ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ І. І. МЕЧНИКОВА

(повне найменування закладу вищої освіти)

Факультет математики, фізики та інформаційних технологій

(повне найменування факультету)

Кафедра інформаційних технологій

(повна назва кафедри)

Кваліфікаційна робота

на здобуття ступеня вищої освіти «Бакалавр»

«Розробка веб-ресурсу для онлайн-придбання одягу на базі фреймворку Django»

(тема кваліфікаційної роботи українською мовою)

«Development of an Online Clothing Purchase Web Resource Based on the Django Framework»

(тема кваліфікаційної роботи англійською мовою)

Виконав: здобувач денної форми навчання

спеціальності 122 Комп'ютерні науки

(код, назва спеціальності)

Освітня програма Комп'ютерні науки

(назва)

Бутяк Гліб Георгійович

(прізвище, ім`я, по-батькові здобувача)

Керівник к.т.н., доцент Фразе-Фразенко О.О.

(науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент к.т.н., Домаскін О.М.

(науковий ступінь, вчене звання, прізвище, ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
Інформаційних технологій .

№ 1 від 09 червня 2024 р.

Завідувачка кафедри

(підпис) КАЗАКОВА Надія .

(прізвище,ім`я)

Захищено на засіданні ЕК № 13 .
протокол № 12 від 20 червня 2024 р.

Оцінка відмінно / A / 90 .

(за національною шкалою/шкалою ECTS/ бали)

Голова ЕК

(підпис) КОПИЧЕНКО Іван .

(прізвище,ім`я)

Одеса 2024

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

HTML – HyperText Markup Language – мова гіпертекстової розмітки;

CSS – Cascading Style Sheets – каскадні таблиці стилів;

JS – JavaScript – мова програмування JavaScript;

AJAX – Asynchronous JavaScript and XML – асинхронний JavaScript і XML;

DRY – Don't Repeat Yourself – не повторюй себе;

XSS – Cross-Site Scripting – міжсайтовий скриптинг;

CSRF – Cross-Site Request Forgery – міжсайтове підроблення запитів;

MVC – Model-View-Controller – модель-вид-контролер;

MVT – Model-View-Template – модель-вид-шаблон;

БД – База Даних;

СУБД – Система Управління Базами Даних;

UI – User Interface – користувацький інтерфейс;

UX – User Experience – користувацький досвід;

ORM – Object-Relational Mapping – об'єктно-реляційне відображення;

CRUD – Create, Read, Update, Delete – створення, читання, оновлення, видалення;

JSON – JavaScript Object Notation – формат обміну даними між клієнтом і сервером;

HTTP – HyperText Transfer Protocol – протокол передачі гіпертексту;

URL – Uniform Resource Locator – уніфікований покажчик ресурсу;

WCAG – Web Content Accessibility Guidelines – керівництво з доступності вебконтенту;

SADT – Structured Analysis and Design Technique – метод структурного аналізу і проектування;

BPMN – Business Process Model and Notation – мова моделювання бізнес-процесів.

ВСТУП

У сучасному світі, ринок ексклюзивного одягу постійно розширюється і набирає популярності серед споживачів. Сьогодні, коли мода та індивідуальність стають важливими елементами самовираження, все більше людей шукають унікальні та обмежені в тиражі речі, що підкреслять їх. Такі споживачі прагнуть виділитися з натовпу, демонструючи свій особистий стиль через ексклюзивні предмети гардеробу від відомих брендів. Це можуть бути не тільки молоді люди, але й дорослі споживачі, які цінують якість та унікальність. Тому створення веб-ресурсу для продажу ексклюзивного одягу є надзвичайно актуальним та відповідає сучасним потребам ринку.

Україна не є винятком з глобальних трендів. Попит на ексклюзивний одяг в Україні також зростає. Споживачі шукають унікальні предмети гардеробу, що допомагають виразити їхню індивідуальність та відповідають останнім модним тенденціям. Український ринок моди активно розвивається, і з'являються нові бренди та дизайнери, які пропонують свої унікальні вироби.

Водночас, український ринок стикається з деякими викликами. Це, зокрема, низька купівельна спроможність населення порівняно з країнами Західної Європи та США, а також обмежена доступність деяких брендів та моделей на внутрішньому ринку. Тим не менш, з розвитком інтернет-торгівлі, українські споживачі отримують можливість купувати ексклюзивний одяг, що сприяє зростанню інтересу до цього сегменту.

Основною метою даного дипломного проекту є розробка та впровадження функціонального веб-сайту для продажу ексклюзивного одягу. Для досягнення цієї мети необхідно вирішити певні завдання, серед яких:

1. Аналіз ринку та визначення вимог: Дослідження тенденцій ринку ексклюзивного одягу, вивчення потреб цільової аудиторії та аналіз конкурентів.

2. Вибір відповідних технологій: Розгляд та обґрунтування вибору технологій, які будуть використовуватися для реалізації проекту, зокрема

Django для серверної частини, HTML, CSS, JavaScript, jQuery та Bootstrap 5 для клієнтської частини, а також PostgreSQL для управління базами даних.

3. Проектування архітектури системи: Розробка архітектурних рішень, що забезпечать стабільну роботу системи, високу продуктивність та безпеку.

4. Розробка функціональних компонентів: Реалізація основних модулів сайту, включаючи інтерфейс користувача, модулі для управління товарами та замовленнями, систему керування контентом (CMS) та інтеграцію з платіжними системами.

5. Впровадження та підтримка системи: Запуск веб-сайту в експлуатацію та забезпечення його подальшої підтримки та оновлень.

Тож в результаті виконання проекту буде створено функціональний веб-сайт для продажу ексклюзивного одягу, який відповідає сучасним стандартам веб-розробки та задовольняє потреби користувачів у пошуку та придбанні речей. Сайт буде забезпечувати швидку та безпечну обробку замовлень та зручний інтерфейс користувача.

Кваліфікаційна робота містить 64 сторінки, 46 рисунків, 3 таблиці, 11 джерел посилань.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАВДАННЯ

1.1 Опис сфери діяльності, поточні тенденції та потреби ринку

Ринок ексклюзивного одягу характеризується продажем обмеженої кількості унікальних виробів від відомих брендів. Такий одяг часто відрізняється високою якістю, оригінальним дизайном та складністю виробництва, що робить його привабливим для споживачів, які цінують індивідуальність та статусність.

Поточні тенденції:

1. Інтерес до індивідуальності: Споживачі все більше цінують можливість виразити свою індивідуальність через одяг. Вони шукають унікальні предмети гардеробу, які допомагають підкреслити їхній особистий стиль.

2. Екологічна свідомість: Зростає попит на екологічно чисті та стійкі продукти. Це стосується і ексклюзивного одягу, який часто виготовляється з високоякісних екологічних матеріалів.

3. Цифровізація та онлайн-продажі: Розвиток технологій сприяє зростанню інтернет-торгівлі. Споживачі все частіше купують одяг через онлайн-магазини, що робить важливим створення зручних та функціональних веб-ресурсів.

4. Колаборації з відомими дизайнерами: Відомі бренди все частіше співпрацюють з дизайнерами, створюючи лімітовані колекції, що підвищує попит на ексклюзивні речі.

Потреби ринку:

1. Унікальність та обмеженість: Споживачі шукають одяг, який не доступний у масовому продажу, що дозволяє їм виділятися з натовпу.

2. Якість та довговічність: Висока якість матеріалів та пошиття є важливими факторами при виборі ексклюзивного одягу.

3. Зручність покупки: Інтернет-магазини повинні забезпечувати зручний інтерфейс, швидку обробку замовлень та надійну доставку.

4. Персоналізація: Покупці очікують індивідуальний підхід, рекомендації на основі їхніх попередніх покупок та інтересів.

1.2 Огляд та аналіз існуючих систем

На ринку існує багато платформ, що пропонують продаж ексклюзивного одягу. Для аналізу було обрано сайти, магазинів, які працюють в Україні. Деякі з них, добре відомі серед молоді, та мають схожі між собою характеристики.

Було розглянути сайти магазинів:

- Agnostic[3];
- Original Store[4];
- Del Garda[5];

Agnostic – є українським онлайн-магазином, що спеціалізується на продажі ексклюзивного одягу, взуття та аксесуарів. Також в них є шоу-рум у Києві, куди можуть прийти клієнти і подивитися і обрати товар наживо. Загалом магазин орієнтований на молодь, яка цінує унікальність та якість. Agnostic пропонує товари від відомих брендів, також в них бувають різні колаборації з українськими дизайнерами. Тож особливостями даного магазину можна виділити:

1. Унікальний асортимент: Широкий вибір ексклюзивних товарів, які важко знайти в інших магазинах.

2. Обмежені колекції: Товари надходять до магазину в обмежених кількостях, що підвищує їхню цінність та привабливість.

3. Якісні матеріали: Використання високоякісних матеріалів, для виготовлення одягу та аксесуарів, що стосується саме їх колекцій.

Також розглянемо головну сторінку їх онлайн-магазину на рисунку 1.1.

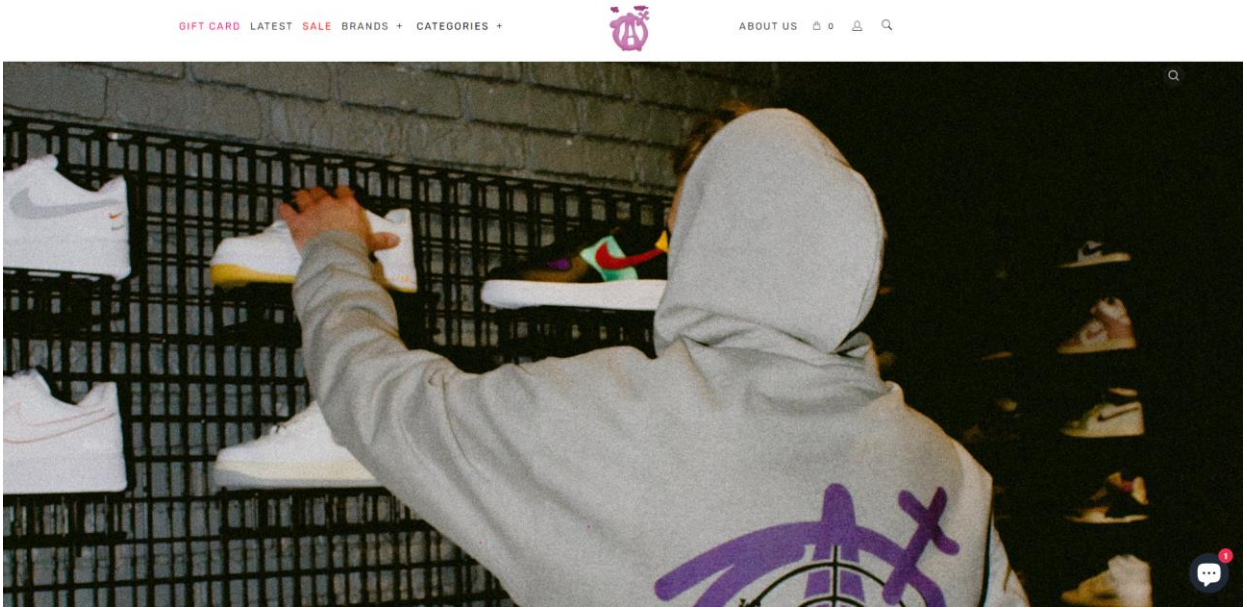


Рисунок 1.1 – Головна сторінка магазину Agnostic

Головна сторінка магазину має сучасний дизайн з акцентом на візуальну складову. Вона включає велику карусель фотографій, яка через деякий час змінюється. Зручна навігація, дозволяє швидко знайти необхідний розділ або товар, тому що навігаційна панель не переповнена зайвими вкладками. Завдяки меню з основними категоріями та брендами: можна одразу легко знайти, що саме потрібно клієнту. Головна сторінка також має секції з останніми надходженнями та спеціальними пропозиціями. З цікавого можна виділити, що деякий текст залишається на англійській, але, що стосується товарів йде українською. З мінусів можна виділити, що не має повної локалізації на англійську мову, хоча в більшості, магазин орієнтований саме на українського покупця.

Original Store – також відомий український онлайн-магазин одягу для молоді, який спеціалізується на продажу ексклюзивного одягу та взуття від світових брендів. Магазин позиціонує себе як постачальник оригінальних та високоякісних товарів. Особливостями магазину можна виділити:

1. Офіційні дистриб’ютори: Магазин працює напряму з деякими виробниками та офіційними дистриб’юторами.

2. Широкий вибір брендів: Магазин надає доволі великій вибір брендів для покупця, що привертає увагу .

3. Сервіс: Проаналізувавши коментарі у мережі. Можна сказати, що магазин має високий рівень обслуговування клієнтів.

Розглянемо головну сторінку сайту на рисунку 1.2.

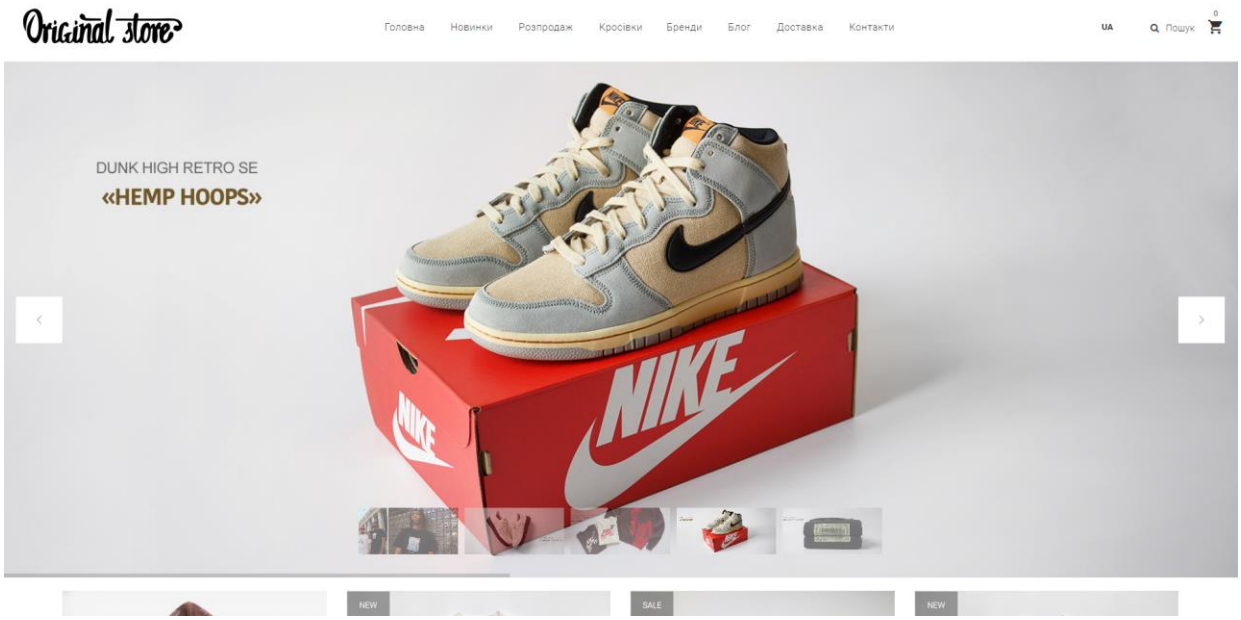


Рисунок 1.2 – Головна сторінка сайту Original Store

Головна сторінка виконана у мінімалістичному стилі з акцентом на фотографії. Використовується велика кількість білого простору для підкреслення продуктів. Якість фотографій та різноманіття, яке використовуються на сайті, є дуже якісними і добре підібраними, що особливо привертає увагу клієнтів. Навігація інтуїтивно зрозуміла, меню з основними категоріями товарів, а також зручні фільтри для швидкого пошуку. Навігаційна панель включає секції з новинками і розпродажами. Дуже вдалі анімації, при наведенні миші на товар. За інформацією з їх сайту на 2024 рік, їх офлайн магазин не працює і наразі продажі ведуться тільки через сайт.

Del Garda – це інтернет-магазин, який також орієнтований на одежу відомих брендів. Їх цільова аудиторія це клієнти, які полюбляють більш

стриманий стиль, але при цьому люблять якісну і ексклюзивну одежу. Особливостями є:

1. Унікальний підбір: Магазин пропонує ретельно відібрані товари, що відповідають останнім модним тенденціям.

2. Обмежені колекції: Товари випускаються у обмеженій кількості і змінюються кожен сезон на нові.

3. Підтримка клієнтів: Клієнту можуть надати консультації щодо вибору розмірів та стилю.

Подивимось на рисунок 1.3. на якому зображено головну сторінку їх сайту.

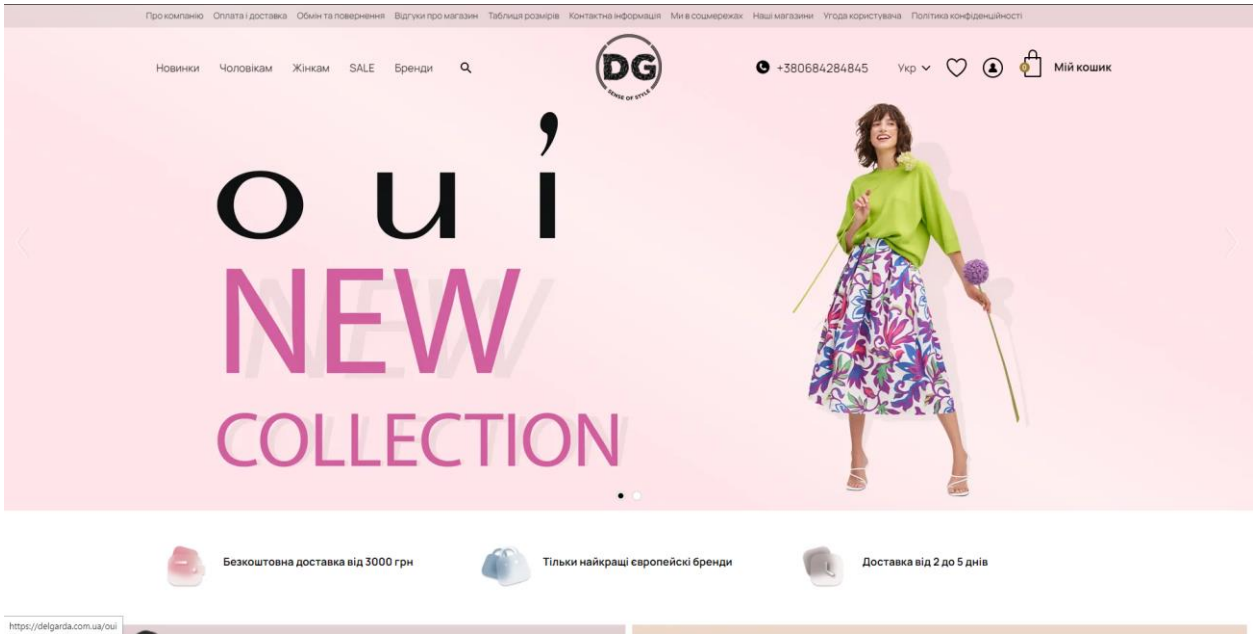


Рисунок 1.3 – Головна сторінка інтернет-магазину Del Garda

Головна сторінка має дизайн з великими банерами, які показують їх нові сезонні колекції. Кольорова гамма відповідає мотивам колекції. Подвійна навігаційна панель, доволі рідкісне явище і загалом вона трохи збиває з пантелику, акцентуючи погляд не на основній панелі. При цьому основа

навігаційна панель, зрозуміла. Головна сторінка також, як і всі сайти до цього, акцентує увагу на розпродажі та новинках.

Підбиваючи підсумки, кожен з магазинів пропонує широкий вибір ексклюзивного одягу, взуття та аксесуарів. Важливо забезпечити різноманітність товарів на сайті, щоб конкурувати з цими магазинами. Також майже всі конкуренти мають зручний інтерфейс та сучасний дизайн, що забезпечує позитивний досвід для користувачів. Важливими аспектами є виділення спеціальних пропозицій та швидка обробка замовлень. Тому при створенні власного сайту, я маю врахувати всі аспекти і повинен забезпечити не менш високий рівень підтримки клієнтів.

1.3 Постановка завдань, які необхідно вирішити під час розробки сайту

Проаналізувавши конкурентів, необхідно поставити ключові завдання для власного проекту. Завдання можна розділити на технічні, функціональні. Організаційний етап також можна віднести до постановки завдань, але це є дипломною роботою, де проект створюється однією людиною, а не командою.

Технічне завдання, має містити перелік чітких вимог до проекту. Якщо детально і правильно скласти цей документ, то результат буде більш якісним, а цілі зрозумілішими. До технічного завдання входять:

Вибір технологічного стеку: Треба визначити найбільш підходящі технології для розробки клієнтської та серверної частин сайту.

Створення архітектури системи: Розробка архітектури веб-сайту, забезпечить стабільність, масштабованість та зручність підтримки майбутнього проекту. Вона повинна враховувати взаємодію між клієнтською та серверною частинами та базою даних.

Проектування бази даних: Створення структури бази даних, що відповідає потребам інтернет-магазину та оптимально підходить під технології проекту.

Забезпечення безпеки даних: Насьогодні в нашому середовищі, це один з найважливіших аспектів який потрібно враховувати при створенні веб-сайту. Захист персональних даних користувачів та забезпечення безпеки фінансових транзакцій.

Оптимізація продуктивності: Оптимізація продуктивності сайту, потрібна для для забезпечення швидкого завантаження сторінок та обробки запитів.

Забезпечення адаптивності: Забезпечити веб-сайт адаптивним дизайном є не менш важливим аспектом. Велика кількість замовлень і покупок через мережу інтернет відбувається, саме через смартфони. Сайт який буде коректно відображатися на різних пристроях, забезпечить збільшення охоплення аудиторії та кількість продажів.

Функціональні завдання – це конкретні задачі, які необхідно створити та вирішити, для забезпечення необхідної функціональності веб-сайту. Це реалізація можливих функцій, сервісів які є доступними для користувача та адміністратора системи. До цього списку входять:

Реалізація інтерфейсу користувача: Розробка зручного та інтуїтивно зрозумілого інтерфейсу, який забезпечить легкий доступ до всіх розділів сайту. Інтерфейс повинен включати сторінки каталогу товарів, детальної інформації про товар, кошик, оформлення замовлення та інші необхідні елементи.

Розробка системи управління контентом: Це необхідна та важлива можливість, щоб забезпечити адміністрування контенту на сайті. Має включаючи додавання, редагування та видалення товарів, управління замовленнями та користувачами.

Система пошуку та фільтрації: Реалізація ефективної системи пошуку та фільтрації товарів за різними. Все це робить сайт більш комфортним та допомагає користувачу швидко підібрати та знайти необхідний товар.

2 ВИБІР АРХІТЕКТУРИ СИСТЕМИ ТА ПРОГРАМНИХ ЗАСОБІВ РЕАЛІЗАЦІЇ

2.1 Вибір Django як фреймворк Python

Фреймворк – це готовий набір бібліотек та інструментів, що надає структурні та базові компоненти для розробки. Завдяки фреймворкам, розробники можуть зосередитися на функціях і логіці створюваного додатку, замість того, щоб витрачати час на написання стандартних функцій. Фреймворки надають змогу вирішувати поширені задачі, що спрощує створення додатків, спрощує їх підтримку та забезпечує якість проекту.

Django[2] – це потужний та високорівневий веб-фреймворк написаний мовою програмування Python. Створений був з метою прискорення розробки веб-додатків та забезпечення їхньої безпеки, масштабованості та підтримки. Django базується на принципі DRY, що сприяє зменшенню дублювання коду та підвищенню ефективності розробки.

Основні переваги Django:

1. Швидкість розробки: Завдяки вбудованим інструментам, як аутентифікація користувачів, обробка форм, Django дозволяє швидко створювати функціональні веб-додатки.

2. Безпека: Django надає засоби захисту від поширених веб-загроз, таких як SQL-ін'єкції, XSS, CSRF та інші.

3. Масштабованість: Завдяки архітектурі, Django можна легко масштабувати для підтримки великої кількості користувачів та обробки великих обсягів даних. Він дозволяє розподіляти навантаження між кількома серверами та використовувати різні бази даних.

4. Активна спільнота та підтримка: Не менш важливим є фактор активної та великої спільноти, які активно підтримують фреймворк, допомагають один одному, випускають оновлення та створюють нові пакети. Це надає доступ до великої кількості матеріалів, прикладів та документації.

5. Batteries Included: Django надає можливість використовувати широкий набір вбудованих інструментів та бібліотек, що покривають більшість потреб у веб-розробці. Це включає аутентифікацію, адмін-панель, систему обробки форм, управління базами даних та багато іншого.

Завдяки мові Python, фреймворк може використовувати архітектуру MVC, що зображено на рисунку 2.1.

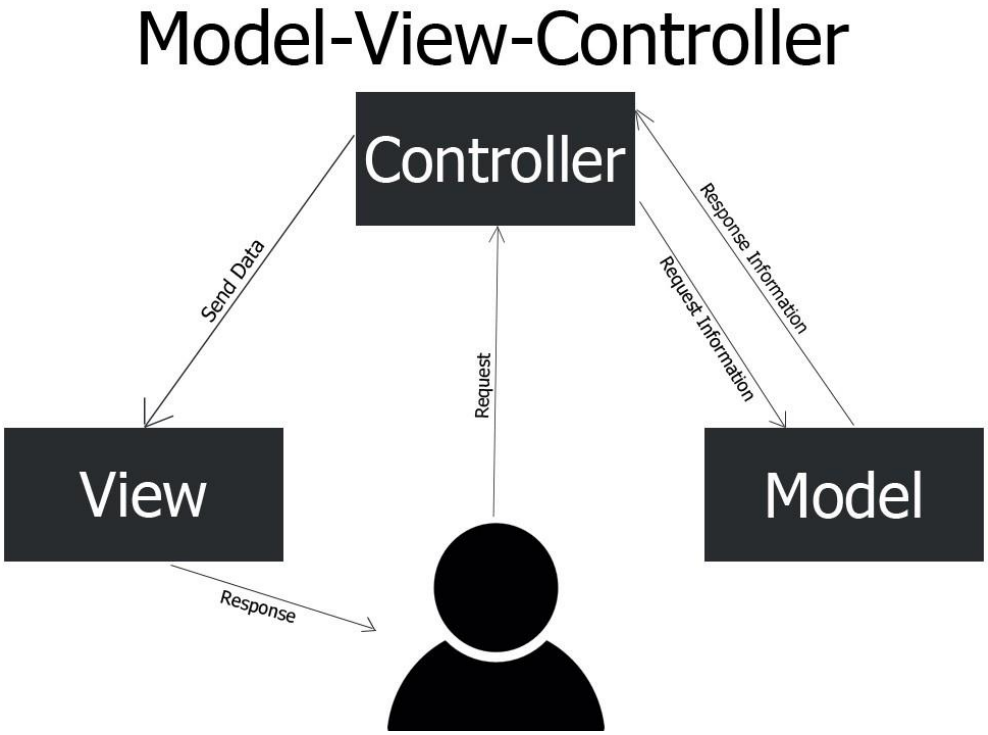


Рисунок 2.1 – Схема архітектури MVC

Проте, складно сказати, що Django використовує саме модель MVC. Контролер обробляється в самому середовищі розробки, а у цьому фреймворку вся основа відбувається в моделях та шаблонах. Тому, правильніше буде сказати, що Django використовує модель MVT[9], при цьому наслідуючи MVC, яку представлено на рисунку 2.2

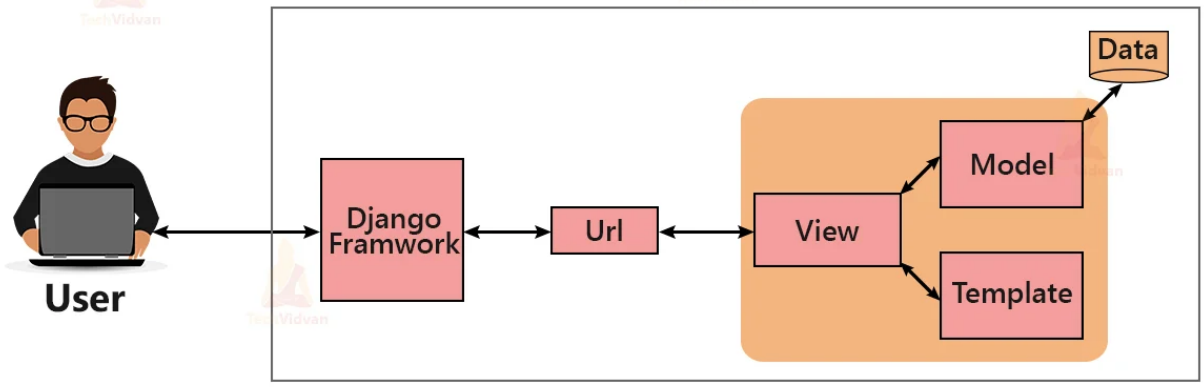


Рисунок 2.2 – Схема архітектури MVT

Розберемо, що собою представляє архітектура MVT:

1. Перший пласт, являє собою моделі. У ньому зберігається інформація про те, як можна отримати ту чи іншу інформацію.

2. Другий пласт, являє собою представлення даних. Цей пласт зберігає у собі проекції даних для проекту і відповідає за те, що має відображатися на веб-сторінці.

3. Третій пласт, вміщує у собі логіку проекту. Саме цей пласт відповідає за вибір шаблонів, які будуть показуватись користувачу при його діях.

Настав час, трохи сказати про Django ORM. В Django реалізована можливість об'єктно-реляційного відображення, саме це і дає йому змогу працювати з базою даних. Замість написання складних SQL-запитів (рис. 2.3), розробники можуть використовувати класи Python (рис. 2.3) для взаємодії з базою даних.

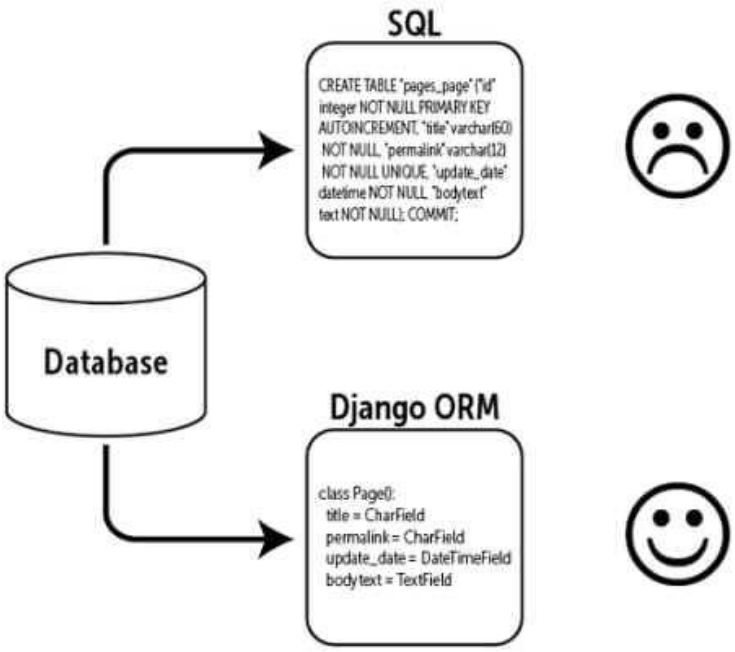


Рисунок 2.3 – Схема SQL та Django ORM запитів

ORM сильно прискорює розробку прототипів і готових додатків. Програмістові навіть не потрібно досконало знати SQL, для взаємодії з базами даних.

Слід зазначити, що у Django також є вже своя встановлена база даних, під назвою с SQLite, яка автоматично створюється на початку проекту.

Важливо додати, що Django не обмежує нас використанням тільки SQLite. Він дозволяє використовувати декілька баз даних в одному проекті. Наприклад SQLite можна використовувати на тестовому сервері під час розробки, а по її завершенню завдяки декільком командам почати працювати з будь-якою іншою. Проте, більшість розробників, майже з самого початку проекту, підключають потрібну БД.

Тому, до основних переваг Django ORM, можна віднести:

1. Інтуїтивність: Простий та зрозумілий інтерфейс для роботи з базами, що робить його доступним для розробників з різним рівнем досвіду.

2. Безпека: Django ORM автоматично захищає від SQL-ін'єкцій, що значно підвищує безпеку додатка.

3. Міграції: Надає інструменти для створення та управління міграціями бази даних, що спрощує процес оновлення структури нашої БД, без втрати даних.

4. Уніфікація: Забезпечує уніфікований спосіб роботи з різними типами баз даних, що робить код більш універсальним та легко підтримуваним.

2.2 Технологій клієнтської частини

Клієнтська частина веб-сайту є основною складовою, яка забезпечує взаємодію користувача з системою. Вона охоплює аспекти, пов'язані з відображенням інформації на екрані користувача та забезпеченням інтерактивності. Використання сучасних технологій для клієнтської частини дозволяє створювати привабливі, зручні та функціональні веб-додатки. У нашому проекті було використано кілька ключових технологій, кожна з яких виконує певну роль у розробці клієнтської частини.

Основою у створенні майбутніх веб-сторінок є HTML. Він визначає структуру веб-сторінки, використовуючи елементи та атрибути. HTML є фундаментом для веб-розробки і його треба використовувати у поєднанні з іншими технологіями для створення повноцінних веб-додатків. Сама технологія, була вперше представлена у 1991 році Тімом Бернерсом-Лі.

HTML5, остання версія HTML, була офіційно випущена в жовтні 2014 року і принесла дуже багато нових покращень. Насьогодні він включає в себе підтримку аудіо та відео без плагінів, нові семантичні елементи, та нові можливості для обробки графіки і анімації.

Згідно зі статистикою сайту W3Techs[10], HTML використовується майже на всіх веб-сайтах без винятку (рис. 2.4), що робить її найпоширенішою технологією в Інтернеті. Вона є основою для більшості інших веб-технологій і стандартів.

Markup Languages

Most popular markup languages

© W3Techs.com	usage	change since 1 May 2024
1. HTML	96.1%	+0.1%
2. XHTML	5.1%	-0.1%

percentages of sites

Рисунок 2.4 – Популярність HTML

Хоча HTML сьогодні є основою для створення веб-сторінок, уявити його без використання CSS не можливо. Мова стилів, використовується для опису зовнішнього вигляду HTML-документів. Вона дозволяє визначати стилі для елементів, такі як кольори, шрифти, відступи, розміри та розташування.

Останньою версією є CSS3, вона включає нові функції, такі як градієнти, тіні, анімації та трансформації, що дозволяють створювати складні та привабливі дизайни.

CSS використовується на більшості сучасних веб-сайтів і є стандартною технологією для створення візуально привабливих і функціональних веб-сторінок. За даними W3Techs, понад 96% всіх веб-сайтів використовують CSS для оформлення своїх сторінок, що ми і бачимо на рисунку 2.5.

Site Elements

Most popular site elements

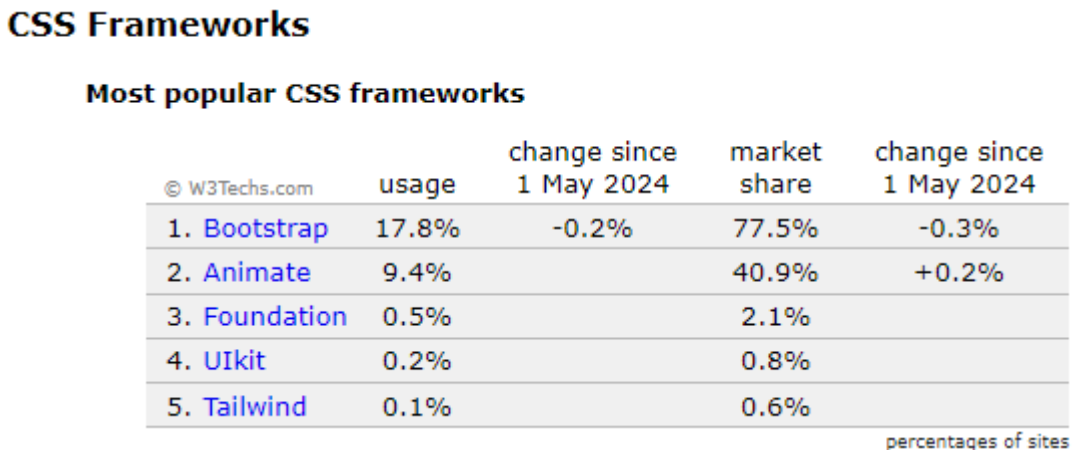
© W3Techs.com	usage	change since 1 May 2024
1. CSS	96.3%	-0.5%
2. Compression	88.7%	-0.1%
3. Default protocol https	85.8%	+0.2%
4. Cookies	41.7%	-0.2%
5. Default subdomain www	39.9%	+0.2%

percentages of sites

Рисунок 2.5 – Популярність CSS

Одним із не менш важливих для розробників є фреймворк Bootstrap. На 2024 рік, останньою версією є саме Bootstrap 5. Представляє собою найпопулярніший фреймворк для розробки адаптивних та мобільних веб-сайтів. Вперше його представили у 2011 році, розробниками виступила команда з Twitter. Завдяки фреймворку, спростилися процеси створення функціональних інтерфейсів. Він надає готові компоненти, такі як кнопки, форми, навігаційні панелі, модальні вікна та багато іншого, що значно прискорює розробку. Хоч дизайн і може виглядати на сьогодні дещо простим, але якщо розробник вже має досвід роботи з Bootstrap, він може створювати гарний дизайн.

Останні версії фреймворку привнесли багато покращень і змін. Головним аспектом стала відмова залежності від jQuery, додаткові компоненти та покращена підтримка для Flexbox і Grid Layout. Також стало помітне зменшення залежності від JavaScript, що в свою чергу дає більше простору, для використання фреймворків від самого JavaScript. Насьогодні майже 20% сайтів у мережі інтернет, використовує цей фреймворк, що видно на рисунку 2.6.



CSS Frameworks

Most popular CSS frameworks

© W3Techs.com	usage	change since 1 May 2024	market share	change since 1 May 2024
1. Bootstrap	17.8%	-0.2%	77.5%	-0.3%
2. Animate	9.4%		40.9%	+0.2%
3. Foundation	0.5%		2.1%	
4. UIkit	0.2%		0.8%	
5. Tailwind	0.1%		0.6%	

percentages of sites

Рисунок 2.6 – Популярність Bootstrap

Мова програмування, що дозволила нам створювати динамічні та інтерактивні веб-сторінки і була вперше представлена у 1995 році, має назву JavaScript. Це один з стовпі, що стоїть поряд з HTML і CSS.

JavaScript є мовою, що швидко розвивається, має велику екосистему бібліотек і фреймворків, таких як React, Angular, Vue.js, які дозволяють створювати складні додатки.

За статистикою W3Techs, вона використовується на 98.9% всіх веб-сайтів, що робить його невід'ємною частиною сучасної веб-розробки. За думкою багатьох, вона ще навіть не досягла свого піку і в майбутньому ми можемо побачити новий стрибок цієї мови у Web3[11] розробках.

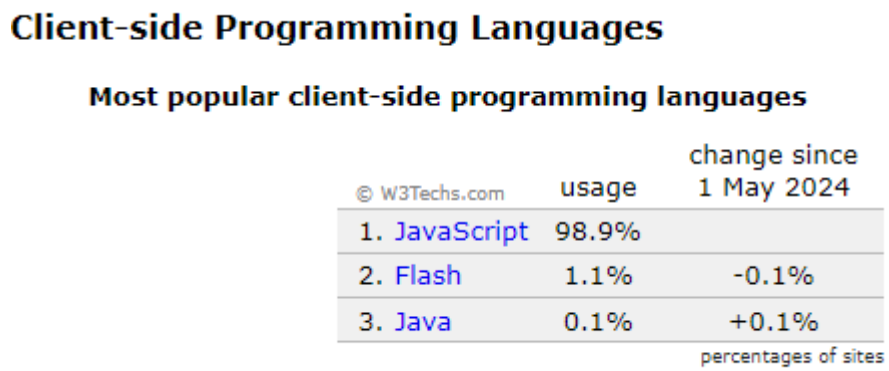


Рисунок 2.7 – Популярність JavaScript

Одна з найпопулярніших бібліотек JavaScript має назву jQuery. Головними її перевагами є легкість, швидкість і багатофункціональність. Ці аспекти, які спрощують роботу з HTML-документами, обробкою подій, анімацією та AJAX-взаємодією. Вперше бібліотека була випущена у 2006 році і стала однією з найпопулярніших бібліотек JavaScript (рис 2.8).

JavaScript Libraries

Most popular JavaScript libraries

© W3Techs.com	usage	change since 1 May 2024	market share	change since 1 May 2024
1. jQuery	76.7%	-0.2%	94.3%	-0.1%
2. Bootstrap	20.0%	-0.3%	24.6%	-0.3%
3. Underscore	9.5%		11.7%	+0.1%
4. Modernizr	7.9%	-0.1%	9.7%	-0.1%
5. Popper	4.5%		5.5%	

percentages of sites

Рисунок 2.8 – Популярність JQuery

Останньою технологією, є AJAX. Він створює асинхронні веб-додатки. Одним з головних факторів і чому його використовують, є те, що AJAX дозволяє здійснювати запити до серверу і оновлювати частини веб-сторінки без повного перезавантаження. Це дуже покращує UX, роблячи взаємодію користувача з веб-додатком більш інтерактивною і швидкою.

Вперше його представили у 2005 році і він став однією з ключових технологій для створення сучасних веб-додатків. Він використовується у таких популярних проектах, як Gmail, Google Maps, Facebook та інші.

Підбиваючи підсумки, можна сказати, що використання сучасних технологій для клієнтської частини є ключовим фактором у створенні успішних і функціональних веб-додатків. HTML, CSS, Bootstrap 5, JavaScript, jQuery та AJAX забезпечують необхідний інструментарій для створення інтерфейсів, які є одночасно привабливими, швидкими та зручними у використанні. Кожна з цих технологій має унікальні особливості, що дозволяє створювати веб-додатки, які відповідають сучасним тенденціям і стандартам.

2.3 Вибір архітектури мережевого WEB-ресурсу

Завдяки аналізу аналогів, було помічено, що архітектура мережевого веб-ресурсу є ключовим елементом у розробці ефективного та масштабованого сайту. Від правильного вибору архітектури залежить ефективність, продуктивність, масштабованість і безпека системи. У сучасному світі, вимоги до проектів постійно зростають, архітектура стає фундаментом, на якому будується весь проект.

Головними значеннями архітектури веб-ресурсу є:

Ефективність та Продуктивність: Архітектура визначає, як компоненти системи взаємодіють між собою. Це обробка запитів, управління даними, виконання бізнес-логіки та відправка відповідей клієнту. Правильне налаштування процесів підвищить швидкість та ефективність роботи. Якщо архітектура є оптимально, вона забезпечить швидке завантаження сторінок та обробку запитів користувачів.

Масштабованість: Сьогодні веб-додатки треба готувати до зростання навантаження. Система має легко масштабуватись, щоб була можливість додати нові сервери або у разі чого, розподілити навантаження між існуючими серверами.

Безпека: Безпека є одним з найважливіших аспектів будь-якого веб-додатка. Архітектура визначає способи захисту даних, управління доступом та запобігання загрозам. Правильний вибір архітектури і її налаштування, дозволяють забезпечити високий рівень безпеки для даних користувачів.

Гнучкість та Легкість у підтримці: Важливим є побудова архітектури, яка дозволяє легко вносити зміни до системи. Додавання нових функцій, виправлення помилок та оновлення компонентів, забезпечує гнучкість та скорочує час на підтримку та розвиток проекту.

Існує декілька актуальних архітектурних моделей, які використовуються для розробки. У кожної з них, є свої переваги та недоліки, які потрібно враховувати при виборі архітектури для проекту.

Клієнт-серверна архітектура (рис. 2.9) є одна з найпоширеніших моделей для веб-додатків. У цій архітектурі клієнт взаємодіє з сервером через HTTP-запити. Сервер обробляє запити клієнта, виконує необхідні операції та повертає відповідь клієнту у вигляді HTML-сторінок, даних або інших форматів.

Основні переваги клієнт-серверної архітектури:

1. Масштабованість: При використанні цієї архітектура, можна легко масштабувати систему, додаючи нові сервери або розподіляючи навантаження між кількома.

2. Безпека: Сервер може забезпечити захист даних та контроль доступу до ресурсів, зменшуючи ризики несанкціонованого доступу та атак.

3. Централізоване управління: Усі дані та логіка обробки знаходяться на сервері, що полегшує їх оновлення та управління.

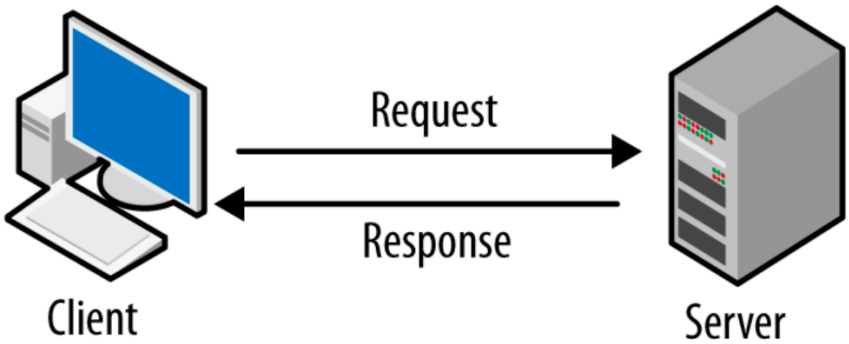


Рисунок 2.9 – Клієнт-серверна архітектура

Трирівнева архітектура, що зображена на рисунку 2.10, розділяє систему на три основні рівні: презентаційний, логічний та рівень даних.

Презентаційний рівень відповідає за інтерфейс користувача та взаємодію з клієнтом.

Логічний рівень додає бізнес-логіку додатка, яка виконується на сервері. Для даного проекту, використовується Django як основний фреймворк для обробки запитів, автентифікації користувачів та інших серверних завдань.

Рівень даних: Відповідає за зберігання та управління даними. Вона має забезпечуючи надійне та масштабоване зберігання інформації про продукти, користувачів, замовлення та інше.

Основні переваги трирівневої архітектури:

1. Розділення обов'язків: Кожен рівень відповідає за свою частину системи, що спрощує розробку, тестування та обслуговування.

2. Масштабованість: Кожен рівень можна масштабувати окремо, додаючи нові сервери для презентаційного, логічного або рівня даних.

3. Гнучкість: Трирівнева архітектура дозволяє легко оновлювати та замінювати окремі компоненти без впливу на інші частини системи.

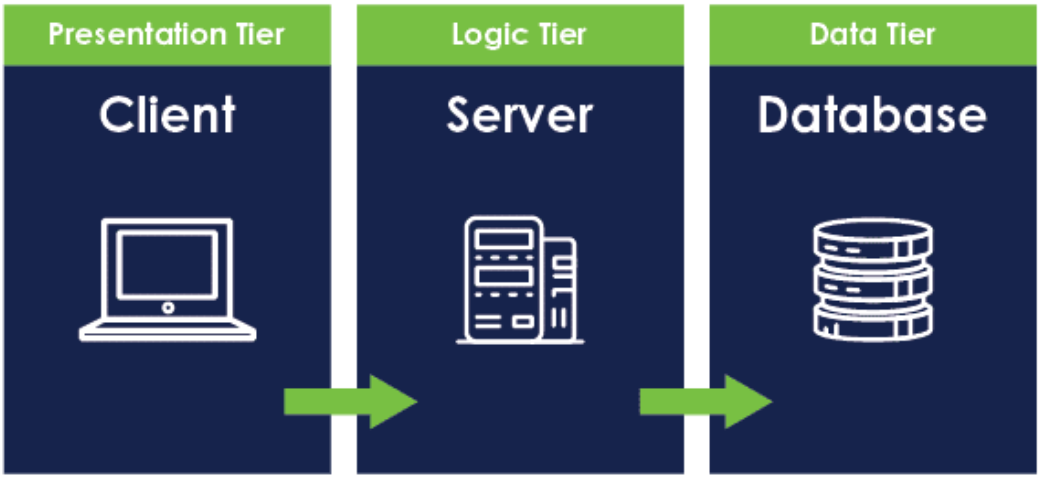


Рисунок 2.10 – Трирівнева архітектура

Також розглянемо, RESTful архітектуру (рис. 2.11). RESTful API дозволяє клієнту виконувати запити до сервера та отримувати відповіді у форматі JSON або XML, що забезпечує ефективний та гнучкий обмін даними.

Основні принципи роботи RESTful архітектури це:

1. Засоби передачі стану: Клієнт виконує запити до серверу, і сервер повертає відповідь, що відображає поточний стан ресурсу.

2. Стандартні методи HTTP: RESTful API використовує стандартні методи HTTP для виконання CRUD-операцій

3. Статус-коди HTTP: Сервер повертає відповідні статус-коди HTTP для інформування клієнта про результат запиту.

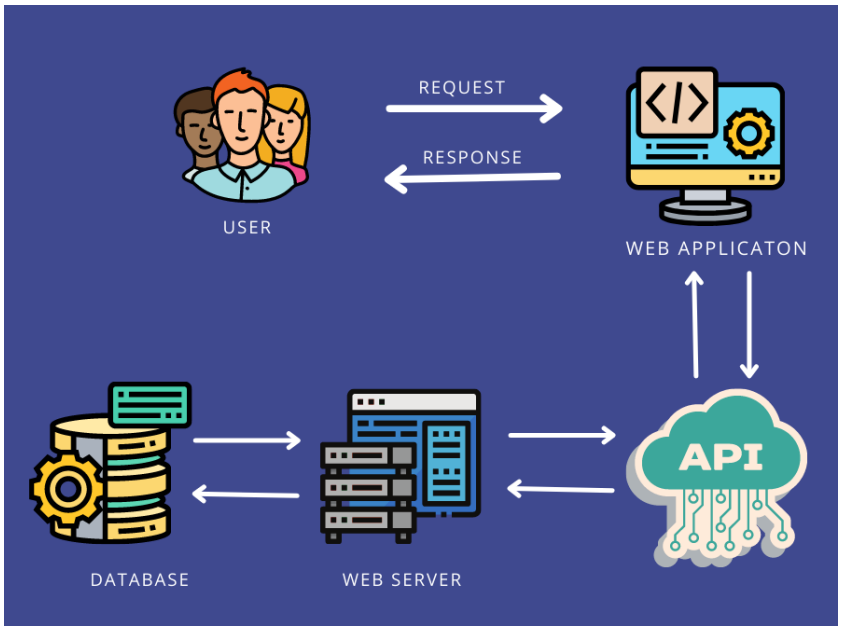


Рисунок 2.11 – RESTful архітектура

Вибір архітектури мережевого веб-ресурсу є важливим етапом у розробці сайту ексклюзивного одягу. Тому для реалізації проекту було обрана трирівнева архітектура, що дозволить реалізувати ефективний веб-ресурс, який відповідає сучасним вимогам ринку.

2.4 Вибір системи управління базами даних

Основним компонентом будь-якої інформаційної системи, яка працює з великими обсягами даних є система управління базами даних. СУБД забезпечує зберігання, організацію, управління та обробку даних, що є

необхідним для ефективного функціонування сучасних веб-додатків, таких як інтернет-магазини. Тому важливо підібрати правильну СУБД для свого проекту, враховуючи всі аспекти.

Існує кілька основних типів СУБД, кожен з яких має свої характеристики та застосування:

1. Реляційні СУБД: Вони базуються на реляційній моделі даних, де дані зберігаються у вигляді таблиць. Відношення між таблицями визначаються за допомогою ключів. До цих баз даних відносяться PostgreSQL, MySQL, Oracle, Microsoft SQL Server. Їх особливістю є висока надійність, підтримка складних запитів, забезпечення цілісності даних.

2. NoSQL СУБД: Не використовують реляційну модель і призначені для роботи з неструктурованими даними або даними, які швидко змінюються. Такими базами є MongoDB, Cassandra, Redis. Їх переваги, це висока продуктивність, масштабованість, гнучкість у роботі з різними типами даних.

3. Графові СУБД: Ці бази спеціалізуються на зберіганні та обробці даних, що мають складні взаємозв'язки. Дані представлені у вигляді графів, де вузли відповідають об'єктам, а ребра — зв'язкам між ними. Як приклад, Neo4j, Amazon Neptune. Їх перевагами є ефективність у роботі з даними, що мають багато взаємозв'язків, таких як соціальні мережі або рекомендаційні системи.

Проаналізувавши всі аспекти нашого проекту, включаючи нашу архітектуру та мову розробки, найбільш відповідною для проекту буде база даних PostgreSQL, що буде показано більш детально у наступному розділі.

PostgreSQL є популярною реляційною системою управління базами даних, яка має багато переваг у порівнянні з іншими СУБД. Подивимось на рейтинг популярних СУБД з сайту DB-Engines[7] на рисунку 2.12.

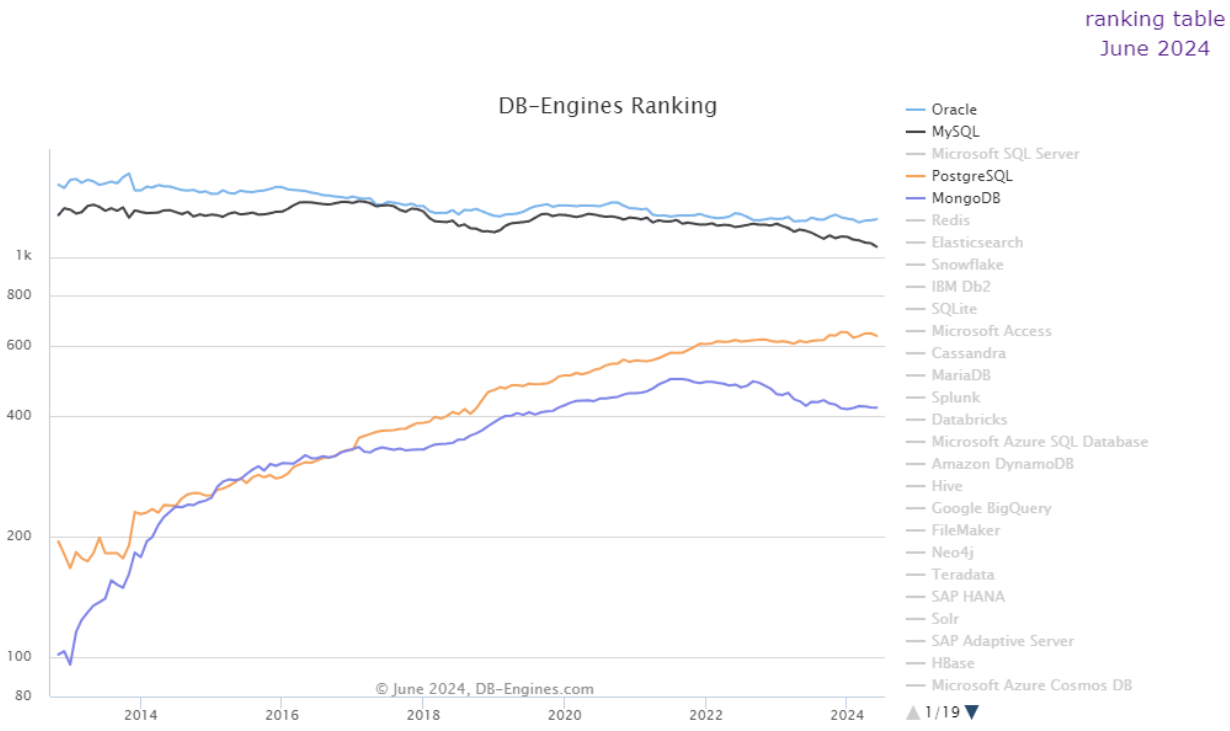


Рисунок 2.12 – рейтинг СУБД

Вибір системи управління базами даних залежить від специфічних вимог проекту. PostgreSQL, як реляційна СУБД з відкритим кодом, забезпечує високу надійність, продуктивність і підтримку складних запитів, що робить її ідеальним вибором для нашого проекту. Вона також пропонує розширену підтримку JSON, що дозволяє легко інтегруватися з сучасними веб-технологіями і обробляти дані в різних форматах.

2.5 Порівняння з альтернативними рішеннями

При розробці мережевого веб-ресурсу для продажу ексклюзивного одягу, взуття та аксесуарів важливо вибрати оптимальну архітектуру та програмні засоби. Для цього розглянемо альтернативні рішення, які можна було б використати замість обраних технологій, та порівняємо їх.

Фреймворки які розглянемо для веб-розробки:

1. Flask (Python)
2. Ruby on Rails (Ruby)
3. Express.js (Node.js)

Системи управління базами даних:

1. MySQL
2. Oracle
3. MongoDB

Фронтенд-технології:

1. React
2. Vue.js
3. Angular

Вибрані технології для веб-розробки розглянемо у вигляді (табл. 2.1)

Таблиця 2.1 Порівняння Django з альтернативними фреймворками

Параметр	Django (Python)	Flask (Python)	Ruby on Rails (Ruby)	Express.js (Node.js)
Архітектура	MVT	WSGI	MVC	Middleware
Набір функцій	Повний набір (вбудовані адмін панель, ORM тощо)	Мінімальний, розширюваний через плагіни	Повний набір (вбудовані засоби)	Мінімальний, розширюваний через плагіни
Складність	Середня	Середня	Середня	Середня
Документація та підтримка	Висока	Висока	Висока	Висока
Масштабованість	Висока	Висока	Висока	Висока
Продуктивність	Висока	Висока	Висока	Висока
Гнучкість	Середня	Середня	Середня	Середня

Бази даних, що були вибрані для аналізу, розглянемо у таблиці 2.2

Таблиця 2.2 – Порівняння PostgreSQL з іншими СУБД

Параметр	PostgreSQL	MySQL	Oracle	MongoDB
Тип СУБД	Реляційна	Реляційна	Реляційна	Документо-орієнтована
Мова запитів	SQL, PL/pgSQL	SQL	SQL, PL/SQL	JSON-подібні запити
Підтримка ACID	Повна	Повна	Повна	Часткова
Масштабованість	Висока	Висока	Дуже висока	Висока
Підтримка JSON	Так	Так з версії 5.7	Так з версії 12c	Так
Розширюваність	Висока (плагіни, розширення)	Обмежена	Висока (плагіни, розширення)	Дуже висока
Ліцензія	PostgreSQL License (відкрита)	GNU GPL (відкрита)	Пропрієтарна	Server Side Public License
Реплікація та резервне копіювання	Так (асинхронна, синхронна)	Так (асинхронна, синхронна)	Так (асинхронна, синхронна, фізична)	Так (асинхронна, синхронна)
Безпека	Висока (розширені механізми)	Висока	Дуже висока (розширені механізми)	Середня (менше функцій безпеки)
Підтримка складних запитів	Так	Так	Так	Обмежена
Продуктивність	Висока (особливо для складних запитів)	Висока (висока швидкість читання)	Дуже висока	Висока
Підтримка різних типів даних	Так (напр., JSON, XML, hstore)	Так (обмежена порівняно з PostgreSQL)	Так (напр., JSON, XML)	Так (напр., BSON)
Транзакційність	Так (повна підтримка ACID)	Так (повна підтримка ACID)	Так (повна підтримка ACID)	Обмежена

Параметр	PostgreSQL	MySQL	Oracle	MongoDB
Масштабування по горизонталі	Так	Так	Так	Так
Ринок використання	Відкритий код, підприємства, стартапи	Веб-додатки, стартапи, малі проекти	Великі підприємства, банки, урядові структури	Великі дані, аналітика, документо-орієнтовані додатки

Вибрані фронтенд-технологій розглянемо у вигляді таблиці 2.3.

Таблиця 2.3 – альтернативні фронтенд-технології

Параметр	React	Vue.js	Angular
Тип	Бібліотека JavaScript	Фреймворк JavaScript	Фреймворк JavaScript
Гнучкість	Висока	Висока	Висока
Продуктивність	Висока	Висока	Висока
Легкість використання	Середня	Висока	Середня
Документація та підтримка	Висока	Висока	Висока
Популярність	Дуже висока	Висока	Висока

3 АРХІТЕКТУРНЕ ПРОЕКТУВАННЯ ТА МОДЕЛЮВАННЯ

3.1 Проектування інтерфейсів користувача та навігації

Проектування інтерфейсів користувача UI та навігації є критичним етапом розробки веб-ресурсу. Інтерфейс користувача визначає, як користувачі взаємодіють з системою, а навігація – як вони пересуваються по сайту. У цьому пункті буде розроблено схеми навігації, яка забезпечить зручність і ефективність у використанні сайту.

Навігація є важливим аспектом користувацького досвіду UX, що забезпечує швидкий і легкий доступ до різних розділів сайту. Хороша навігація повинна бути інтуїтивно зрозумілою і логічною.

Основні принципи проектування навігаційної структури:

1. Простота. Прості і зрозумілі меню та навігація шляхів. Користувачу не потрібно багато зайвих елементів, зоб його не перевантажувати.

2. Консистентність. Елементи на сторінці, мають бути однакового вигляду. Навігаційні елементи мають функціонувати на всіх сторінках. Це дозволяє клієнту швидко зрозуміти, як користуватися сайтом.

3. Доступність. Створення навігації, яка буде зручною для всіх користувачів. Використання стандартів веб-доступності WCAG для забезпечення цього аспекту.

Також існує, декілька типів навігацій, кожна з яких має свої певні умови та місця для розташування:

1. Глобальна навігація. Головне меню або навігаційна панель, яке зазвичай розташоване у верхній частині сторінки. Вона включає основні посилання на різні розділи та сторінки сайту. Розглянемо таку навігацію, для нашого проекту на рисунку 3.1.

2. Локальна навігація. Такий тип меню, що відображається тільки на конкретних сторінках або з’являється при певних умовах. Це надає комфортний доступ до додаткових функцій, або налаштувань.

3. Контекстуальна навігація. Посилання та елементи, що з’являються в залежності від контексту сторінки, на якій перебуває користувач. Зазвичай, це навіть можуть бути анімаційні елементи або реклама, що з'являється для інформування користувача.

Рисунок 3.1 – Глобальна навігація проекту

Навігаційні елементи є основою, з чого складається навігаційна структура. Частина з них може відноситися до глобальної навігації, частина до локальної. До них відносять:

– Меню: Використання горизонтальних або вертикальних меню для забезпечення доступу до основних розділів сайту. Це є важливою складовою, тому вони повинні бути зрозумілими та легко доступними.

– Хлібні крихти. Це елемент навігації, що показують шлях користувача від головної сторінки до поточної. Вони допомагають користувачам зрозуміти, де вони знаходяться на сайті, і швидко повернутися до попередніх розділів.

– Фільтри та сортування. Важливий інструмент особливо в побудові інтернет-магазину. Завдяки фільтрам, потенційний клієнт може детально налаштувати пошук, під свої потреби, що дає йому змогу швидко знайти потрібний товар. Як вони будуть виглядати у нашому проекті, зображено рисунку 3.2.

– Пошук. Пошуковий механізм, також є елементом, який дозволяє швидко знаходити інформацію на сайті за ключовими словами.

Рисунок 3.2 – Локальна навігація фільтрів

Для того, щоб спроектувати правильну навігаційну структуру, потрібно дослідити користувачів. В цьому може, допомогти проведення аналізу цільової аудиторії, що надасть нам інформацію про їх потреби та очікування від навігації. Зазвичай, треба проводити інтерв'ю, опитування, або можна оцінити вже готові аналізи поведінки користувачів на аналогічних веб-ресурсах.

Створення інформаційної архітектури, потребує визначення логічної структури сайту. Сюди включають категорії та підкатегорії, що забезпечують зручний доступ до інформації та функцій.

Використання інструментів для прототипування є дуже корисним, їх можна створювати у таких додатках як Canva або Figma. Корисним буде проведення юзабіліті-тестувань з реальними користувачами, щоб оцінити ефективність навігаційної структури. Завдяки цьому, можна отримати зворотній зв'язок та внести коригування на основі результатів, ще до початку основної розробки.

3.2 Функціональне моделювання системи SADT

Методологія SADT – це методологія функціонального моделювання, яка використовується для аналізу та проектування систем. Вона була розроблена в 1970-х роках та забезпечує засоби для графічного подання функцій системи та взаємодій між цими функціями. Основною метою SADT є забезпечення зрозумілої та структурованої моделі проекту, яка полегшує аналіз та проектування.

Які переваги надає використання SADT:

1. Чітке визначення функцій: SADT дозволяє чітко визначити всі функції системи та взаємодії між ними.

2. Структуроване проектування: Завдяки декомпозиції функцій на підфункції, SADT допомагає створити ієрархічну структуру системи.

3. Комунікація: Графічні моделі, створені за допомогою SADT, легкі для розуміння всіма учасниками проєкту, включаючи розробників, аналітиків та зацікавлених осіб.

У нашому проєкті розробки інтернет-магазину ексклюзивного одягу ми використовуватимемо SADT для створення чіткої та структурованої моделі системи. Це допоможе детально описати всі функції інтернет-магазину, включаючи обробку замовлень, управління товарним асортиментом, взаємодію з користувачами та забезпечення безпеки даних. На рисунку 3.3 зобразимо контекстну діаграму інтернет-магазину у графічному вигляді.

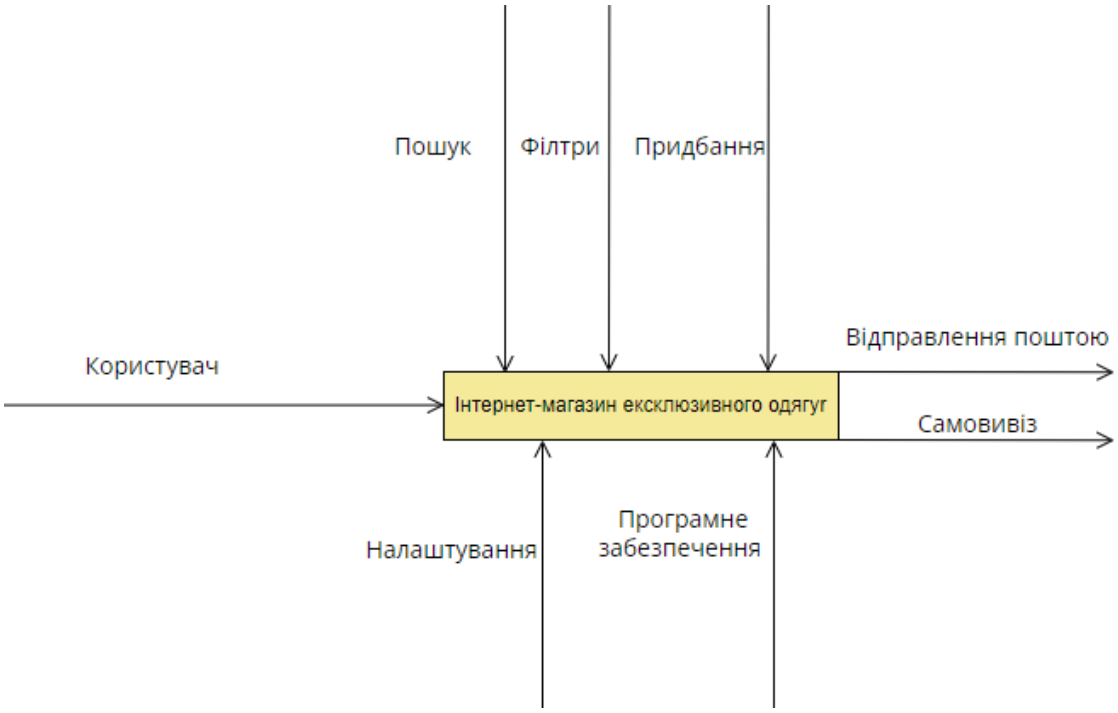


Рисунок 3.3 – Контекстна діаграма інтернет-магазину

Наступним етапом є діаграма декомпозиції системи, вона представляє собою метод для розбиття складної системи на менші, більш прості компоненти. Це дозволяє краще зрозуміти, як система працює та які процеси вона включає.

Ця діаграма розбиває інтернет-магазин на основні функціональні блоки, а потім подальшу декомпозицію цих блоків на їх складові частини. Основна мета такої діаграми - дозволити детально розглянути всі функції і компоненти системи, вказуючи на їхні взаємозв'язки та взаємодії.

У проєкті інтернет-магазину ми використовуємо діаграму декомпозиції для кращого розуміння функцій системи та планування розробки. Це дозволяє нам ефективно визначити всі необхідні функції, які має виконувати система, та забезпечити їх взаємодію для досягнення максимальної ефективності і зручності для користувачів.

Створимо діаграму, що покаже нам продаж доступних товарів на сайті на рисунку 3.4. Діаграма декомпозиції продажу товарів показує основні функціональні блоки, які включають процес продажу товарів у інтернет-магазині. Це допомагає зрозуміти, як організований процес від перегляду товарів до завершення покупки.

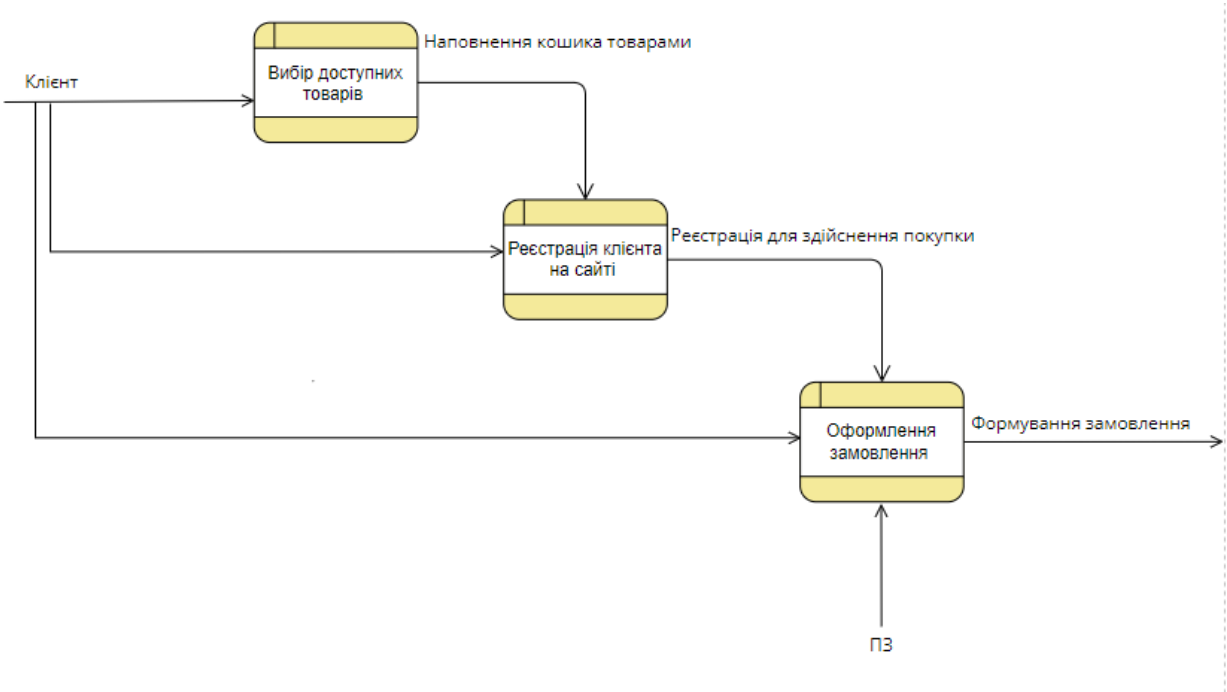


Рисунок 3.4 – діаграма декомпозиції продажу товарів

Створення діаграми декомпозиції системи обробки замовлень покаже нам процеси, які відбуваються від моменту отримання замовлення до його виконання. Це допомагає детально зрозуміти, як здійснюється обробка кожного замовлення в системі, що зображено на рисунку 3.5.

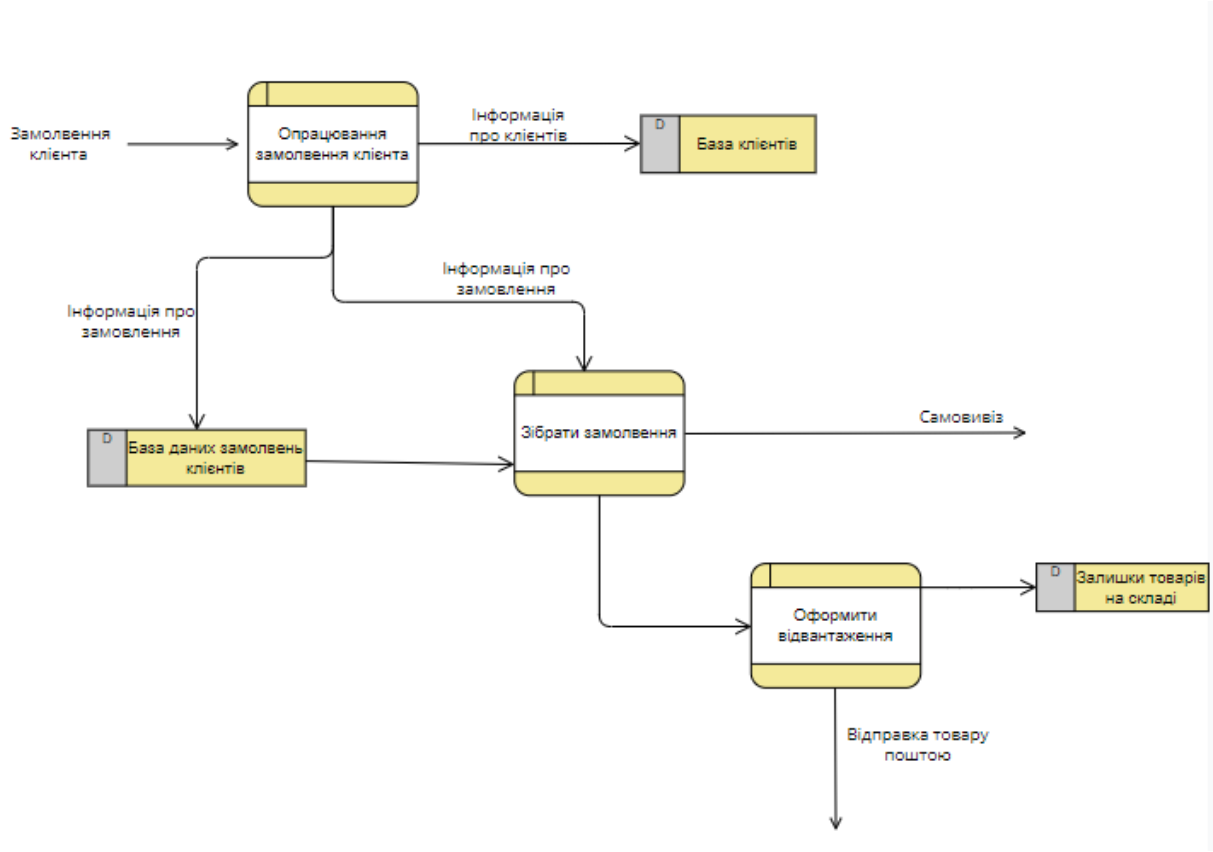


Рисунок 3.5 – діаграма декомпозиції системи обробки замовлень

3.3 Моделювання бізнес-процесів Workflow Diagramming

Workflow Diagramming – це метод візуалізації бізнес-процесів, який використовується для відображення послідовності кроків або дій, що виконуються в межах певного процесу або операції. Такі діаграми допомагають зрозуміти потік роботи, та ідентифікувати можливі проблемні місця, що дасть змогу виправити ці процеси і досягти більшої ефективності.

Workflow Diagramming використовується для:

– Візуалізації послідовності кроків у бізнес-процесі.

– Ідентифікації ролей та відповідальності в межах процесу.
– Виявлення потенційних проблем або вузьких місць у процесах.
– Оптимізації та покращення бізнес-процесів.
– Документування існуючих процесів для навчання та аналізу.

Типи Workflow Diagramming:

1. Flowchart: Має вигляд блок-схеми і використовується для зображення послідовності кроків у процесі з використанням символів, таких як овали, прямокутники, ромби і стрілки.

2. Swimlane Diagram: Має вигляд діаграми з плаваючими лініями і розбиває процес на паралельні доріжки або смуги, кожна з яких представляє окрему роль або підрозділ.

3. BPMN: Являє собою стандарт для бізнес-процесів, який використовує набір графічних нотаток для представлення бізнес-процесів у вигляді діаграм.

Створимо діаграму, яка ілюструє етапи, які проходить клієнт від моменту реєстрації на сайті до завершення процесу оформлення замовлення (рис. 3.6). Вона охоплює основні кроки та взаємодії з системою, необхідні для успішного здійснення покупки. Діаграма служить візуальним посібником, що демонструє ключові етапи взаємодії клієнта з веб-ресурсом, починаючи від створення облікового запису, перегляду та вибору товарів, і закінчуючи остаточним оформленням замовлення та підтвердженням отримання товару.

Основна мета діаграми — показати послідовність дій користувача, забезпечити чітке розуміння функціональних можливостей системи, і висвітлити всі важливі точки контакту клієнта з веб-ресурсом. Вона також допомагає забезпечити плавний і ефективний користувацький досвід, а також впровадити необхідні покращення для підвищення зручності та задоволеності клієнтів.

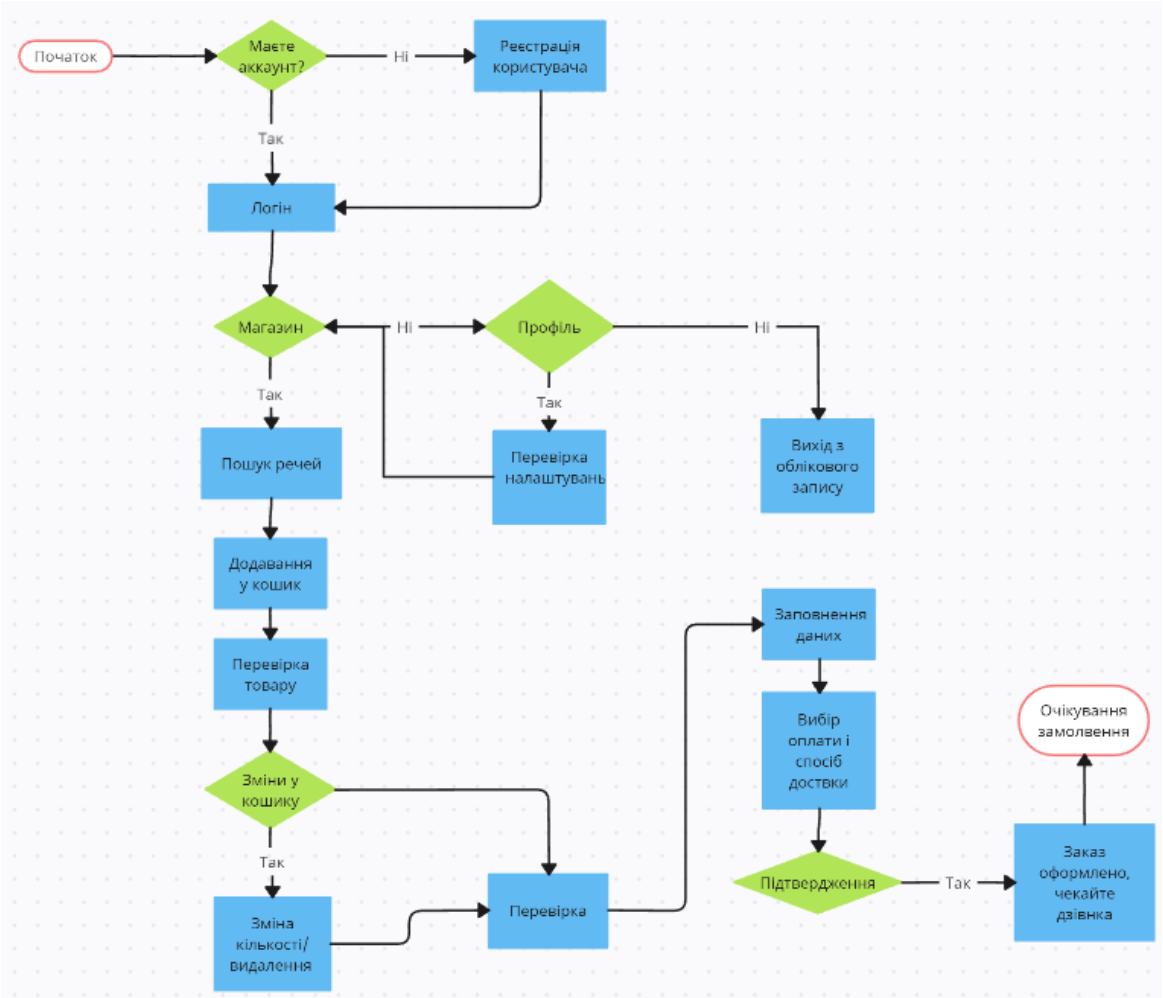


Рисунок 3.6 – Діаграма декомпозиції дій клієнта

Ця діаграма є важливим інструментом для розробників та дизайнерів, які працюють над створенням інтернет-магазину, оскільки вона дозволяє детально вивчити всі етапи взаємодії клієнта з системою та забезпечити їх ефективну реалізацію.

3.4 Проектування бази даних

Проектування бази даних є одним з найважливіших етапів розробки веб-ресурсу, оскільки від його правильності залежить продуктивність, масштабованість і надійність системи. Для інтернет-магазину, що спеціалізується на продажу ексклюзивної одягу, обрання відповідної

структури бази даних є критично важливим для забезпечення ефективного зберігання та обробки великих обсягів інформації про товари, користувачів, замовлення та інші сутності. База даних повинна бути добре спроектована для забезпечення швидкого доступу до даних, ефективного виконання запитів, забезпечення цілісності даних та безпеки.

У нашому проекті ми використовуємо трирівневу архітектуру, яка складається з трьох рівнів, що були розглянуті до цього. Такий підхід забезпечує модульність, масштабованість та зручність у розробці. Трирівнева архітектура дозволяє ефективно розподіляти навантаження між різними рівнями, що особливо важливо для забезпечення швидкої роботи веб-ресурсу при високих навантаженнях.

Для нашого проекту було обрано реляційну систему управління базами даних PostgreSQL, яка надає потужні можливості для роботи з великими обсягами даних та забезпечує високу продуктивність. Подивимось на основні етапи проектування бази даних

1. Визначення сутностей та атрибутів: На цьому етапі, визначається основні сутності таблиць, такі як користувачі, товари, замовлення, категорії, бренди тощо. Визначення атрибутів для кожної сутності. Наприклад, для сутності товари, це можуть бути атрибути: назва, опис, ціна, кількість на складі, зображення, категорія, бренд тощо.

2. Встановлення зв'язків між сутностями: Визначення типів зв'язків між сутностями є важливими і необхідним етапом. Наприклад, зв'язок між сутностями користувачі і замовлення є один-до-багатьох, оскільки один користувач може мати багато замовлень.

3. Нормалізація даних: Виконання нормалізації бази даних для уникнення надмірності інформації і забезпечення їх цілісності.

4. Створення схем бази даних: Розробка схеми бази даних, яка відображає структуру таблиць і зв'язків між ними. Ця схема бази даних, дозволить візуально представити структуру даних і їх взаємозв'язки.

5. Реалізація та тестування: Тестування бази даних для перевірки її працездатності, продуктивності та цілісності даних.

Django використовує систему ORM, яка дозволяє розробникам працювати з базою даних за допомогою об'єктів Python. Це значно спрощує розробку, оскільки зменшує необхідність написання складних SQL-запитів і підвищує продуктивність. Для налаштування підключення до бази даних у Django[8] використовуються параметри конфігурації у файлі settings.py, де визначаються ім'я бази даних, користувач, пароль, хост та порт.

Настав час розібрати усі моделі, сутності та зв'язки, які створені у нашій базі даних для інтернет-магазину.

Модель «Продукт» має наступні атрибути, що вказані на рисунку 3.7. Бачимо, що саме включає у себе структура продуктів.

Рисунок 3.7 – Таблиця «Продукт»

Модель «Розміри» має наступні атрибути, що вказані на рисунку 3.7. Вказано одразу дві таблиці пов’язаних з розмірами, що мають між собою зв’язок.

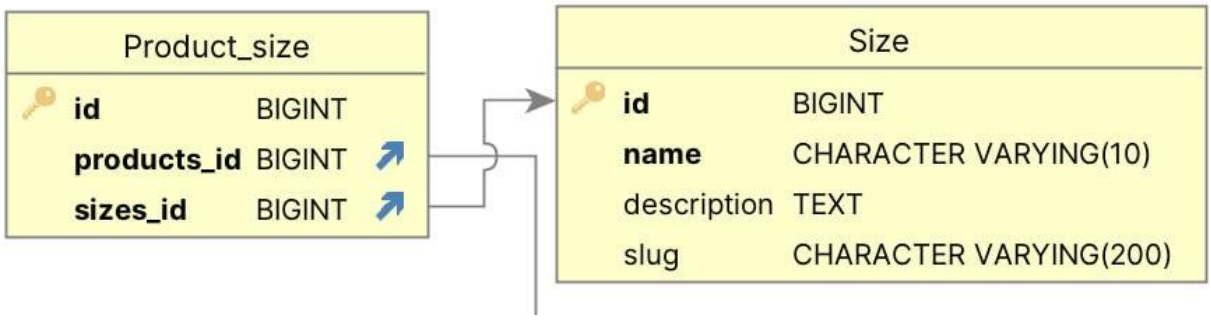


Рисунок 3.8 – Таблиця «Розміри»

Модель «Бренд» має наступні атрибути, що вказані на рисунку 3.7. Дана таблиця, створена для того, щоб записувати бренди, компанії чиїх будуть продаватися в інтернет-магазині.

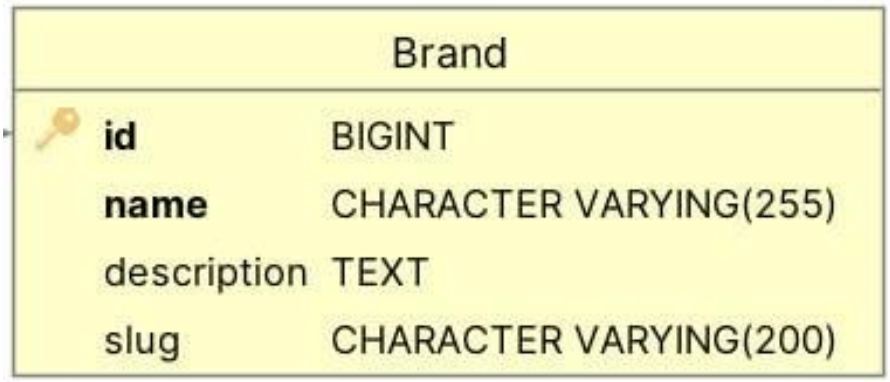


Рисунок 3.9 – Таблиця «Бренд»

Модель «Категорії», що показана на рисунку 3.10, створена для фільтрації товарів на сайті. Це дає змогу користувачу, швидше шукати потрібні позиції товарів.

Рисунок 3.10 – Таблиця «Категорії»

Модель «Користувач» (рис. 3.11), показує, які дані і налаштування покупців знаходяться в базі даних.

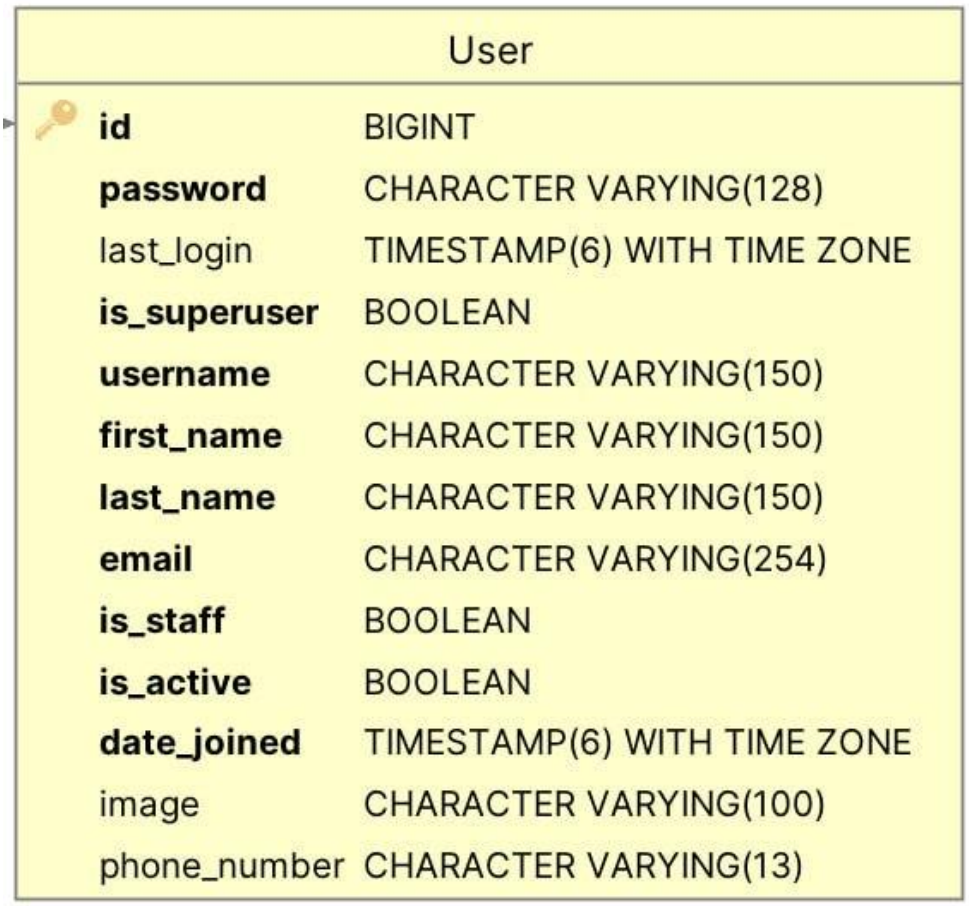


Рисунок 3.11 – Таблиця «Користувач»

Модель «Кошик», створення для зберігання внесених користувачем продуктів (рис. 3.12). Як ми бачимо, кожен кошик прив’язаний до кожного користувача за їх user_id, тим, що було показано перед цим.

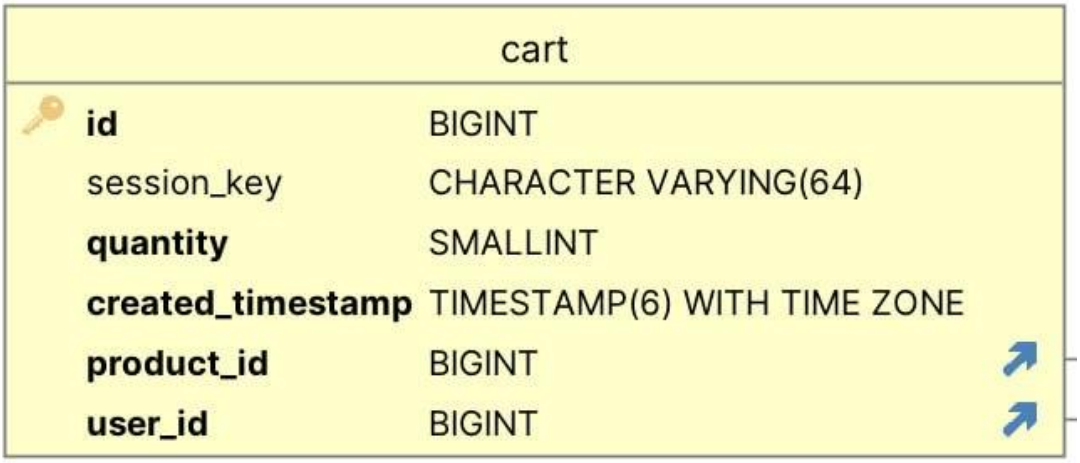


Рисунок 3.12 – Таблиця «Кошик»

Модель «Замовлення», зберігає в собі дані, що вводить користувач при оформленні замовлення, можемо це побачити які саме на рисунку 3.13.

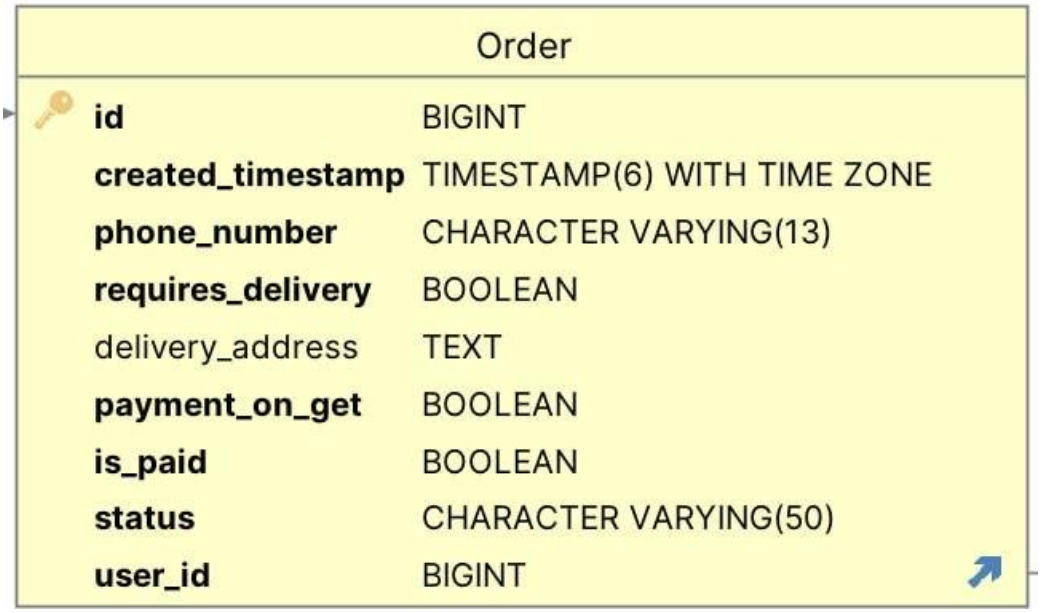


Рисунок 3.13 – Таблиця «Замовлення»

Модель «Замовленні товари» на рисунку 3.14. Створена для того, щоб бачити, які саме товари замовив користувач оформлюючи замовлення. Також це допомагає магазину, вести підрахунок проданих товарів і бачити залишки товарів на складі, після продажу.

Рисунок 3.14 – Таблиця «Замовленні товари»

Описавши всі основні моделі бази даних для проекту інтернет-магазину ексклюзивного одягу, наступним кроком є побудова моделі «сутність-зв'язок» (рис. 3.15). Ця модель дає загальне уявлення про структуру даних і взаємозв’язки між ними, що дозволяє зрозуміти систему в цілому. Вона відображає ключові сутності, які присутні у нашому проекті.

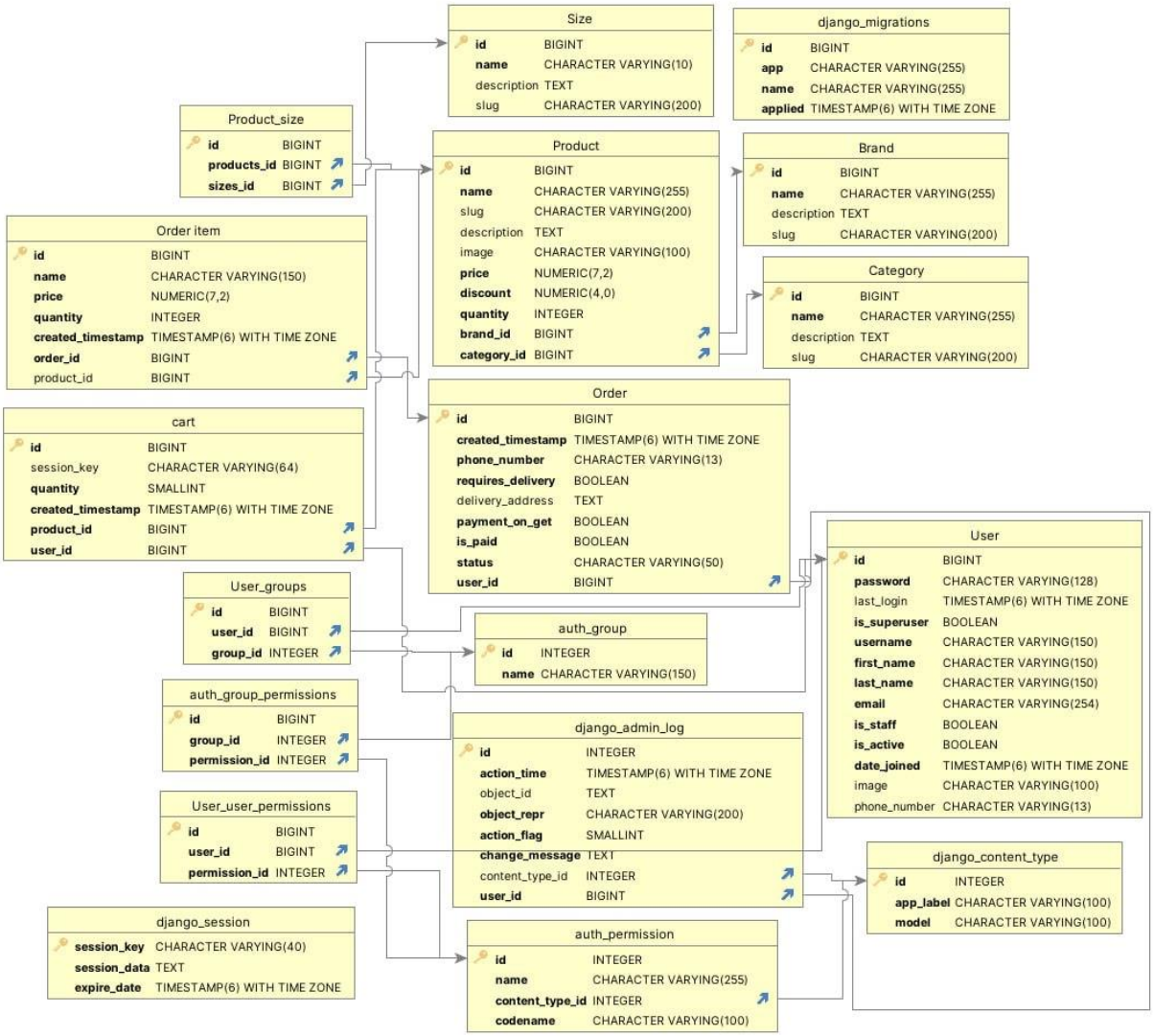


Рисунок 3.15 – Модель «сутність-зв'язок»

4 ПРАКТИЧНА РЕАЛІЗАЦІЯ ПРОЕКТУ

4.1 Реалізація та опис функціоналу клієнтського додатку системи

Розроблено інтернет-магазин ексклюзивного одягу, який надає можливість користувачу придбати одяг, аксесуари та взуття. Завдяки цьому сайту, магазин зможе залучити нових клієнтів, збільшити прибутковість та надасть можливість для клієнтів, що проживають за межами доступу до шоу-руму, купувати товари. Головна сторінка зображена на рисунку 4.1.

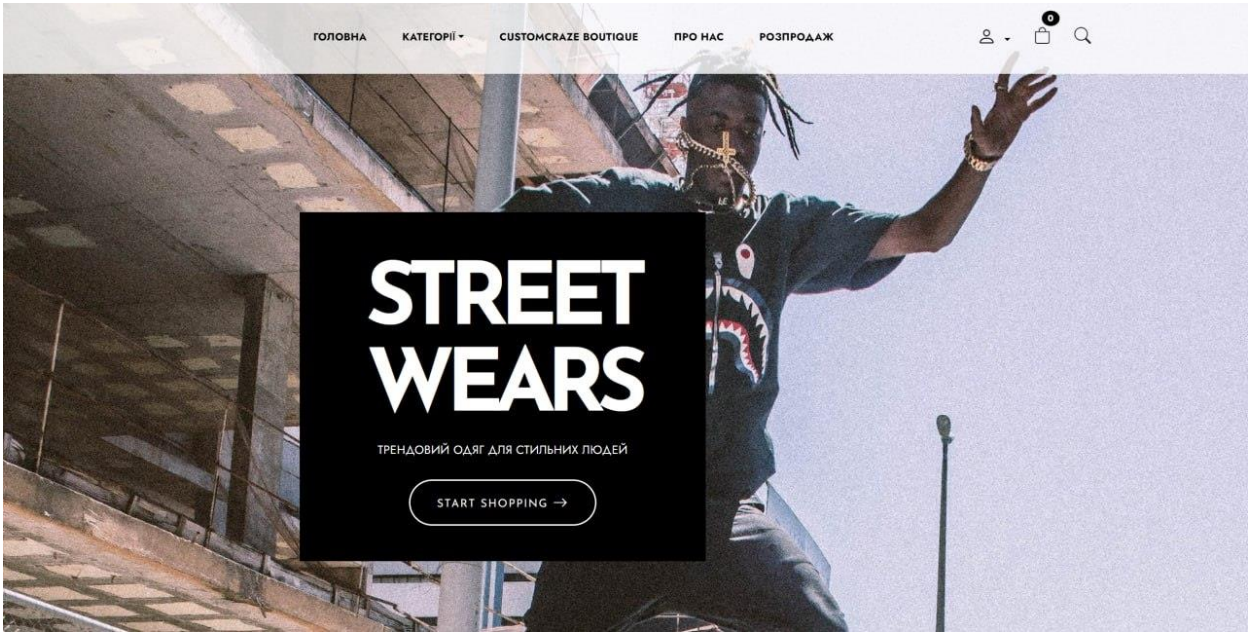


Рисунок 4.1 – Головна сторінка сайту

На головній сторінці користувачу надається зручна та інтуїтивно зрозуміла навігаційна панель. На цій панелі користувач має доступ до всіх основних розділів сайту, включаючи каталог товарів, особистий кабінет, кошик покупок та інші важливі функції. Це забезпечує користувачу легкий доступ до всіх можливих дій та інформації, що є на сайті. Частина фрагменту

коду, що відповідає за відображення цієї навігаційної панелі, представлений на рисунку 4.2.

```
<body>
  <nav class="main-menu d-flex navbar fixed-top
    <div class="container">
      <div class="offcanvas offcanvas-end" tabindex="-1"
        <div class="offcanvas-body justify-content-center">
            <li class="nav-item">
              <a href="{% url "main:index" %}"
              class="nav-link mx-2 active">Головна</a>
            </li>
            <li class="nav-item dropdown">
              <a class="nav-link mx-2 dropdown-toggle"
              role="button" id="pages"
              data-bs-toggle="dropdown"
                aria-expanded="false">Категорії</a>
              <ul class="dropdown-menu"
              aria-labelledby="pages">
                {% tag_categories as categories %}
                {% for category in categories %}
                  <li><a href="{% url "catalog:index"
                  category.slug %}" class="dropdown-item">
                  {{category.name}}</a></li>
                {% endfor %}
              </ul>
            </li>
            <div class="main-logo"><a href="{% url
            "main:index" %}" class="nav-link mx-2
            active">CustomCraze Boutique</a>
            </div>
            <li class="nav-item">
```

Рисунок 4.2 – Фрагмент коду навігаційної панелі

Сторінка каталогу переносить користувача до списку доступних товарів. На цій сторінці користувачеві залишається доступною навігаційна панель, що дозволяє легко переміщуватися по сайту. Візуальне оформлення сторінки каталогу представлено на рисунку 4.3.

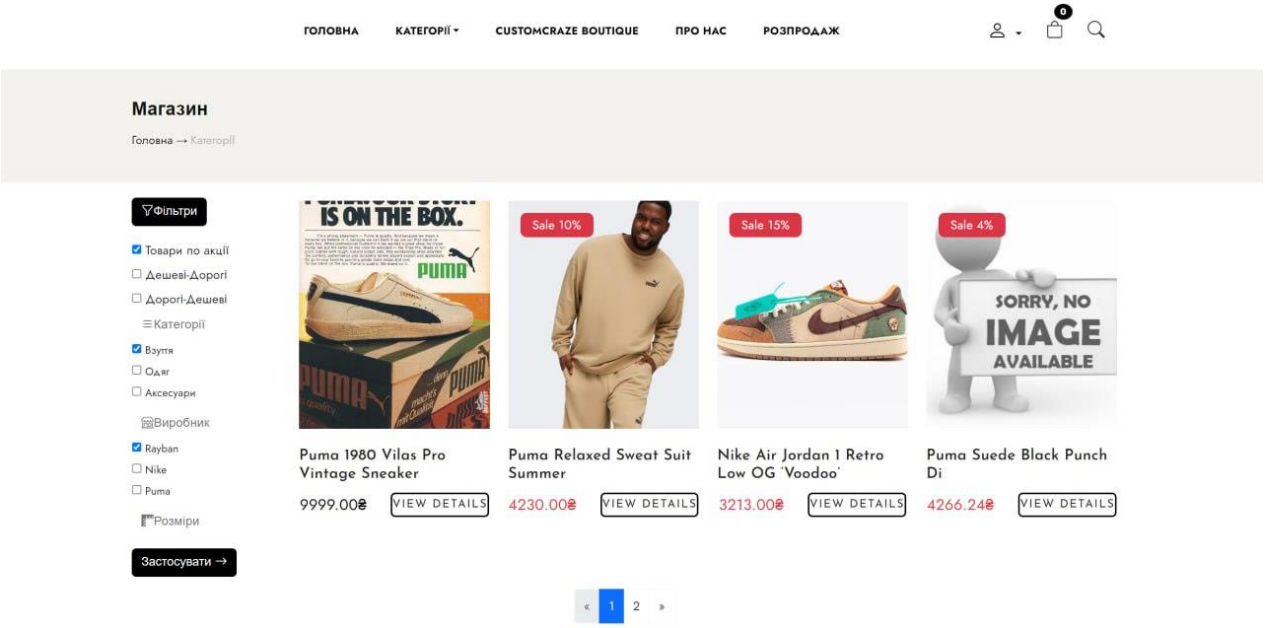


Рисунок 4.3 – Сторінка «Каталог»

Крім того, для зручності користувачів передбачено систему фільтрів, що дозволяє здійснювати більш точний пошук товарів за різними критеріями, такими як ціна, бренд, розмір та інші параметри. Це забезпечує швидкий доступ до необхідної інформації та спрощує процес вибору товару. Ознайомитися з фрагментом коду, що відповідає за відображення товарів на сайті, можна на рисунку 4.4

Переходимо до сторінки «Про нас», яка розповідає про історію магазину, його утворення та місію. На цій сторінці користувачі можуть дізнатися більше про команду проекту, побачити інформацію про ключових співробітників та їхні ролі в компанії. Також тут представлено бренди, що співпрацюють з магазином, що демонструє рівень партнерства та якість товарів, що пропонуються. Вигляд сторінки «Про нас» зображено на рисунку 4.5.

```
<section class="shop">
  <div class="container">
    <div class="row">
      <div class="col-lg-10">
        <div class="row">
          {% for product in goods %}
          <div class="col-md-6 col-lg-3 my-4">
            {% if product.discount > 0 %}
              <div class="z-1 position-absolute rounded-3
              m-3 px-3 border-danger">Sale {{ product.
              discount }}%</div>
            {% endif %}
            <div class="product-item">
              <div class="image-holder" style="width: 100%;
              height: 300px;">
                {% if product.image %}
                <img src="{{ product.image.url }}" alt="{{
                product.name }}" class="product-image
                img-fluid">
                {% else %}
                <img src="{% static 'deps/image/
                No-image-available.jpg' %}" alt="..."
                class="product-image img-fluid">
                {% endif %}
              </div>
              <div class="product-detail d-flex flex-column
              align-items-center mt-4">
                <h4 class="product-title mb-2 fs-5">
                  <a href="{% url 'catalog:product' product.
```

Рисунок 4.4 – Фрагмент коду товару на сторінці каталогу

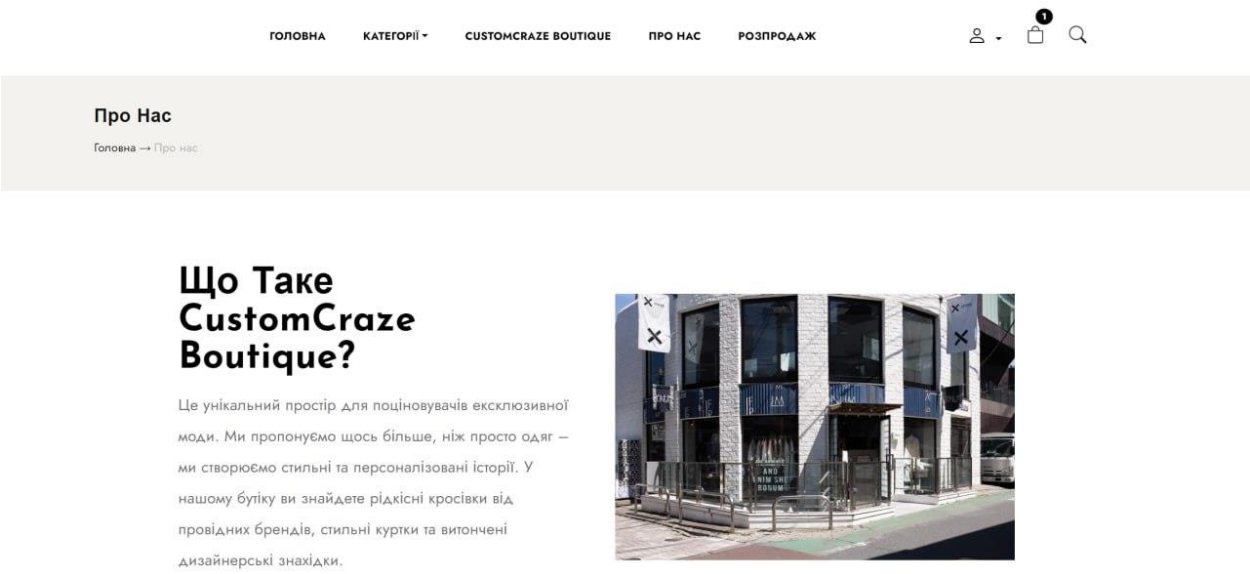


Рисунок 4.5 – Вигляд сторінки «Про нас»

Для ознайомлення з фрагментом коду, який відповідає за цю сторінку, дивіться рисунок 4.6.

```
<section class="about">
  <div class="container col-xxl-8 px-4 py-5">
    <div class="row flex-lg-row-reverse
    align-items-center g-5 py-5">
      <div class="col-10 col-sm-8 col-lg-6">
        <img src="{% static "deps/image/castom.jpg" %}"
        class="d-block mx-lg-auto img-fluid" alt=""
          width="700" height="500" loading="lazy">
      </div>
      <div class="col-lg-6">
        <h1 class="display-5 fw-bold lh-1 mb-3">Що таке
        CustomCraze Boutique?</h1>
        <p class="lead">Це унікальний простір для
        поціновувачів ексклюзивної моди. Ми пропонуємо
        щось більше, ніж
          просто одяг – ми створюємо стильні та
          персоналізовані історії. У нашому бутіку ви
          знайдете рідкісні кросівки
          від провідних брендів, стильні куртки та
          витончені дизайнерські знахідки.</p>
      </div>
    </div>
  </div>
</section>
```

Рисунок 4.6 – Фрагмент коду сторінки «Про нас»

Розглянемо сторінку реєстрації. На цій сторінці користувач може створити акаунт, завдяки якому він зможе оформлювати замовлення, переглядати історію покупок та користуватися іншими функціями сайту. Вигляд сторінки зображено на рисунку 4.7.

Для створення аккаунту користувачу необхідно вигадати нікнейм, ввести своє ім'я, прізвище та електронну пошту. Крім того, потрібно створити пароль, який повинен складатися не менше ніж з восьми символів, включаючи літери та цифри для підвищення безпеки. Це забезпечує захист акаунту від несанкціонованого доступу. Фрагмент коду, що відповідає за функціонал сторінки реєстрації, можна переглянути на рисунку 4.8.

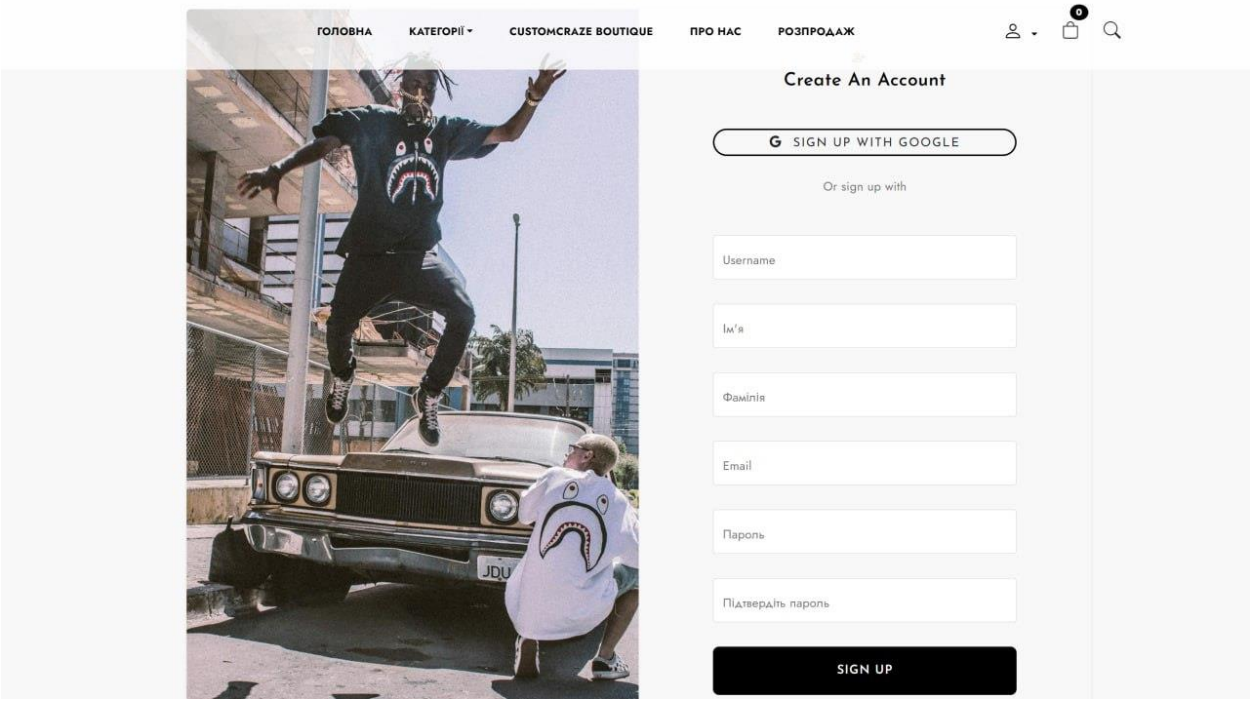


Рисунок 4.7 – Сторінка реєстрації

```
<section class="bg-light p-3 p-md-4 p-xl-5">
  <div class="container">
    <div class="row justify-content-center">
      <div class="col-12 col-xxl-11">
        <div class="card border-light-subtle shadow-sm">
                    <form action="{% url 'user:sign_up' %}"
                    method="post" onsubmit="return
                    validateForm()">
                    {% csrf_token %}
                    <div class="row gy-3 overflow-hidden">
                      <div class="col-12">
                        <div class="form-floating mb-3">
                          <input type="text"
                          class="form-control"
                          name="username" id="id_username"
                          placeholder="Введіть Username"
                          value="{{ form.username.value|
                          default_if_none:'' }}" required>
                            <label for="id_username"
                            class="form-label">Username</
                            label>
                            {% if form.username.errors %}
                              <div class="alert
                              alert-danger
                              alert-dismissible fade show"
                              role="alert">
                              {{ form.username.errors }}
                              </div>
                            {% endif %}
```

Рисунок 4.8 – Фрагмент коду реєстрації

Також подивимось на сторінку «Вхід». На цій сторінці користувач може ввести свої облікові дані для доступу до свого акаунту. Дизайн сторінки входу зображено на рисунку 4.9. Для входу користувачеві потрібно ввести свій нікнейм та пароль, які були вказані під час реєстрації. Фрагмент коду, що відповідає за функціонал сторінки входу, можна переглянути на рисунку 4.10.

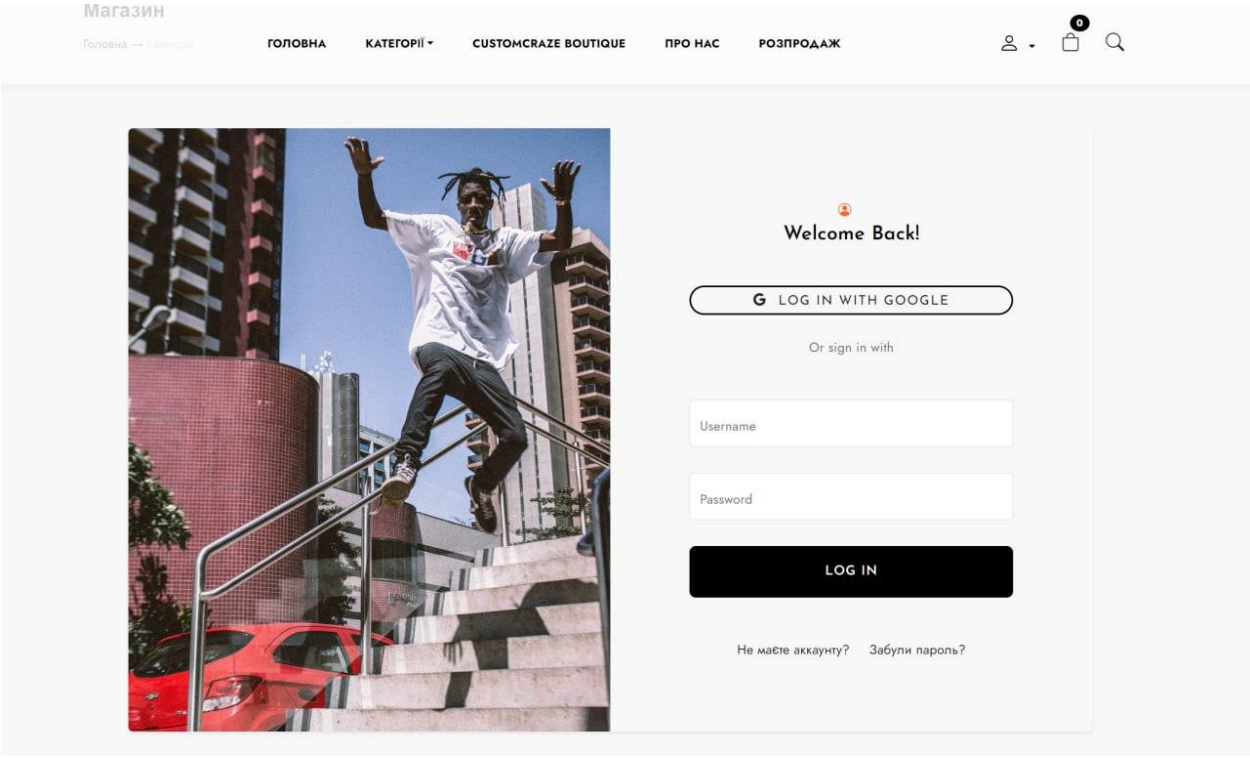


Рисунок 4.9 – Зовнішній вигляд сторінки «Вхід»

Перейдемо до важливої сторінки під назвою кошик. На цій сторінці користувач може переглянути товари, які він додав до кошика під час перегляду каталогу. Вигляд сторінки кошика зображено на рисунку 4.11. Користувач має можливість змінити кількість кожного товару або видалити товар з кошика, якщо він передумав його купувати. Також на сторінці кошика відображається загальна кількість та вартість обраних товарів, яка автоматично оновлюється при внесенні змін. Після остаточного вибору товарів користувач може перейти до оформлення замовлення, натиснувши відповідну кнопку.

```
<section class="bg-light p-3 p-md-4 p-xl-5">
  <div class="container">
    <div class="row justify-content-center">
      <div class="col-12 col-xxl-11">
        <div class="card border-light-subtle shadow-sm">
          <div class="row g-0">
            <div class="col-12 col-md-6">
              <img class="img-fluid rounded-start w-100
              h-100 object-fit-cover" loading="lazy"
                src="{% static "deps/image/banner1.jpg" %}"
                alt="Welcome back you've been missed!">
            </div>
            <div class="col-12 col-md-6 d-flex
            align-items-center justify-content-center">
              <div class="col-12 col-lg-11 col-xl-10">
                <div class="card-body p-3 p-md-4 p-xl-5">
                  <div class="row">
                    <div class="col-12">
                      <div class="mb-5 text-center">
                        <i class="bi bi-person-circle fa-2x
                        me-3" style="color: #ff6219;"></i>
                        <h4>Welcome Back!</h4>
                      </div>
                    </div>
                  </div>
                  <div class="row">
                    <div class="col-12">
                      <div class="d-flex gap-3 flex-column">
                        <a href="#!" class="btn btn-lg
```

Рисунок 4.10 – Фрагмент коду сторінки «Вхід»

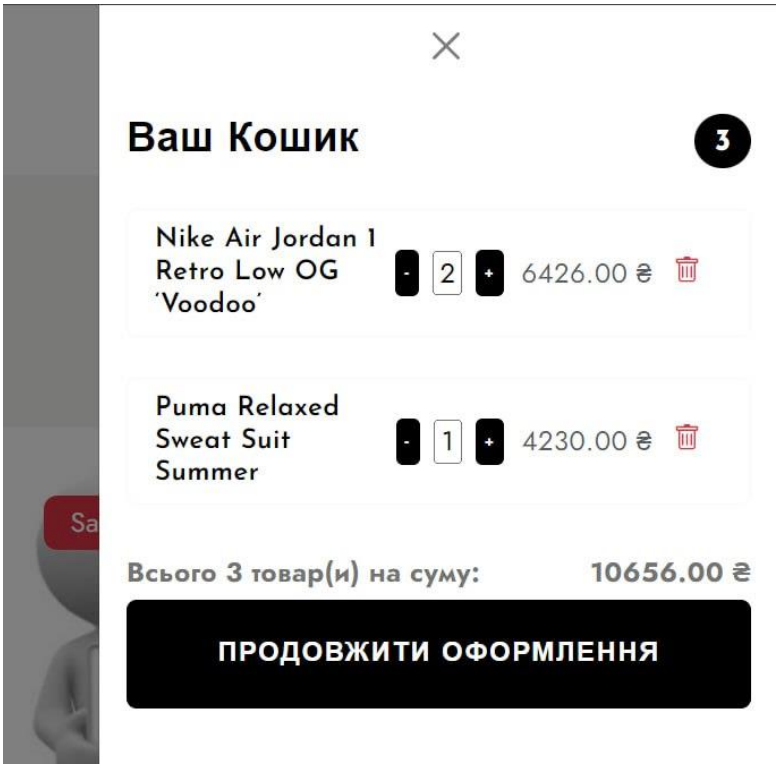


Рисунок 4.11 – Кошик товарів

4.2 Реалізація та опис функціоналу адміністративного додатку системи

Адміністративне керування дуже важливе для магазину. Для цього була використана адміністративна панель Django, яка була модифікована і доповнена відповідно до потреб проекту. Адміністратор має змогу додавати нові товари, створювати бренди, переглядати замовлення, а також за потреби клієнта змінювати їх. Також адміністратор може редагувати інформацію про товари, зокрема змінювати їх опис, ціни, наявність на складі та інші параметри. Важливою функцією є можливість створення та редагування категорій товарів, що дозволяє структурувати продукцію у каталозі. Для того, щоб почати роботу з адміністративною панеллю, треба зайти до аккаунту адміністратора, що зображено на рисунку 4.12.

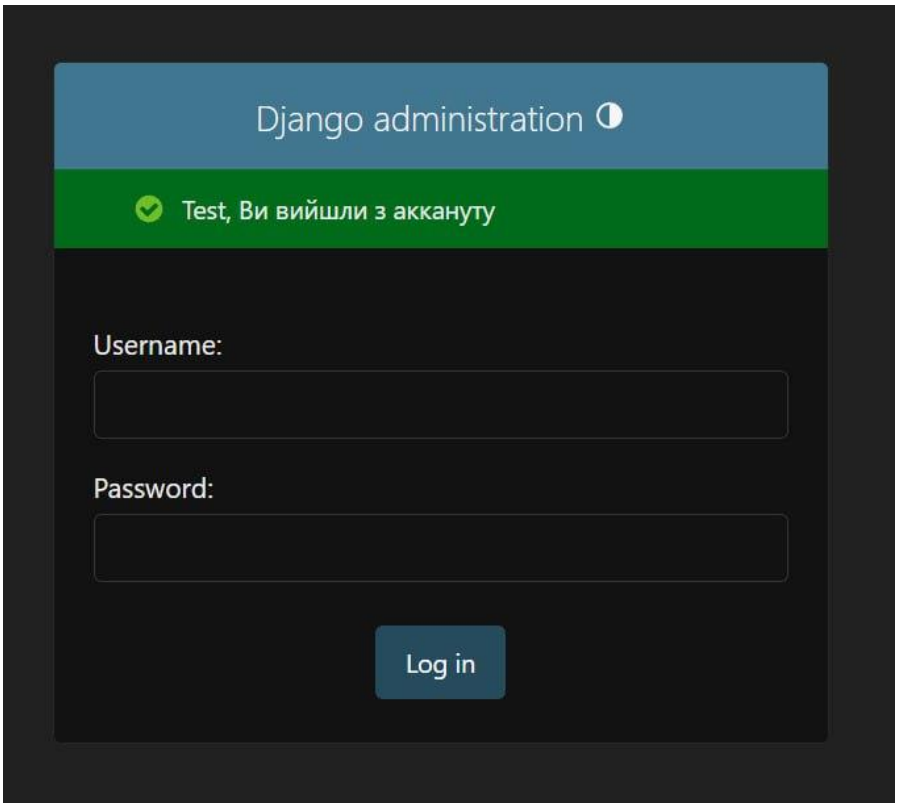


Рисунок 4.12 – Вхід до адміністративної панелі

Після того як адміністратор ввів правильні дані для входу, для нього стане доступною сторінка зі всіма необхідними інструментами для функціонування сайту. На цій сторінці адміністратор може побачити зручну панель навігації, яка надає швидкий доступ до основних функцій системи. Панель навігації містить такі розділи, як «Товари», «Замовлення», «Користувачі», «Кошики». Вигляд цієї панелі зображено на рисунку 4.13.

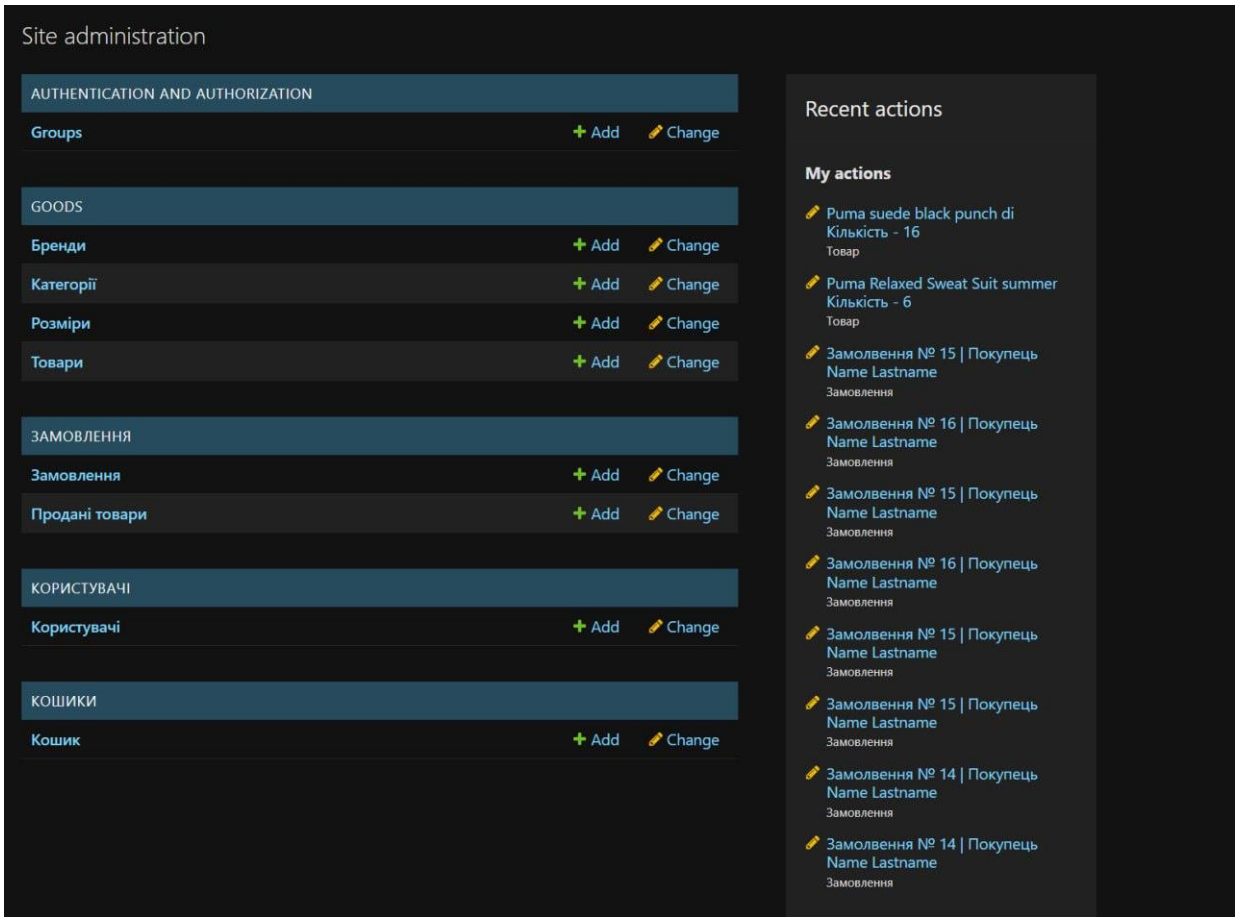


Рисунок 4.13 – Загальний вигляд адміністративної панелі

Розглянемо розділ «Товари». Через нього адміністратор може додавати нові товари, забезпечуючи оновлення асортименту магазину. У цьому розділі адміністратор має можливість вказувати ціну товару, додаючи актуальну вартість. Також можна завантажувати фото товарів.

Крім того, адміністратор може встановлювати знижки на товари, що допомагає стимулювати продажі. Адміністратор також може вказувати кількість товару на складі, що дозволяє відстежувати наявність продукції та забезпечувати своєчасне поповнення запасів. Ознайомитися з виглядом цієї панелі, можна на рисунку 4.14.

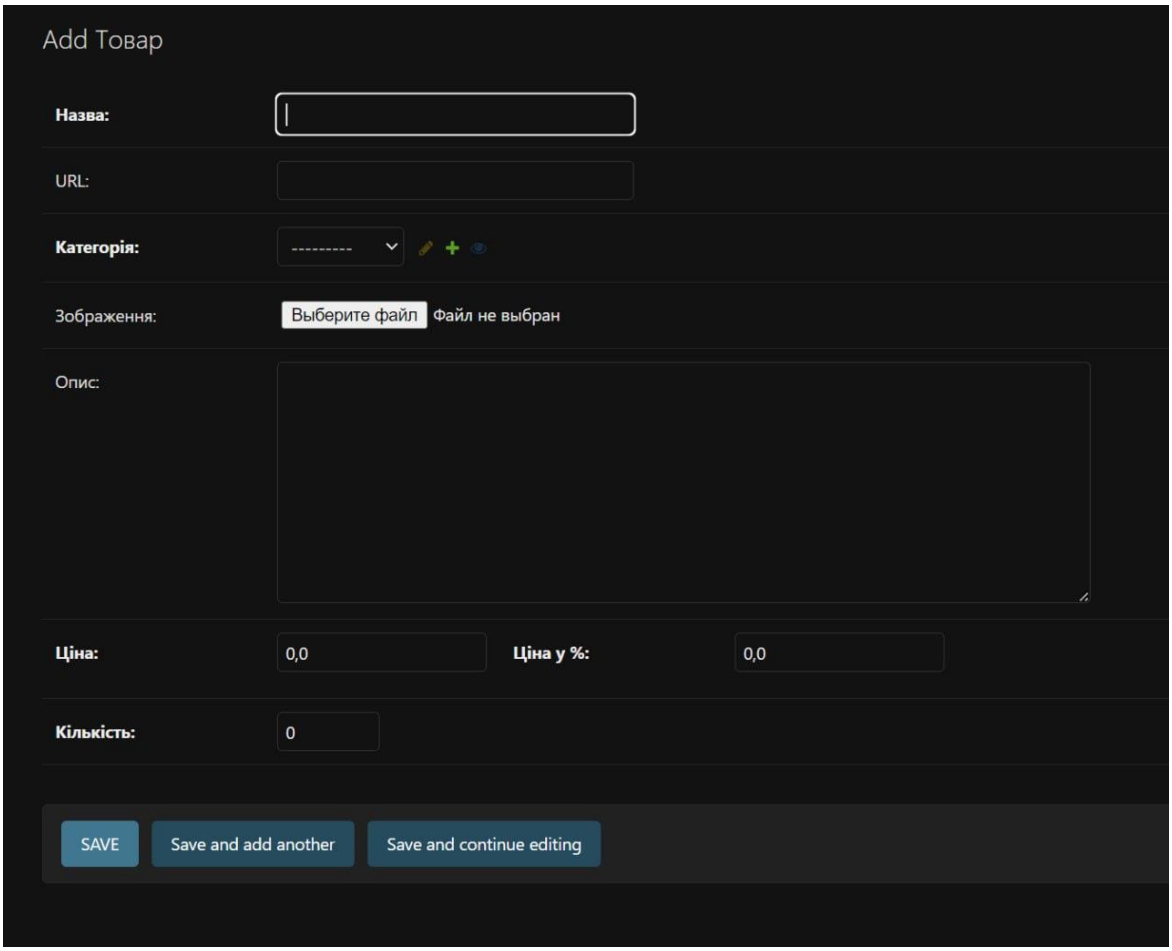


Рисунок 4.14 – Панель додавання товарів

Подивимось на розділ «Замовлення». Тут відображаються всі замовлення, що були створені користувачами. Адміністратор має змогу переглядати деталі кожного замовлення, включаючи список замовлених товарів, їх кількість та загальну вартість. Також тут можна побачити, які замовлення вже завершені, а які ще знаходяться в обробці, що допомагає адміністратору ефективно управляти процесом виконання замовлень.

У цьому розділі також вказується, чи обрав клієнт доставку, та який спосіб оплати був обраний. Інформація про дату створення замовлення дозволяє відстежувати хронологію замовлень та визначати їх пріоритетність.

Крім того, адміністратор має можливість редагувати статус замовлення, змінювати дані доставки або контактну інформацію клієнта у разі виникнення проблем у клієнта. Вигляд цієї панелі, зображено на рисунку 4.15.

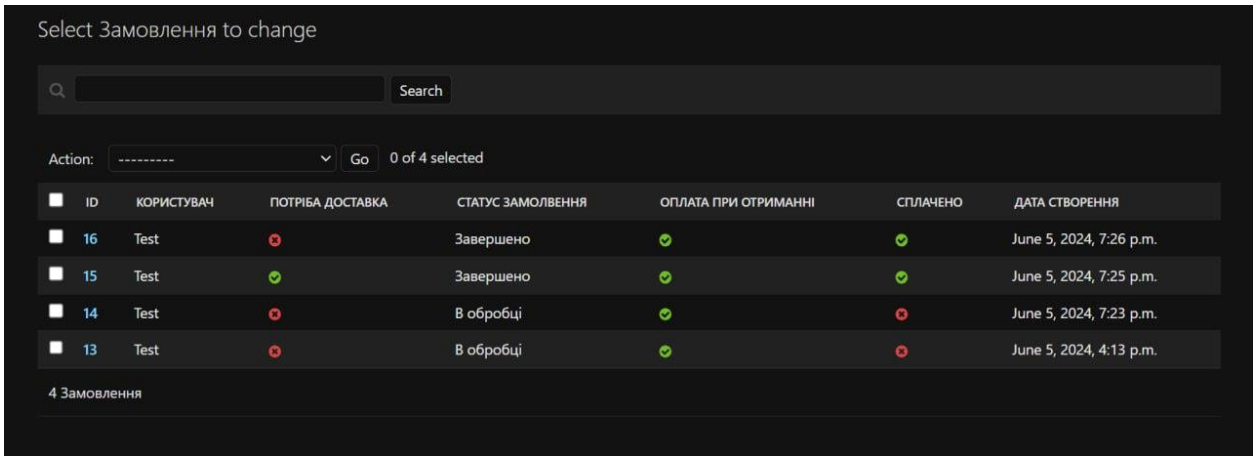


Рисунок 4.15 – Розділ «Замолвення»

Розділ «Користувачі» надає інформацію про всіх зареєстрованих користувачів та адміністраторів системи. У цьому розділі можна переглянути дані користувачів, які вони ввели під час реєстрації, такі як ім'я, прізвище, електронна пошта та інші контактні дані, за винятком паролів. Паролі користувачів зашифровані, що гарантує їхню безпеку і конфіденційність, тому ні адміністратор, ні будь-який інший користувач не зможуть побачити їх у відкритому вигляді. Також у цьому розділі відображається час останнього входу на сайт, що дозволяє відстежувати активність користувачів.

Крім того, адміністратор може бачити наповненість кошика кожного користувача. Це є гарною практикою, вона дозволяє нам уявити, що планує придбати користувач і надати йому спеціальні пропозиції. Також відображаються вже виконані замовлення, якщо такі були. Це дає змогу

адміністратору бачити історію покупок користувача, що може бути корисним для вирішення питань гарантії, повернень або обробки повторних замовлень. Дана панель є на рисунку 4.16.

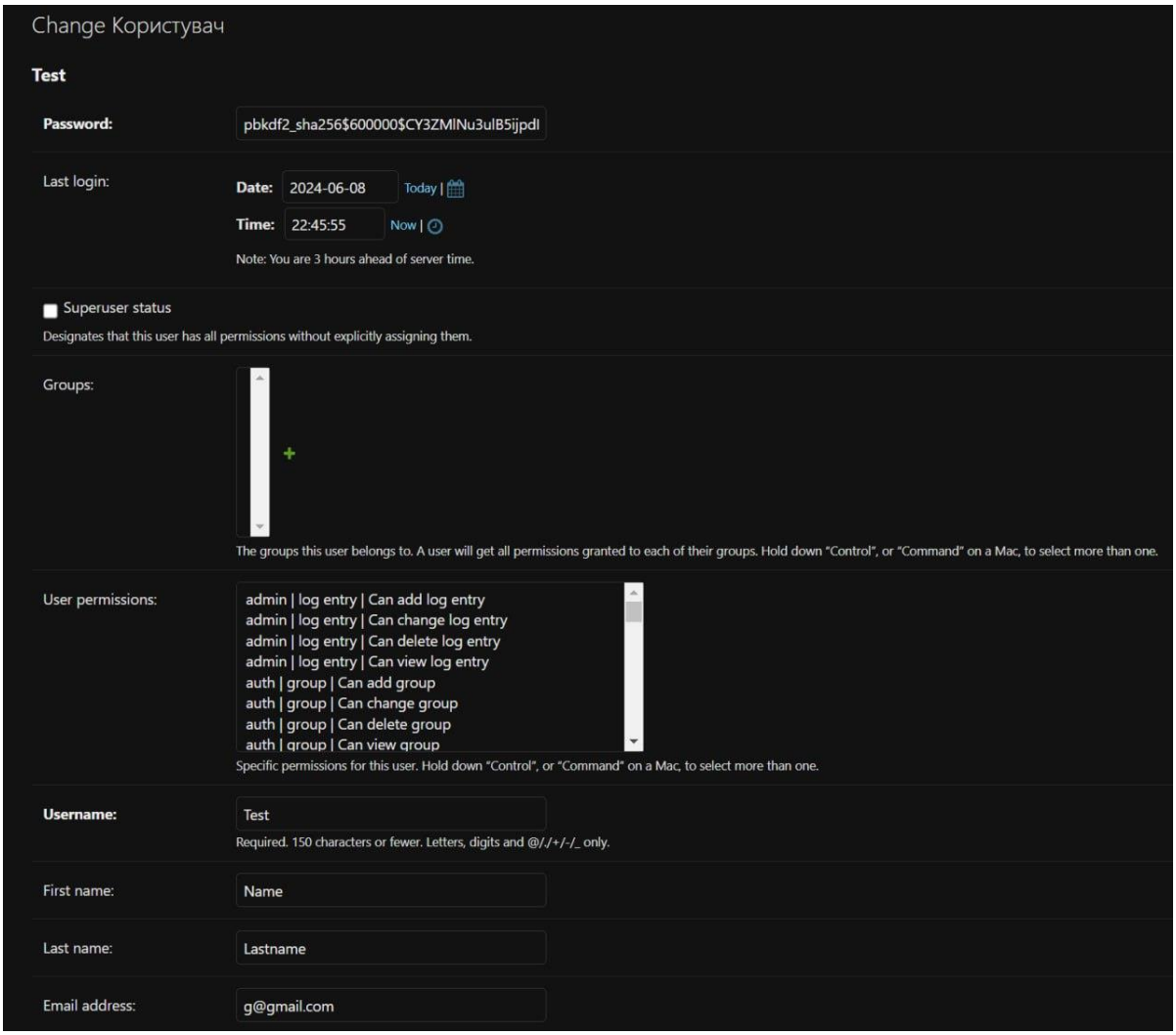


Рисунок 4.16 – Панель «Користувач»

ВИСНОВКИ

У рамках даного дипломного проекту було розроблено повноцінний інтернет-магазин ексклюзивного одягу, взуття та аксесуарів, який відповідає сучасним вимогам ринку та забезпечує високий рівень користувацького досвіду. Під час роботи над проектом були виконані численні завдання, починаючи від аналізу предметної області і закінчуючи реалізацією адміністративної панелі для управління контентом сайту.

На першому етапі проекту був проведений ретельний аналіз предметної області, що включав огляд існуючих конкурентів на ринку України, таких як Agnostic, Original Store та Del Garda. Було вивчено їх функціональні можливості, особливості інтерфейсу та основні елементи головних сторінок. Це дозволило визначити ключові аспекти, які необхідно врахувати при розробці власного сайту.

Вибір архітектури системи та програмних засобів реалізації був здійснений на основі сучасних технологій, таких як Django, PostgreSQL, HTML, CSS, JavaScript, jQuery, Bootstrap 5 та AJAX. Використання Django як фреймворку для серверної частини обумовлено його багатофункціональністю, надійністю та широкими можливостями для швидкої розробки веб-додатків. PostgreSQL була обрана як система управління базами даних завдяки її потужності, гнучкості та підтримці масштабованості. Клієнтська частина була реалізована з використанням HTML, CSS, JavaScript, jQuery, Bootstrap 5 та AJAX, що забезпечує високий рівень інтерфейсу користувача та швидку взаємодію з сервером.

Процес проектування включав розробку діаграм інтерфейсів та навігації, функціональне моделювання за допомогою методології SADT та моделювання бізнес-процесів за допомогою Workflow Diagramming. Це дозволило створити чітку структуру системи, визначити основні функціональні блоки та їх взаємодію, а також забезпечити логічну послідовність дій користувачів.

Проектування бази даних було здійснено на основі трирівневої архітектури, що забезпечує високий рівень безпеки та масштабованості. Було розроблено моделі бази даних, які відображають основні сутності системи та їх взаємозв'язки. Це дозволило забезпечити ефективне управління даними та оптимізувати процеси їх обробки.

Практична реалізація проекту, включала розробку як користувацької, так і адміністративної частин. Користувачі отримали можливість швидко та зручно знаходити необхідні товари, здійснювати покупки, користуватися фільтрами для точного пошуку. Реалізація адміністративної панелі на основі Django забезпечила зручне управління контентом, додавання нових товарів, створення брендів, перегляд замовлень та управління користувачами.

Загалом, створений інтернет-магазин відповідає сучасним вимогам ринку, забезпечує високий рівень безпеки та продуктивності, а також надає зручний інтерфейс для користувачів та адміністрації. Виконані завдання дозволили створити конкурентоспроможний продукт, який має всі необхідні функції для успішного функціонування на ринку ексклюзивного одягу.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Документація Django. URL: https://docs.djangoproject.com/en/4.2/
2. David Maxwell, Leif Azzopardi – «Tango with Django 4», 76 с.
3. Інтернет-магазин «Agnostic». URL: https://www.agnostic.ua
4. Інтернет-магазин «Original Store». URL: https://originalstore.com.ua/uk/
5. Інтернет-магазин «Del Garda». URL: https://delgarda.com.ua
6. JQuary/Ajax документація. URL: https://api.jquery.com/category/ajax/
7. Популярність баз даних. URL: https://db-engines.com/en/ranking_trend
8. Використання PostgreSQL з Django. URL: https://www.enterprisedb.com/postgres-tutorials/how-use-postgresql-django
9. MVT модель у Django. URL: https://angelogentileiii.medium.com/basics-of-django-model-view-template-mvt-architecture-8585aecffbf6
10. Популярність використання веб-технологій. URL: https://w3techs.com\
11. Що таке Web3 техгологія. URL: https://learn.metamask.io/uk-UA/lessons/what-is-web3