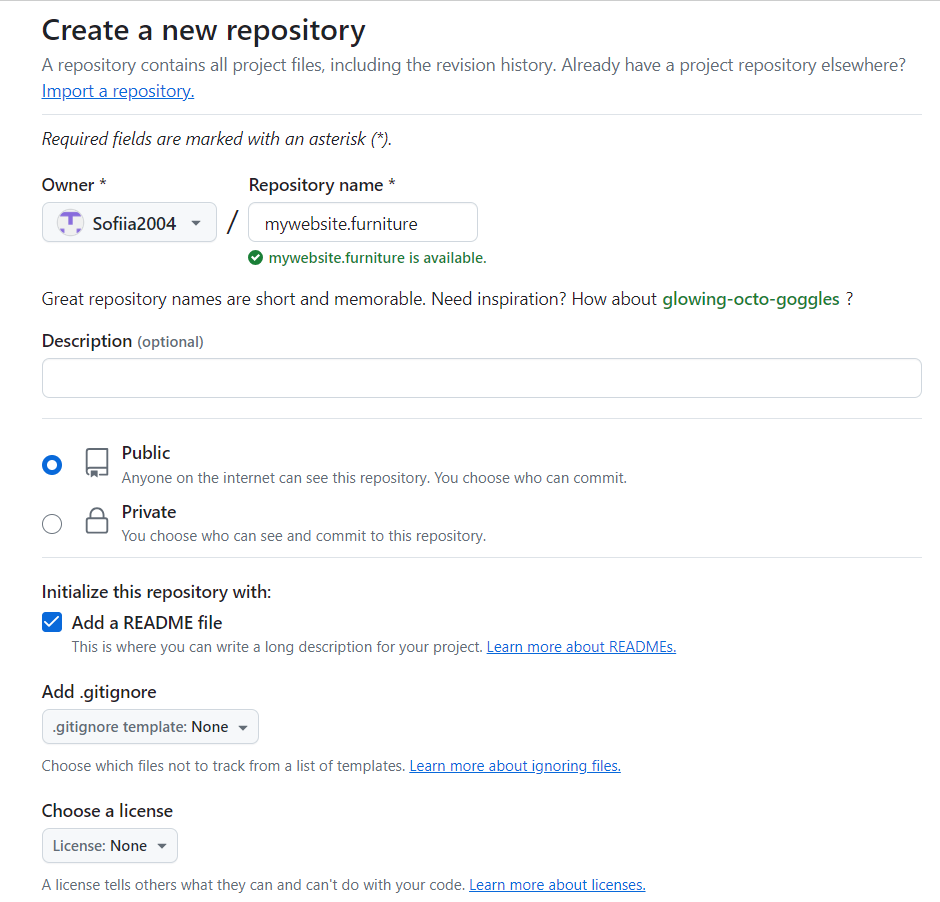
});

Для роботи всіх форм Адмінпанелі також розроблено скрипти, зокрема, функція додавання товару addProduct(), функція видалення товару deleteProductById(), функція для рендеренгу товарів renderProducts(), інформації про клієнтів renderCustomers(), замовлення renderOrders(), функція для зміни ціни товару changeProductPrice() та багато інших. Частина з них розташовується в файлі admin.js, частина у вигляді скриптів в admin.html.

**3.5. Розміщення веб-сайту на локальному віртуальному середовищі або в Інтернеті**

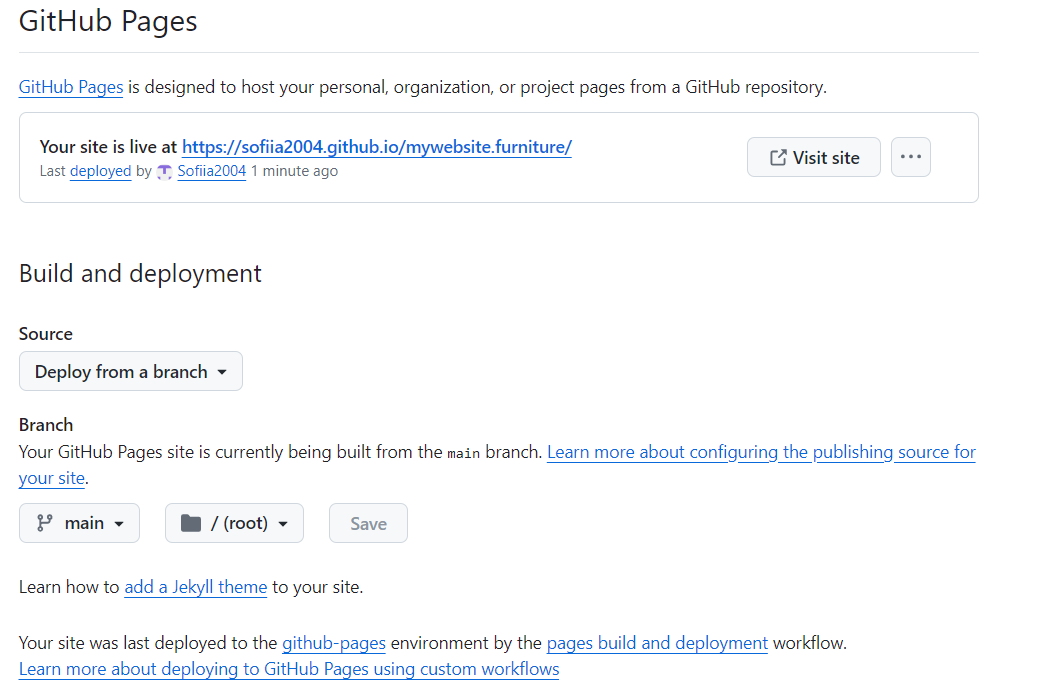
Для того, щоб забезпечити доступ до веб-сайту, його було розміщено в мережі за допомогою GitHub Pages. Таким чином можна захостити тільки статичні сайти, тобто такі, які складаються з HTML, CSS та JavaScript [10].

Перш за все необхідно створити новий репозиторій (рис. 3.6.), в даному випадку він називається «mywebsite.furniture» та вивантажити в нього всі файли, основні та додаткові, з яких складається код веб-сайту.



**Рис. 3.6. Створення репозиторію**

Після цього необхідно перейти в GitHub Pages (рис. 3.7.), обрати «Commit to main» та «Publish branch» і GitHub поверне робоче посилання на веб-сайт: <https://sofiia2004.github.io/mywebsite.furniture/>. Розміщення веб-сайту на GitHub Pages є простим та ефективним способом забезпечити доступність проекту в Інтернеті. До того ж, це безкоштовно.



**Рис. 3.7. GitHub Pages**

GitHub Pages дозволяє виконувати додаткові налаштування, такі як використання власного доменного імені, налаштування HTTPS для забезпечення безпечного з'єднання, та інтеграція з системами безперервної інтеграції та розгортання (CI/CD). Для додавання власного домену, потрібно створити файл CNAME у корені репозиторію та вказати у ньому бажане доменне ім'я. Потім у панелі управління доменом необхідно налаштувати відповідні DNS-записи.

Якщо сайт потребує специфічного дизайну, GitHub Pages пропонує широкий вибір тем, які можна застосувати до вашого проекту. Це дозволяє легко створити привабливий та професійний вигляд сайту без необхідності в глибоких знаннях веб-дизайну.

До цікавих та корисних можливостей GitHub відноситься також те, що він автоматично визначає, які мови було використано при написанні коду та висвітлює їх у відсоткових значеннях (рис. 3.8.). Веб-сайт меблевого магазину на 91% написаний на HTML, 8.7% коду це JavaScript та всього 0,3% – CSS.



**Рис. 3.8. Використані мови**

Розміщення веб-сайту за допомогою GitHub Pages є ефективним та зручним способом забезпечення доступу до статичних сайтів в Інтернеті. Це рішення підходить для невеликих проектів та сайтів, що не вимагають серверної логіки. За допомогою GitHub Pages можна швидко розгорнути сайт, налаштувати його під свої потреби та забезпечити доступність для користувачів з усього світу. Крім того, GitHub Pages інтегрується з іншими інструментами та сервісами, що дозволяє автоматизувати процеси розробки та розгортання, забезпечуючи високу якість та безпеку кінцевого продукту.

ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Що таке html? css.in.ua. Статті. URL: https://css.in.ua/article/shcho-take-css\_3 (дата звернення: 10.05.2024).
2. Що таке CSS і для чого потрібен веб-розробнику. Itstep. Блог. URL: https://kiev.itstep.org/blog/what-is-css-and-why-does-a-web-developer-need-it (дата звернення: 10.05.2024).
3. Що таке JavaScript. CASES. Статті. URL: https://cases.media/article/sho-take-javascript (дата звернення: 10.05.2024).
4. База даних MySQL. promoter.net. Статті. URL: https://promoter.net.ua/articles/baza-danix-mysql.html (дата звернення: 10.05.2024).
5. Типи даних в SQL. aCode. Уроки SQL. URL: https://acode.com.ua/data-types-sql/ (дата звернення: 15.05.2024).
6. Підтримка обмежень цілісності в БД. UA5.ORG. Бази даних. URL: https://ua5.org/database/2005-pidtrymka-obmezhen-czilisnosti-v-bd.html (дата звернення: 18.05.2024).
7. Тетяна Кучер. Об’єднуємо, фільтруємо, групуємо: SQL-скрипти для отримання вибірок. GENESIS. Блог. URL: https://www.gen.tech/post/sql-skrypty (дата звернення: 18.05.2024).
8. Ігор Генов. Що таке мапа сайту (файл Sitemap)? Inweb. Блог. URL: https://inweb.ua/blog/ua/what-is-sitemap/ (дата звернення: 20.05.2024)
9. Коло Іттена. Як правильно поєднувати кольори. STILEX. Статті. URL: https://stilex.ua/blogs/news/kolo-ittena-shcho-tse-i-yak-pravilno-poednuvati-kolory (дата звернення: 21.05.2024)
10. Мілан Всезнаючий. Як запустити свій сайт на github? Друкарня. Статті. URL: https://drukarnia.com.ua/articles/yak-zapustiti-svii-sait-na-github-jpZO\_ (дата звернення: 22.05.2024)

ДОДАТКИ

Додаток А

**Лістинг коду, що створює таблиці**

CREATE DATABASE furniture;

USE furniture;

CREATE TABLE IF NOT EXISTS `employee` (

`employee\_id` int AUTO\_INCREMENT NOT NULL,

`user\_name` varchar(255) NOT NULL,

`first\_name` varchar(30) NOT NULL,

`last\_name` varchar(30) NOT NULL,

`position` varchar(15) NOT NULL,

`employment\_date` date NOT NULL,

`manager\_id` int,

`bonus` float,

`gender` varchar(1) NOT NULL,

`birth\_date` date NOT NULL,

PRIMARY KEY (`employee\_id`)

);

CREATE TABLE IF NOT EXISTS `product` (

`product\_id` int NOT NULL,

`product\_name` varchar(50) NOT NULL,

`product\_description` varchar(800) NOT NULL,

`category` varchar(30) NOT NULL,

`manufacture` varchar(30) NOT NULL,

`amount` int NOT NULL,

`price` float NOT NULL,

`color` varchar(30) NOT NULL,

`material` varchar(30) NOT NULL,

`warehouse\_id` int NOT NULL,

`height` int NOT NULL,

`width` int NOT NULL,

`length` int NOT NULL,

PRIMARY KEY (`product\_id`)

);

CREATE TABLE IF NOT EXISTS `customer` (

`customer\_id` int AUTO\_INCREMENT NOT NULL,

`first\_name` varchar(30) NOT NULL,

`last\_name` varchar(30) NOT NULL,

`user\_name` varchar(30) NOT NULL,

`gender` varchar(1) NOT NULL,

`birth\_date` date NOT NULL,

`phone\_number` int NOT NULL,

`email` varchar(50) NOT NULL,

`discount` int,

`password` varchar(30) NOT NULL,

`city` varchar(30) NOT NULL,

PRIMARY KEY (`customer\_id`)

);

CREATE TABLE IF NOT EXISTS `orders` (

`orders\_id` int AUTO\_INCREMENT NOT NULL,

`product\_id` int NOT NULL,

`invoice\_id` int NOT NULL,

`order\_date` datetime NOT NULL,

`amount` int NOT NULL,

PRIMARY KEY (`orders\_id`)

);

CREATE TABLE IF NOT EXISTS `Invoice` (

`invoice\_id` int AUTO\_INCREMENT NOT NULL UNIQUE,

`employee\_id` int NOT NULL,

`customer\_id` int NOT NULL,

`payment\_method` int NOT NULL,

`transaction\_moment` datetime NOT NULL,

`status` varchar(30) NOT NULL,

PRIMARY KEY (`invoice\_id`)

);

CREATE TABLE IF NOT EXISTS `Warehouse` (

`warehouse\_id` int AUTO\_INCREMENT NOT NULL UNIQUE,

`warehouse\_name` varchar(50) NOT NULL,

`city` varchar(50) NOT NULL,

`street` varchar(40) NOT NULL,

`building\_nomber` varchar(10),

PRIMARY KEY (`warehouse\_id`)

);

-- вторинні ключі

ALTER TABLE `employee` ADD CONSTRAINT `employee\_fk6` FOREIGN KEY (`manager\_id`) REFERENCES `employee`(`employee\_id`);

ALTER TABLE `product` ADD CONSTRAINT `product\_fk9` FOREIGN KEY (`warehouse\_id`) REFERENCES `Warehouse`(`warehouse\_id`);

ALTER TABLE `orders` ADD CONSTRAINT `orders\_fk1` FOREIGN KEY (`product\_id`) REFERENCES `product`(`product\_id`);

ALTER TABLE `orders` ADD CONSTRAINT `orders\_fk2` FOREIGN KEY (`invoice\_id`) REFERENCES `Invoice`(`invoice\_id`);

ALTER TABLE `Invoice` ADD CONSTRAINT `Invoice\_fk1` FOREIGN KEY (`employee\_id`) REFERENCES `employee`(`employee\_id`);

ALTER TABLE `Invoice` ADD CONSTRAINT `Invoice\_fk2` FOREIGN KEY (`customer\_id`) REFERENCES `customer`(`customer\_id`);

Додаток Б.

Відобразити всі товари та скільки разів вони були замовлені за певний період часу, а також загальну кількість замовлень в категорії, до якої відноситься товар:

SELECT CONCAT(p.product\_id, ' - ', p.product\_name, ' - ', COUNT(o.orders\_id), ' - ', (SELECT COUNT(\*) FROM orders o2 JOIN product p2 ON o2.product\_id = p2.product\_id WHERE p2.category = p.category)) AS 'Результат' FROM `product` AS p LEFT JOIN `orders` AS o ON p.product\_id = o.product\_id WHERE o.order\_date BETWEEN '2023-01-02 16:00:00' AND '2024-05-19 20:20:00' GROUP BY p.product\_id, p.product\_name;

Показати постачальників та кількість різних категорій продуктів, які вони постачають з вказанням цих категорій:

SELECT CONCAT(p.manufacture, ' - ', GROUP\_CONCAT(DISTINCT p.category ORDER BY p.category ASC), ' - ', COUNT(DISTINCT p.category)) AS 'Результат' FROM `product` AS p GROUP BY p.manufacture;

Список товарів, які не були куплені ні разу, з вказанням виробника та ціни товару.

SELECT p.product\_name AS 'Назва', p.manufacture AS 'Виробник', p.price AS 'Ціна' FROM `product` AS p LEFT JOIN `orders` AS o ON p.product\_id = o.product\_id WHERE o.orders\_id IS NULL;

Статистика продажу товарів різних виробників за роками:

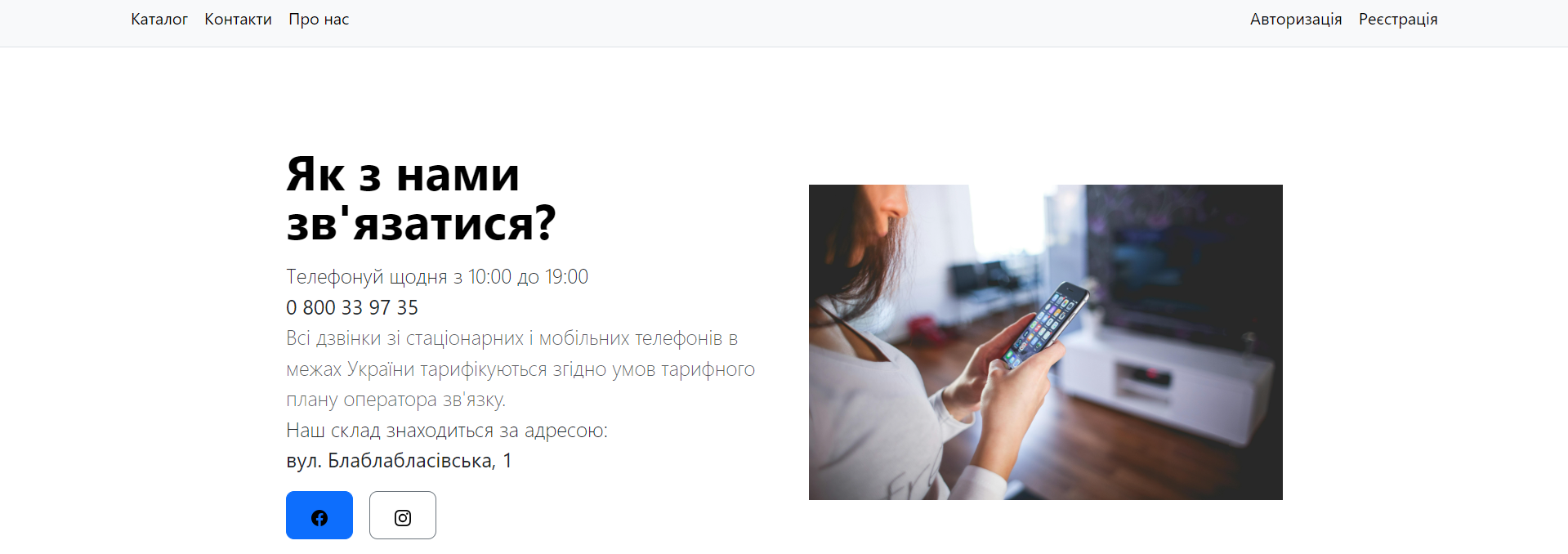
SELECT p.manufacture AS 'Виробник',YEAR(i.transaction\_moment) AS 'Рік',COUNT(DISTINCT o.orders\_id) AS 'Кількість\_замовлень',SUM(p.price \* o.amount) AS 'Загальна\_вартість\_замовлень' FROM `product` AS p LEFT JOIN `orders` AS o ON p.product\_id = o.product\_id LEFT JOIN `Invoice` AS i ON o.invoice\_id = i.invoice\_id GROUP BY p.manufacture, YEAR(i.transaction\_moment)ORDER BY p.manufacture, YEAR(i.transaction\_moment);

Список товарів, що були куплені більше певної кількості разів:

SELECT p.product\_id AS 'ID', p.product\_name AS 'Назва', SUM(o.amount) AS 'Кількість\_проданих\_одиниць' FROM `product` AS p

JOIN `orders` AS o ON p.product\_id = o.product\_id GROUP BY p.product\_id, p.product\_name HAVING SUM(o.amount) > 5 ORDER BY SUM(o.amount) DESC;

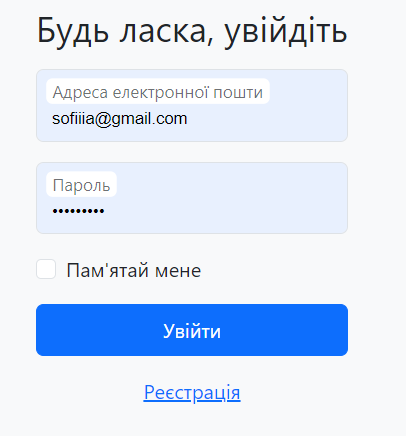
Додаток В.



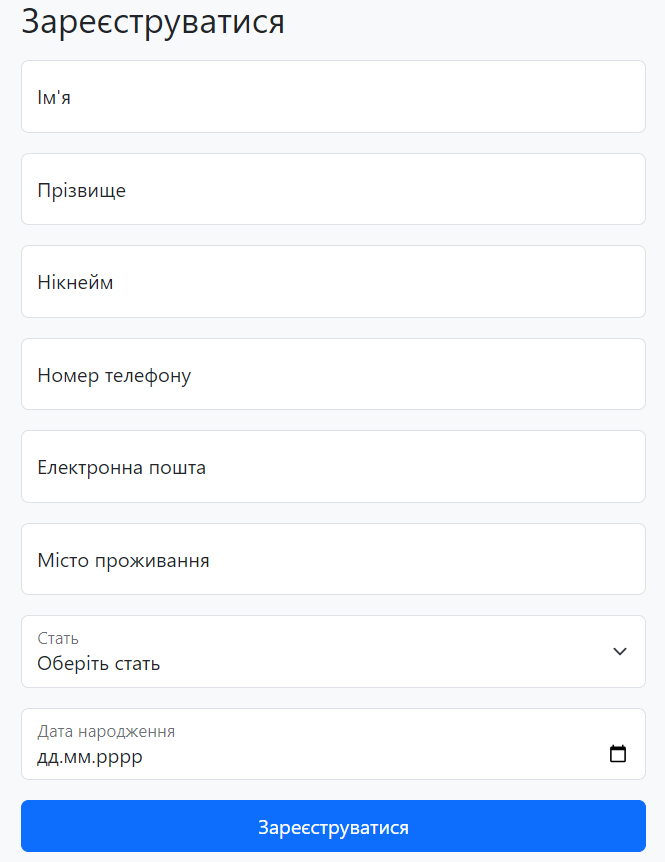
**Рис. В.1. Контакти**



**Рис. В.2. Про нас**



**Рис. В.3. Авторизація**



**Рис. В.4. Реєстрація**