МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Кафедра інженерії програмного забезпеченення

**КУРСОВА РОБОТА**

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

з дисципліни: «Веб-технології. Частина 2»

на тему:

**«Інтернет-магазин електроніки»**

студента ІІ курсу групи ІПЗ-20-4

спеціальності 121 «Інженерія програмного забезпечення»

Дубинченка Богдана Миколайовича

(прізвище, ім’я та по-батькові)

Керівник ст. викладач Кафедри ІПЗ, Олена Геннадіївна Чижмотря

Дата захисту: " \_\_\_ " \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ р.

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оцінка: ECTS \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Граф М.С. \_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кузьменко О.В. .

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Чижмотря О.Г. \_

(підпис) (прізвище та ініціали)

Житомир – 2022

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Факультет інформаційно-комп’ютерних технологій

Кафедра інженерії програмного забезпеченення

Освітній рівень: бакалавр

Спеціальність 121 «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»

В.о зав кафедри ІПЗ

А.В. Морозов

“\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_2021 р.

ЗАВДАННЯ

НА КУРСОВИЙ ПРОЕКТ СТУДЕНТУ

Дубинченку Богдану Миколайовичу

1. Тема роботи: Інтернет-магазин електроніки

керівник роботи: Морозов А.В.

1. Строк подання студентом: “18” січня 2022р.
2. Вихідні дані до роботи: Розробити веб-сайт для продажу електроніки – розробити та реалізувати систему адміністрування для сайту.
3. Зміст розрахунково-пояснювальної записки (перелік питань, що підлягають розробці)
   * + 1. Постановка завдання
       2. Аналіз аналогічних розробок
       3. Алгоритми роботи програми

4. Опис роботи програми

5. Програмне дослідження

1. Перелік графічного матеріалу(з точним зазначенням обов’язкових креслень)

1. Презентація до КР\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Посилання на репозиторій: <https://gitlab.com/2020-2024/ipz-20-4/dubinchenko-bogdan/kursova-2022>

1. Консультанти розділів проекту (роботи)

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Прізвище, ініціали та посади консультанта | Підпис, дата | |
| завдання  видав | завдання прийняв |
| 1,2 | Чижмотря О.Г., ст. викладач каф. ІПЗ |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Дата видачі завдання “ 15 ” вересня 2021 р.

|  |  |  |  |
| --- | --- | --- | --- |
| № з/п | Назва етапів курсової роботи | Термін виконання етапів роботи | Примітки |
| 1 | Постановка задачі | 10.12.21-11.12.21 | Виконано |
| 2 | Пошук, огляд та аналіз аналогічних розробок | 11.12.21-13.12.21 | Виконано |
| 3 | Формулювання технічного завдання | 13.12.21-17.12.21 | Виконано |
| 4 | Опрацювання літературних джерел | 17.12.21-23.12.21 | Виконано |
| 5 | Проектування структури | 23.12.21-25.12.21 | Виконано |
| 6 | Написання програмного коду | 25.12.21-07.01.22 | Виконано |
| 7 | Відлагодження | 07.01.22-10.01.22 | Виконано |
| 8 | Написання пояснювальної записки | 10.01.22 -15.01.22 | Виконано |
| 9 | Захист | 18.01.22 |  |

**КАЛЕНДАРНИЙ ПЛАН**

**Студент** \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дубинченко Б.М.

(підпис) (прізвище та ініціали)

**Керівник проекту** \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Морозов А.В.

(підпис) (прізвище та ініціали)

**РЕФЕРАТ**

Пояснювальна записка до курсового проекту на тему «Розробка інтернет-магазину електроніки» складається з переліку умовних скорочень, вступу, трьох розділів, висновків, списку використаної літератури та додатку.

Текстова частина викладена на 34 сторінках друкованого тексту.

Пояснювальна записка має 5 сторінок додатків. Список використаних джерел містить 6 найменувань і займає 1 сторінку. В роботі наведено 25 рисунків. Загальний обсяг роботи – 41 сторінок.

У першому розділі було обґрунтовано створення сайту на тему “ Інтернет-магазин електроніки ”

У другому розділі проведено проектування і розробка сайту.

У третьому розділі проведено тестування програмного продукту.

Висновок містить в собі результати виконаної роботи при створенні сайту на тему “ Інтернет-магазин електроніки ”.

У додатку представлений лістинг розробленого програмного продукту.

Ключові слова:

SQLILE, JAVASCRIPT, PYTHON, DJANGO, Web-сайт, MVC, AJAX, АДМІН, МАГАЗИН, ІНТЕРНЕТ, ДАНІ.

Зміст

[**ВСТУП** 7](#_Toc93264402)

[**РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ** 8](#_Toc93264403)

[**1.1 Аналіз задачі, засобів та методів її вирішення** 8](#_Toc93264404)

[**1.2 Аналіз існуючого програмного забезпечення за тематикою курсової роботи.** 9](#_Toc93264405)

[**1.3 Технічне завдання на курсову роботу** 12](#_Toc93264406)

[**1. Загальне положення** 12](#_Toc93264407)

[**2. Підстава для розробки** 12](#_Toc93264408)

[**3. Вимоги до програми** 13](#_Toc93264409)

[**Висновки до першого розділу :** 15](#_Toc93264410)

[**РОЗДІЛ 2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ** 16](#_Toc93264411)

[**2.1 Проектування загального алгоритму роботи веб-додатку** 16](#_Toc93264412)

[Загальна схема роботи програми : 17](#_Toc93264413)

[**2.2 Розробка функціональних алгоритмів роботи програми** 18](#_Toc93264414)

[**2.3 Розробка програмного забезпечення** 19](#_Toc93264415)

[**Висновки до 2 розділу :** 24](#_Toc93264416)

[**РОЗДІЛ 3 ОПИС РОБОТИ З ДОДАТКОМ** 25](#_Toc93264417)

[**3.1 Опис роботи з додатком (Опис інтерфейсу)** 25](#_Toc93264418)

[**3.2 Тестування веб-додатку** 30](#_Toc93264419)

[**Висновки до 3 розділу :** 32](#_Toc93264420)

[**ВИСНОВКИ** 33](#_Toc93264421)

[**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ** 34](#_Toc93264422)

[**ДОДАТКИ** 35](#_Toc93264423)

[Додаток А 36](#_Toc93264424)

[Додаток Б 36](#_Toc93264425)

**ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ**

БД – База даних

КР – курсова робота

MVC – Модель–вигляд–контролер (або Модель–представлення–контролер, Model-view-controller, MVC) — архітектурний шаблон, який використовується під час проектування та розробки програмного забезпечення.

ПЗ – Програмне забезпечення

# **ВСТУП**

У цій курсовій роботі буде наведено процес розробки інтернет-магазину електроніки.

**Актуальність** роботи полягає в тому, що інтернет-магазини стають все більш популярними. По-перше це економія часу, не потрібно шукати потрібні магазини з певними товарами, все можна замовити не виходячи з дому. По-друге доступність, в інтернеті можна знайти будь-який товар навіть за нижчими цінами чім в звичайних магазинах.

**Метою** курсової роботи є дослідження особливостей проектування та реалізації cистеми адміністрування інтернет-магазину електроніки.

Завданням на курсову роботу є:

* аналіз теоретичних засад проектування та реалізації CMS;
* аналіз існуючих веб-сайтів на обрану вами тематику, визначення унікальності;
* визначення інформаційних потреб предметної області дослідження;
* розробка адаптивного інтерфейсу веб-сайту за допомогою HTML5, CSS3, JavaScript;
* розробка математичної та алгоритмічної моделі функціонування системи на основі БД;
* проектування бази даних за визначеною предметною областю;
* написання серверної частини;
* реалізація CMS.

**Об’єктом дослідження** є методи та засоби розробки CMS для інтернет-магазину електроніки.

**Предмет дослідження** є використання веб-технологій для забезпечення інформаційних потреб в сферах освіти та саморозвитку.

# **РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ**

## **1.1 Аналіз задачі, засобів та методів її вирішення**

Задача полягає в тому, шо потрібно:

1. Визначити потреби звичайного користувача
2. Зрозуміти основні принципи та алгоритми роботи інтернет-магазину
3. Продумати можливість додавання та редагування товарів і категорій.
4. Реалізувати сортування та пошук товарів
5. Продумати можливість створення нових адмінів
6. Створити коректну базу даних
7. Продумати дизайн сайту

Для розробки сайту буде використано мови програмування Python + фреймворк Django, машрутизатор Jinja2 та JS.

Для КР роботи використано таке середовище розробки, як PyCharm, тому що він підтримує всі необхідні в ході розробки інструменти для зручної та швидкої роботи.

Для написання серверної частини сайту було обрано мову програмування Python та фреймворк Django. Python доволі проста мова програмування з зручною динамічною строгою типізацією, а Django зручний у вивчені та головне, що він використовує шаблон проектування MVC.

У якості БД було обрано Sqlite.

Для клієнтської частини сайту було обрано мову програмування JS, для асинхронності Ajax(Asynchronous Javascript and XML), який дозволяє не перезавантажувати всю сторінку.

## **1.2 Аналіз існуючого програмного забезпечення за тематикою курсової роботи.**

При аналізі вже існуючих веб-сайтівза тематикою курсової роботи було виявлено декілька проектів. Всі вони схожі і мають подібну реалізацію, адже всі вони є інтернет магазинами, які призначені для продажу електроніки.

Aналоги інтернет-магазину за заданою темою:

Приклад 1: назва - Lemans, посилання на джерело - <https://lemanso.com.ua/>

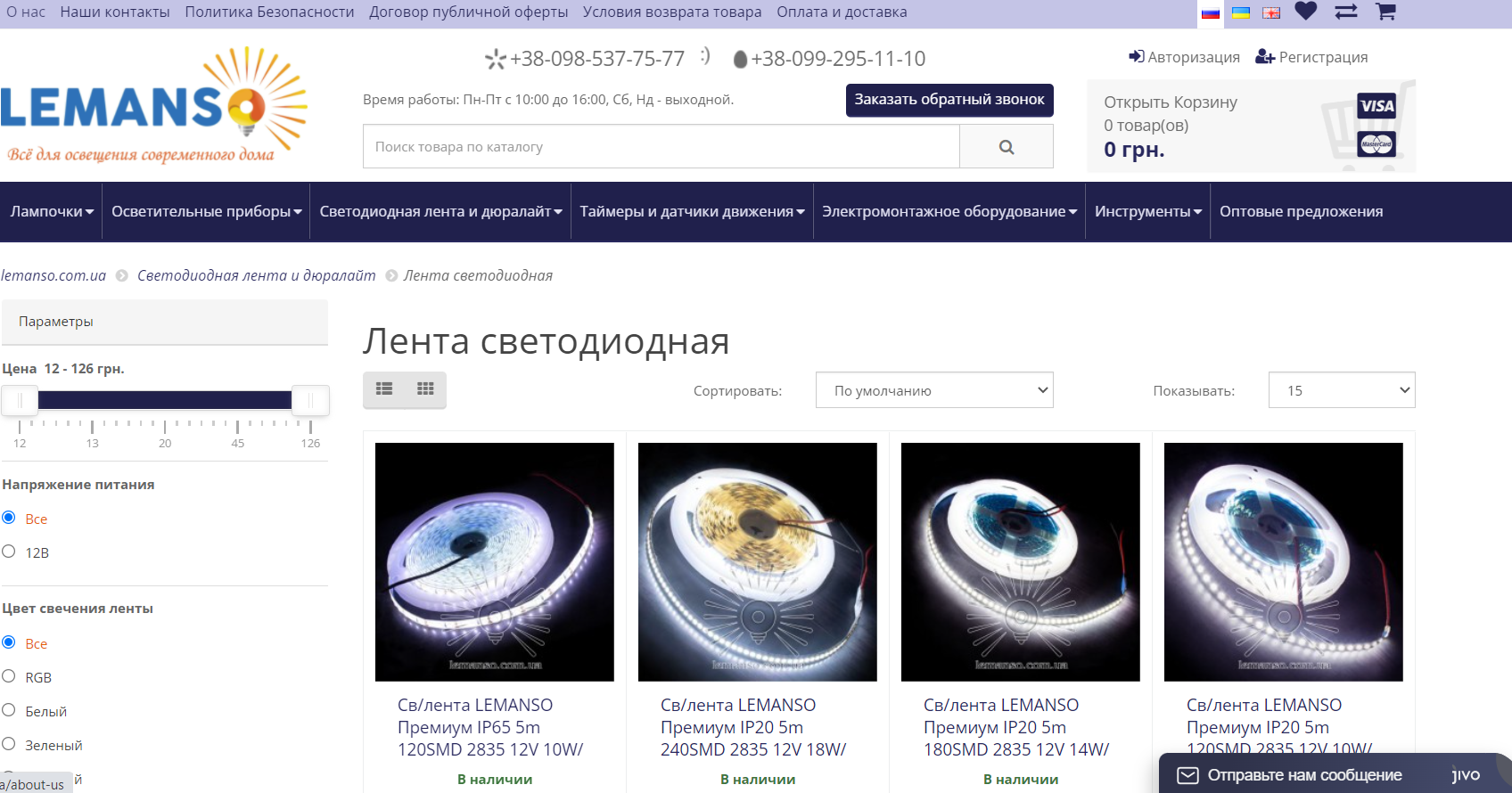


Рисунок 1.1 Lemans

Інтернет-магазин Lemans, сайт який продає товари для освітлення дому. Має функцію зворотного зв’язку з відвідувачами. Серед мінусів можна виділити навантажений інтерфейс.

Основні плюси:

1. Широкий вибір параметрів для сортування
2. Пошук за запитом
3. Присутній адаптив

Основні мінуси:

1. Навантажений інтерфейс
2. Проблема з фільтрами при адаптиві

Приклад 2: назва – brille, посилання на джерело - <https://www.brille.ua/>

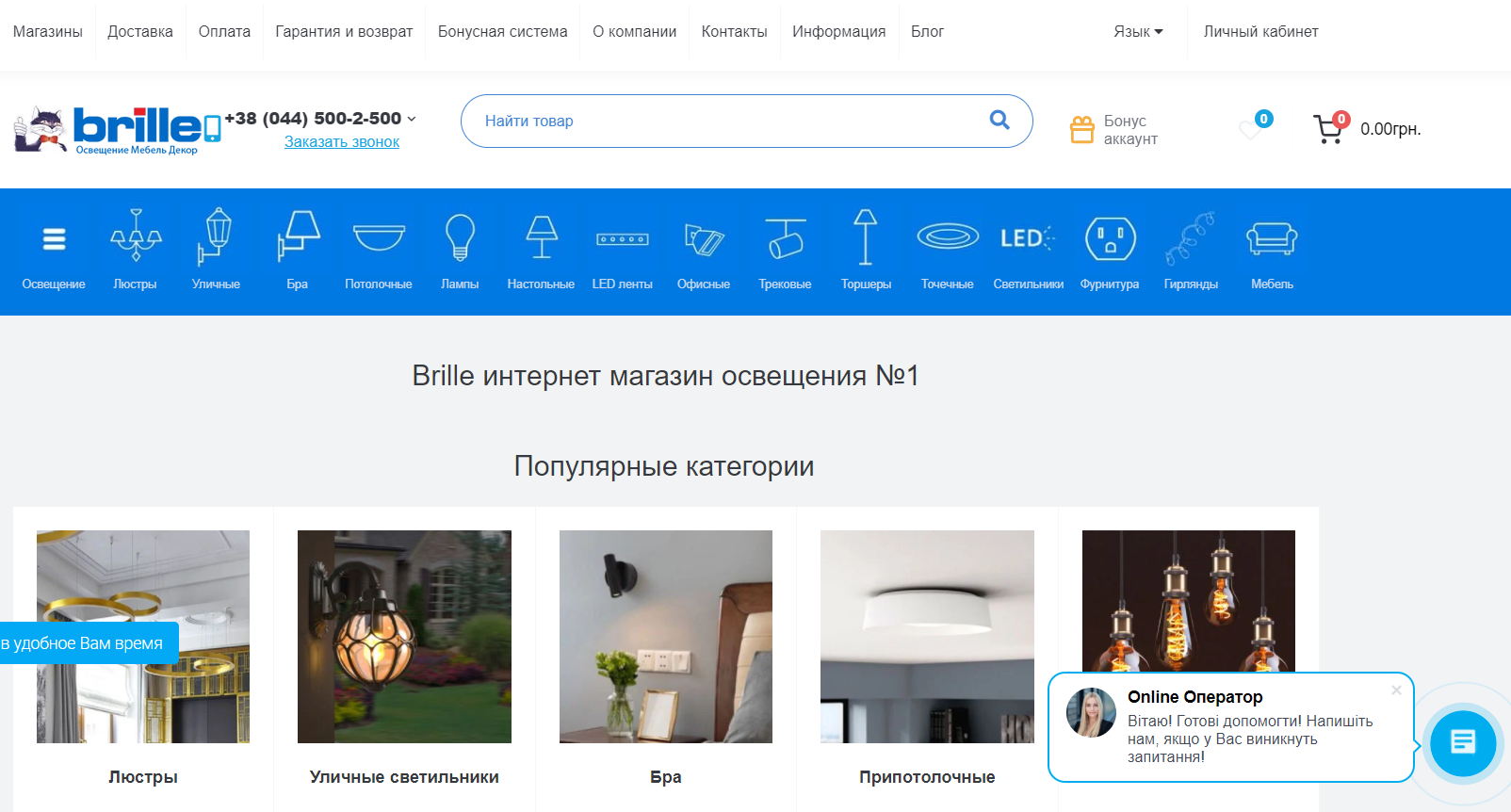


Рисунок 1.2 brile

Інтернет магазин brile, має великий асортимент товарів. Для кожної категорії товарів присутні фільтри. На сайті є можливість зв’язатися з оператором, які завжди допоможуть з замовленням. Один з мінусів це не стабільна робота в списках категорій.

Основні плюси:

1. Великий вибір параметрів для сортування
2. Пошук за запитом
3. Присутній адаптив
4. Стильний дизайн

Основні мінуси:

1. Не стабільна робота в списках категорій

Приклад 3: назва – italТехно , посилання на джерело – https://ital-tecno.com.ua/ru/

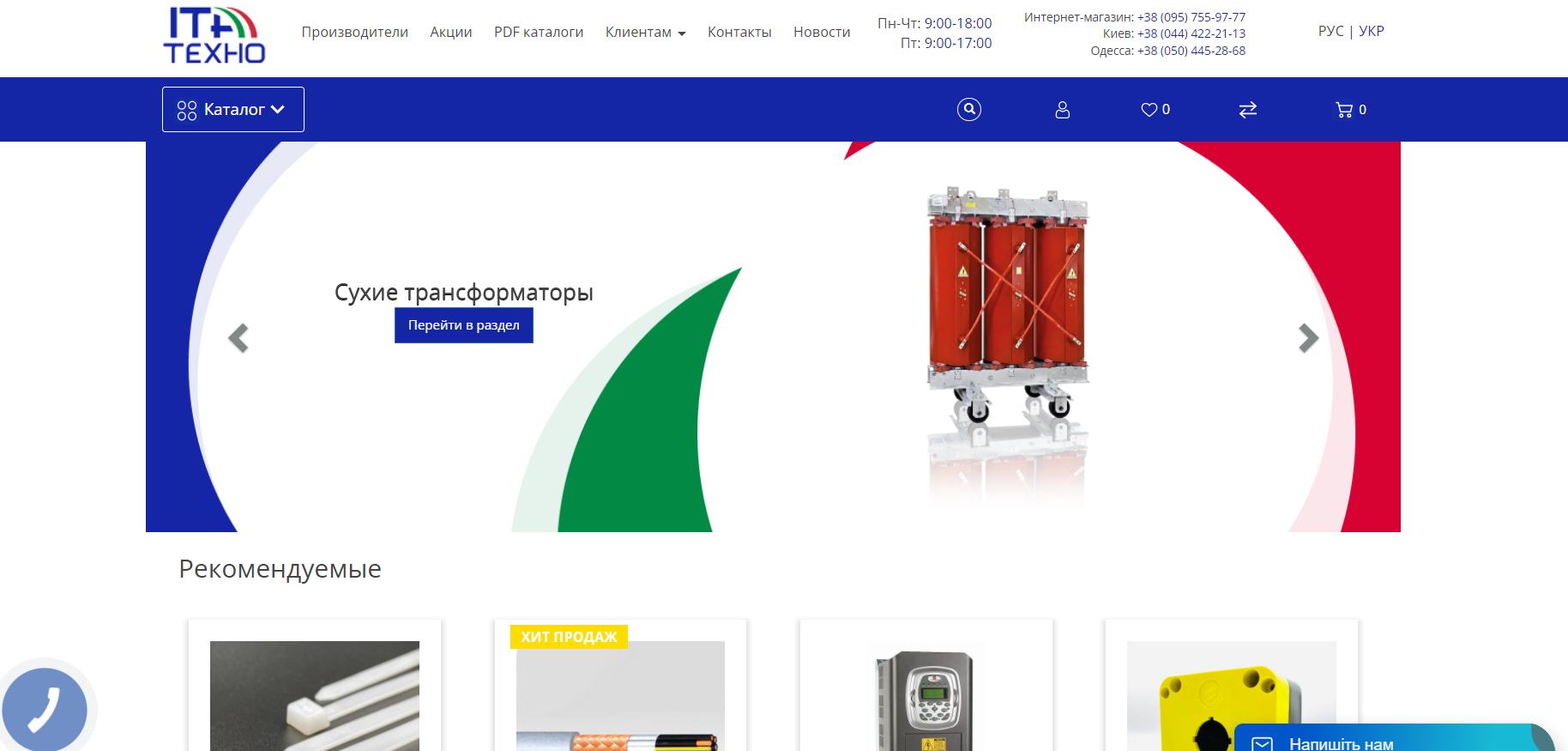


Рисунок 1.3 italТехно

Інтернет-магазин italТехно, має простий та зрозумілий інтерфейс. Присутній адаптив, але трохи не зручно переглядати товари.

Основні плюси:

1. Великий вибір параметрів для сортування
2. Присутній адаптив
3. Порівняння товарів

Актуальним напрямком реалізації власного інтернет-магазину є створення веб-сайту з логікою подібного до реалізованої у всіх представлених прикладах, реалізувати користувацький інтерфейс який представлений у третьому прикладі, уникаючи помилок в першому прикладі. Також придумати сучасний дизайн сайту схожий до дизайну brille та реалізувати адаптив.

## **1.3 Технічне завдання на курсову роботу**

### **1. Загальне положення**

#### **1.1 Найменування програмного засобу**

Повне найменування програмної системи: "Розробка інтернет-магазину електроніки" (надалі "веб-додаток"). Коротка назва програмної системи - " Інтернет-магазин"

#### **1.2. Призначення розробки та область застосування**

Веб-сайт "Інтернет-магазин електроніки" призначений для розміщення каталогу електроніки, їх пошуку, та продажу.

Система може бути впроваджена у роботу інтернет-магазину.

Веб-сайт " Інтернет-магазин електроніки" дозволить швидко переглядати інформацію про електротовари, та забезпечить керівництву підприємства своєчасною та повною інформацією про актуальний стан замовлення покупок інтернет-магазину у повному обсязі, а також надасть користувачам можливість вибору електротоварів різних каталогів в залежності від їх потреби.

#### **1.3. Найменування розробника та замовника.**

Розробник даного продукту - студент групи ІПЗ-20-4 Дубинченко Богдан Миколайович (надалі "розробник"). Замовник програмного продукту – кафедра інженерія програмного забезпечення Житомирського державного технологічного університету в межах виконання курсової з дисципліни «Web-тееології» Чижмотря Олена Геннадіївна, Морозов Андрій Васильович

### **2. Підстава для розробки**

#### **2.1. Документ на підставі якого ведеться розробка**

Робота ведеться на підставі навчального плану за напрямом 121 «Інженерія програмного забезпечення».

### **3. Вимоги до програми**

#### **3.1. Вимоги до функціональних характеристик.**

#### **3.1.1. Загальні вимоги**

Веб-додаток має забезпечувати:

• можливість дистанційної роботи з робочих станцій локальної та глобальної мережі підприємства;

• постійний доступ користувачів веб-додатку;

• оформлення замовлення;

• організацію управління сайтом;

• можливість доступ до бази даних;

#### **3.1.2. Склад виконуваних функцій**

Розробити інтернет-магазин електроніки, що підтримує виконання наступних операцій:

1. Пошук електротоварів.
2. Зміна інформації про товар.
3. Додавання запису про товар:

* Назва
* Картинка
* Опис
* Ціна
* Знижка
* Ціна зі знижкою
* Кількість
* Категорія

1. Фільтрація книг за параметрами:

* Назва
* Категорія
* Ціна

1. Перегляд інформації про електротовар

#### **3.1.3. Організація вхідних і вихідних даних**

Вхідними даними є інформація про електротовар (Назва, картинка, опис, ціна, знижка, кількість, категорія)

#### **3.1.4. Часові характеристики і розмір пам'яті, необхідної для роботи програми.**

* Час реакції програми на дії користувача (маніпуляції з пристроями введення даних) не повинен перевищувати 0,25 с.
* Час виконання команд меню не більше 1 с.
* Відображення масивів даних за запитами не більше 3 хвилин.
* Доступність БД – 90% цілодобово.
* Операції з’єднання з БД не більше 1 хвилини.
* Обсяг оперативної пам'яті, необхідний для роботи програми не менше 1Гб.
* Дисковій простір, необхідний для збереження програми і файлів даних не більше 300 Мбат для робочої станції та 20 ГБайт.
* Інсталяційний пакет програми, що містить у складі БД не повинні перевищувати 100 Мбайт.

#### **3.2. Вимоги до методів рішення і мов програмування**

Вибір методів рішення здійснюється розробникам без узгодження з замовником.

#### **3.2.1. Вимоги до системи програмних засобів.**

OpenServer, PHP 5, MySQL, HTML 5, CSS 3, JavaScript:

Вимоги до програмного забезпечення робочої станції:

PHP Storm

## **Висновки до першого розділу :**

У ході виконання першого розділу було поставлено завдання КР, а саме який має бути функціонал і визначено актуальність теми. Також визначено мови і фреймворки для розробки та актуальний напрямок розробки власного продукту. Розглянуто існуючі Web-сайти зі схожою тематикою та більш детально проаналізовано.

# **РОЗДІЛ 2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

## **2.1 Проектування загального алгоритму роботи веб-додатку**

На даному етапі розробки розглянемо загальний алгоритм роботи інтернет-магазину.

Загальна схема роботи приведена на рисунку 2.1.

Для початку роботи потрібно створити моделі бази даних та міграції, для полів визначити атрибути та налаштувати показ таблиць в Django admin.

Після цього потрібно реалізувати форми для зручного редагування та додавання даних. Більшість форм наслідуємо від моделей, а деякі створюємо власноруч. Для більшість полів реалізувати валідацію та можливість відправляти помилки користувачу.

Для авторизації та реєстрації використаємо можливості Django. Напишемо views та templates для авторизації. Використаний фреймворк надає можливість безпечно зберігати данні користувача в БД.

Загальна діаграми БД наведена на рисунку 2.2.

Створюємо нові urls та для кожного з них створюємо views. Кожний urls буде мати свій метод, який буде відправляти та приймати форми, для взаємодії з користувачем. В views буде можливість записувати данні до БД, перевіряти що відправити на клієнтську частину, також перевірятиме запити ajax.

При запуску сайту користувач буде бачити основну сторінку магазину, а саме продукти, фільтрацію, пошук та кнопки реєстрації та авторизації. При спробі добавити товар в корзину, користувачу буде запропоновано зареєструватися. Після успішної авторизації він отримає можливість добавляти товари в корзину, оформляти замовлення, та переглядати данні про себе та свої

замовлення.

### Загальна схема роботи програми :

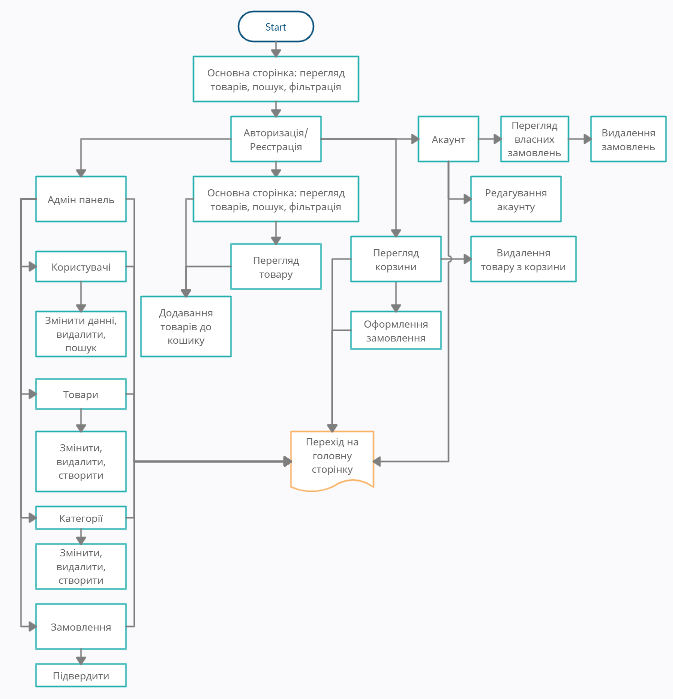


Рисунок 2.1 – Загальна схема роботи програми

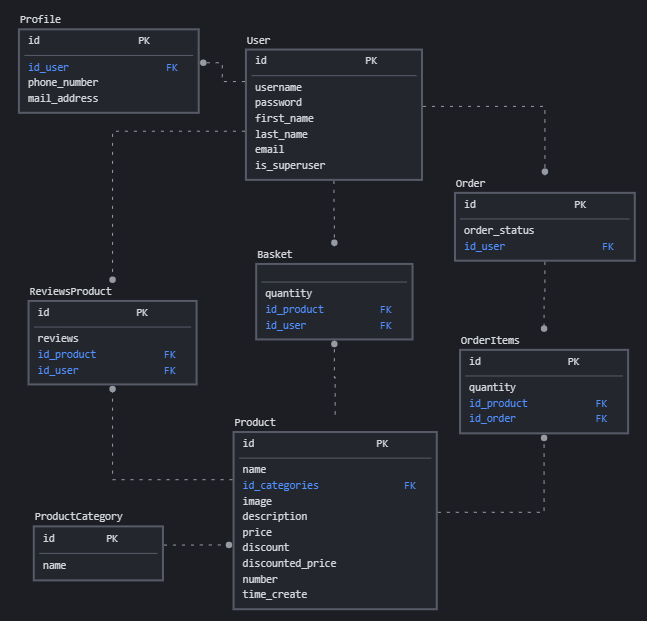


Рисунок 2.2 - Діаграма „сутність-зв‘язок” (логічний рівень) системи

## **2.2 Розробка функціональних алгоритмів роботи програми**

Маючи загальний алгоритм веб-додатку, розглянемо роботу основних алгоритмів.

Основна задача інтернет-магазину, це реалізація продажу електроніки, тому розглянемо цей алгоритм. Для того щоб продати електротовар, насамперед потрібно його добавити на головну сторінку, щоб робити зміни в БД, потрібно авторизоватися з роллю admin. Таким користувачам доступна можливість зайти в адмін-панель.

Для виставлення електротовару на продаж потрібно заповнити відповідну форму, після того продукт розміститься на сайті. При заповненні форми будуть з’являтися підказки про поля які потрібно обов’язково заповнити. Після відправки форми, вона проходить валідації, якщо виникають будь-які помилки, то про це повідомляють користувача.

Покупець може добавити товар до корзини, звідки можна оформити замовлення. Для цього потрібно заповнити форму з номером телефона та адресою поштового відділення.

В роботі цього додатку основним алгоритмом є:

Створення запиту –> отримання даних на бекенді –> обробка отриманих данних та взаємовія з БД –> передача оброблених даних на фронтенд.

## **2.3 Розробка програмного забезпечення**

Структура рішення курсової роботи:

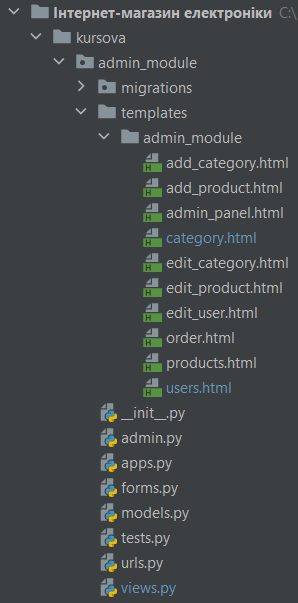


Рисунок 2.2 – Структура КР

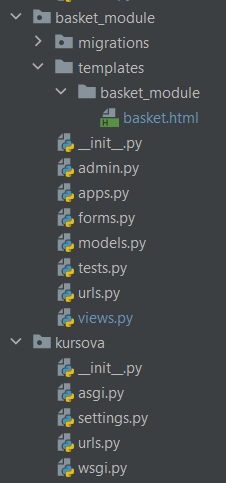


Рисунок 2.3 – Структура КР

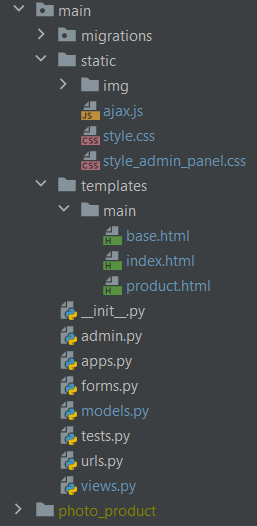


Рисунок 2.4 – Структура КР

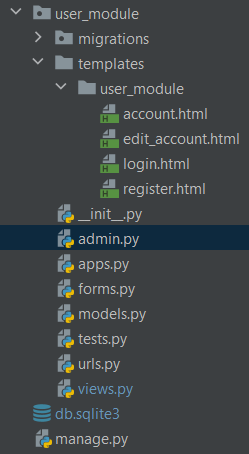


Рисунок 2.5 – Структура КР

Для прикладу детально розглянемо розробку головного модулю main, він працює відповідно до алгоритму, який був наведений вище. В кожному модулі є обов’язково каталог з темплейтами та файли для реалізації патерну MVC(Рис. 2.4).

Після створення модулю потрібно прописати моделі та відтворити міграції. Розглянемо приклад коду моделі для продукту:

class Product(models.Model):  
 name = models.CharField('Назва товару', max\_length=70)  
 image = models.ImageField(upload\_to="photo/%Y/%m/%d/", null=True, blank=True)  
 description = models.TextField('Опис продукту', null=True)  
 price = models.FloatField('Ціна')  
 discount = models.IntegerField('Знижка', max\_length=3)  
 discounted\_price = models.FloatField('Ціна зі знижкою', max\_length=20, default=0)  
 number = models.IntegerField('Кількість', max\_length=10)  
 categories = models.ForeignKey(ProductCategory, null=True, on\_delete=models.SET\_NULL)  
 time\_create = models.DateTimeField(auto\_now\_add=True)  
 time\_update = models.DateTimeField(auto\_now=True)  
  
 def \_\_str\_\_(self):  
 return self.name  
  
 class Meta: # змінюємо назву таблиці для адміна  
 verbose\_name = 'Тавар'  
 verbose\_name\_plural = 'Товари'

При створенні нової моделі, прописуємо всі потрібні поля, для кожної змінної надаємо тип, це може бути як звичайний тест, картинка, число, або дата. Для кожного товару має бути своя категорія, тому для поля categories, реалізовуємо зв'язок один до багатьох до таблиці PorductCategory. Полям дописуємо атрибути, це може бути як назва, максимальна величина, відсутність даних, або запис часу створення.

Після створення всіх потрібних моделей, прописуємо форми в файлі forms, які спрощують валідацію даних та допомагають зв’язати їх з моделями.

Для прикладу розглянемо форму для додавання продукту:

class ProductForm(ModelForm):  
 class Meta:  
 model = Product  
 fields = ['name', 'image', 'description', 'price', 'discount', 'number', 'categories']  
 widgets = {  
 'discount': forms.NumberInput,  
 'number': forms.NumberInput,  
 }

def clean\_discount(self):  
 discount = self.cleaned\_data['discount']

try:  
 discount = int(discount)  
 except ValueError:  
 raise ValidationError("Невірно вказана знижка")  
  
 if discount < 0 or discount > 100:  
 raise ValidationError("Невірно вказана знижка")  
  
 return discount

Як бачимо форма виходить не великою, через те що ми наслідуємся від ModelForm, який надає можливість створити форму з моделі. Для змінити типу поля у widgets записуємо властивості, прописуємо валідації для полів.

У Django за контролер відповідають два файли. Перший urls.py у якому прописуються url, метод або який буде викликаний, та назва для зручного звертання до нього, а другий це views.py, він взаємодіє з БД та приймає запроси у вигляді об’єкту request, генерує деякі результати, які потім відправляються користувачу.

Для прикладу розглянемо алгоритм для відображення товару та додавання відгуку про товар:

Першим кроком йде виклик методу через визначення маршруту и функції path. Url адресою ми передаємо id продукту.

urlpatterns = [  
 path('product/<int:pk>/', views.product, name='product')  
]

def product(request, pk):  
 if request.method == "POST":  
 form = ReviewsForm(request.POST)  
  
 if form.is\_valid():  
 form = form.save(commit=False)  
 form.user = request.user  
 form.product = Product.objects.get(id=pk)  
 form.save()  
  
 return HttpResponseRedirect(reverse('product', args=(pk,)))  
  
 product = Product.objects.get(id=pk)  
 reviews = ReviewsProduct.objects.filter(product=product)  
  
 form = ReviewsForm()

context = {  
 'title': 'Головна сторінка сайту',  
 'product': product,  
 'basket\_count': basket\_count(request),  
 'reviews': reviews,  
 'form': form  
 }  
  
 return render(request, 'main/product.html', context)

При звичайному GET запиту, метод звертається до БД і отримує електротовар з певним id та знаходить всі відгуки про його. Кожен метод містить словник context, у якому міститься наприклад: форми, заголовок сторінки, продукти, замовлення, відгуки.

POST запит передає отримані данні в форму, якщо форма проходить успішно валідації, до неї додаються поля про користувача та продукт, після чого ці данні зберігаються у БД.

Всі інші запити працюють по схожому алгоритму, крім запиту Ajax, розглянемо приклад асинхронного пошуку користувачів:

$("#search\_user").click(function (){  
 $.ajax({  
 type: 'GET',  
 url: 'users',  
 data: {  
 'first\_name': $("#search\_name\_user").val(),  
 },  
 dataType: 'text',  
 cache: false,  
 success: function (data) {  
 let table = ***document***.getElementById('table');  
 if (table){ table.remove() }  
  
 let context = ***JSON***.parse(***JSON***.parse(data));  
   
 showInfoUsers(context);  
 }  
 });  
});

Тут ми можемо побачити, що при натисненні відповідної кнопки створюється Ajax запит. У поле type вказується тип запиту. Для відправки даних на сервер у поле data передається об’єкт. Success має анонімну функцію яка виконається у разі успішного виконання запиту на бекенді. В ції функції ми отримуємо рядок, який парсимо у Json, після цього викликається функція, яка генерує html код і додає його на сторінку користувачу.

def users\_db(request):   
 if is\_ajax(request):  
 if request.GET.get('first\_name'):  
 users = User.objects.filter(first\_name\_\_icontains=request.GET.get('first\_name'))  
 else:  
 users = User.objects.all()  
  
 users = serializers.serialize("json", users)  
 return JsonResponse(users, safe=False)

При відправки запиту Ajax на бекенд, перевіряється чи запит є типу ajax. Якщо запит є асинхронним, то знаходимо користувачів по фільтрам та серелізуємо їх у формат JSON.

## **Висновки до 2 розділу :**

Спроектовано загальну схему програми. Показано структуру КР. Визначено та детально описано основні можливості програми. Пояснено алгоритми найважливіших методів інтернет-магазину та приведено код їх основних частин. Приведено приклади коду деяких методів. Роз’яснено деякі нюанси роботи програми.

# **РОЗДІЛ 3 ОПИС РОБОТИ З ДОДАТКОМ**

## **3.1 Опис роботи з додатком (Опис інтерфейсу)**

Після запуску Web-сайту з’явиться головна сторінка (Рис. 3.1), на якій користувач бачить список товарів та може їх переглядати, сортувати, шукати за назвою.

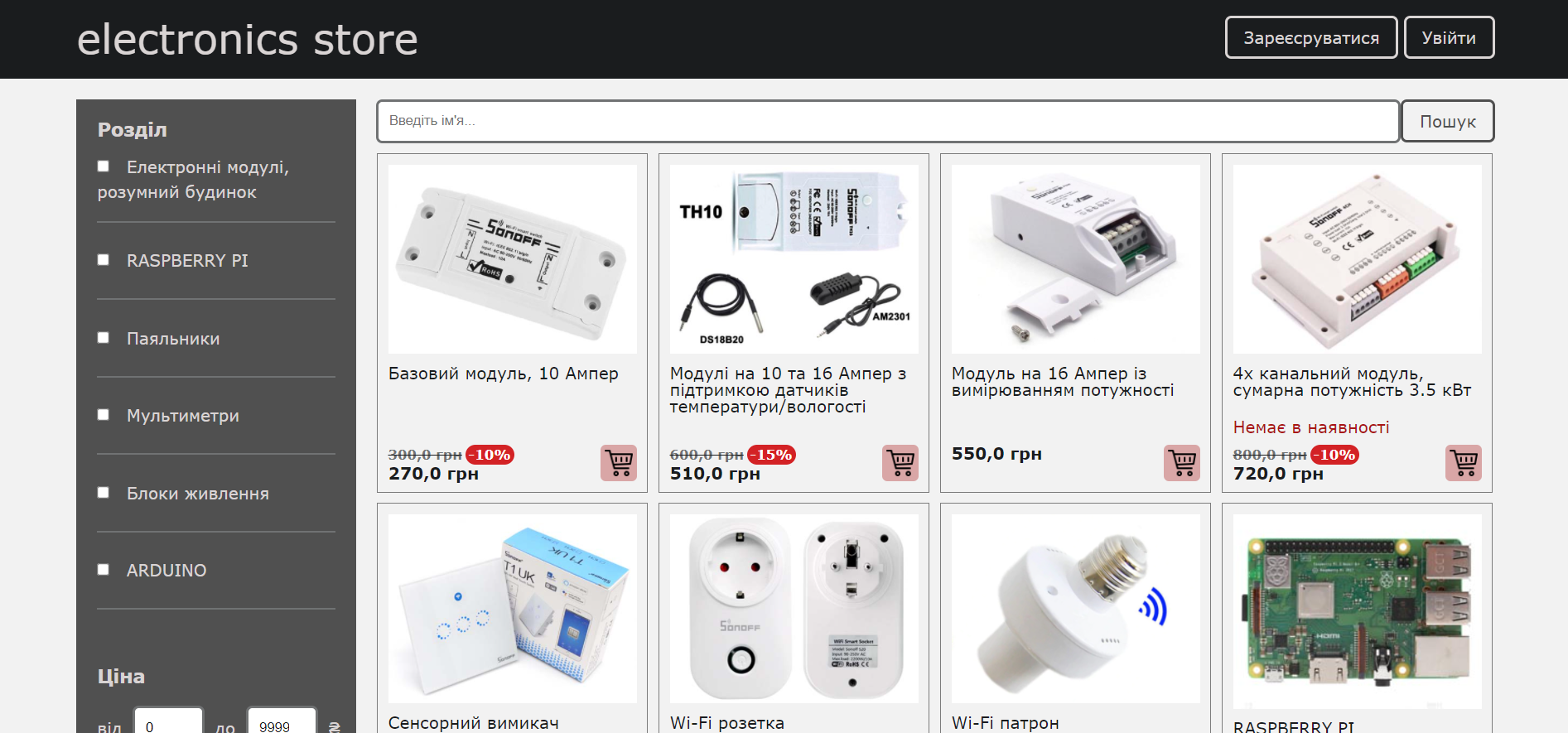


Рисунок 3.1 Головна сторінка

Головна сторінка має можливості переходу до конкретного товару. Також реалізована пагінація у виді кнопки “Показати ще”. У верхньому правому куті знаходяться кнопки меню, а саме реєстрація та авторизація.

Спочатку користувач обирає товар, він може застосувати для цього фільти (Рис. 3.3) та пошук (Рис. 3.3). При кліку на карточку з відповідним товаром, користувач переходить на сторінку з повною інформацією (Рис. 3.4), де він може більш детально прочитати про товар та переглянути відгуки.

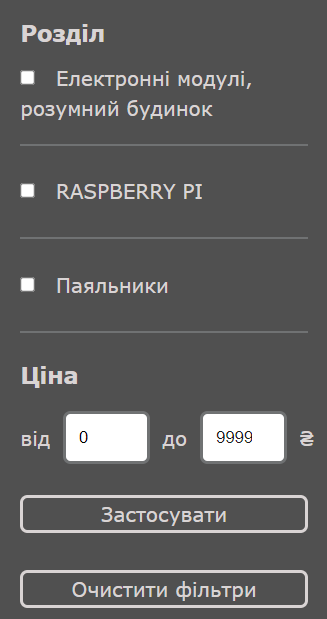


Рисунок 3.2 Фільтри



Рисунок 3.3 Пошук



Рисунок 3.4 Сторінка продукту

При спробі добавити товар до кошику, неавторизованому користувачу сайт запропонує зареєструватися. Після успішної авторизації користувач має можливі додавати товари до кошику (Рис. 3.5). Для прикладу добавимо декілька товарі до корзини. Як можемо бачити, тут відображаються всі наші товари.

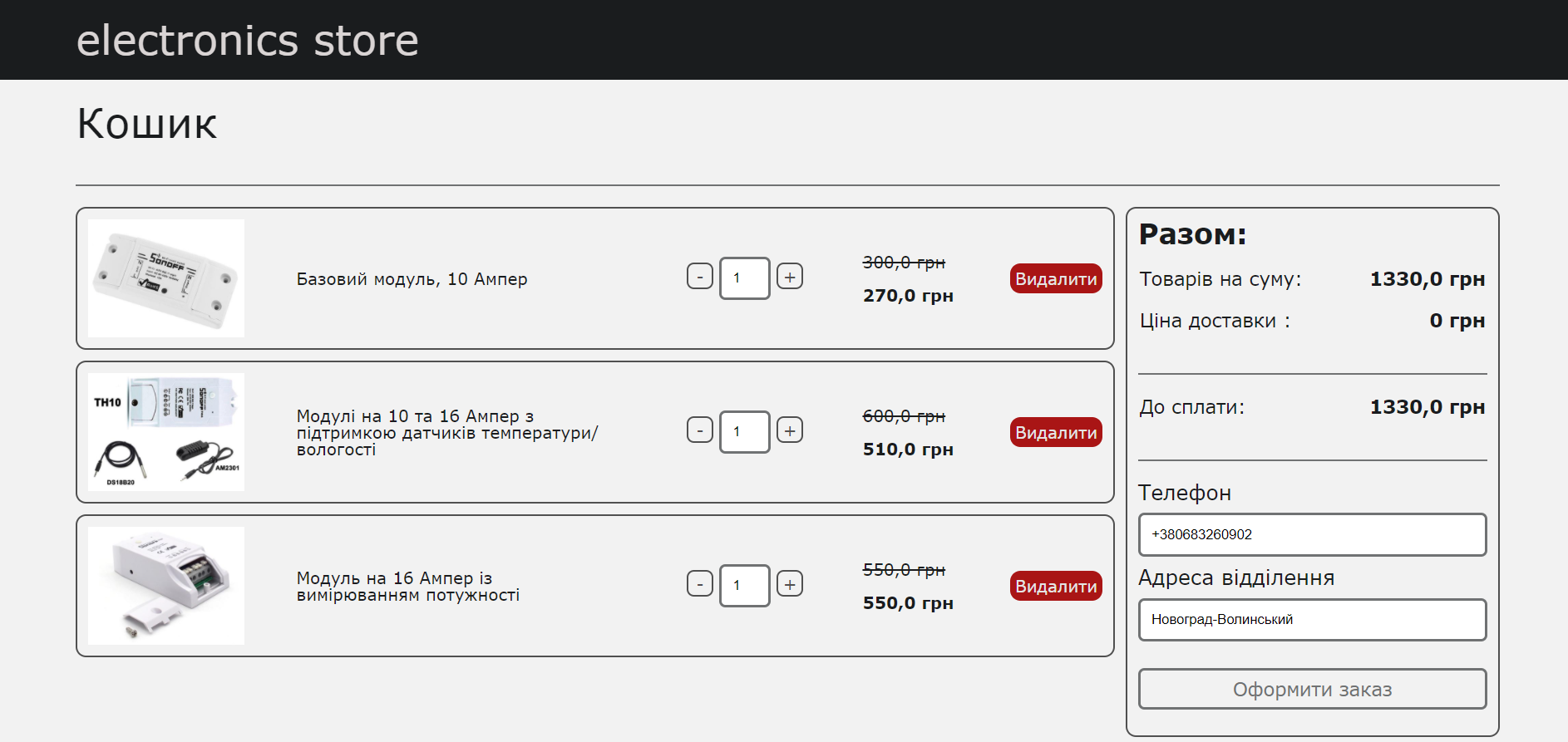
У

Рисунок 3.5 Сторінка кошику

Також за потреби користувач може в корзині змінити кількість товарів, або взагалі видали його. З правого боку екрану відображається блок з загальною сумою, та поля для оформлення замовлення. Після оформлення форми, замовлення додається до списку замовлень (Рис. 3.6).



Рисунок 3.6 Сторінка замовлень

На сторінці акаунт (Рис 3.7) можна переглядати персональні дані про себе, за потреби змінити їх (Рис 3.8). Ще на цій сторінці є можливість переглядати всі замовлення, у користувача є можливість відмінити замовлення, переглянути статус замовлень та їх історію.

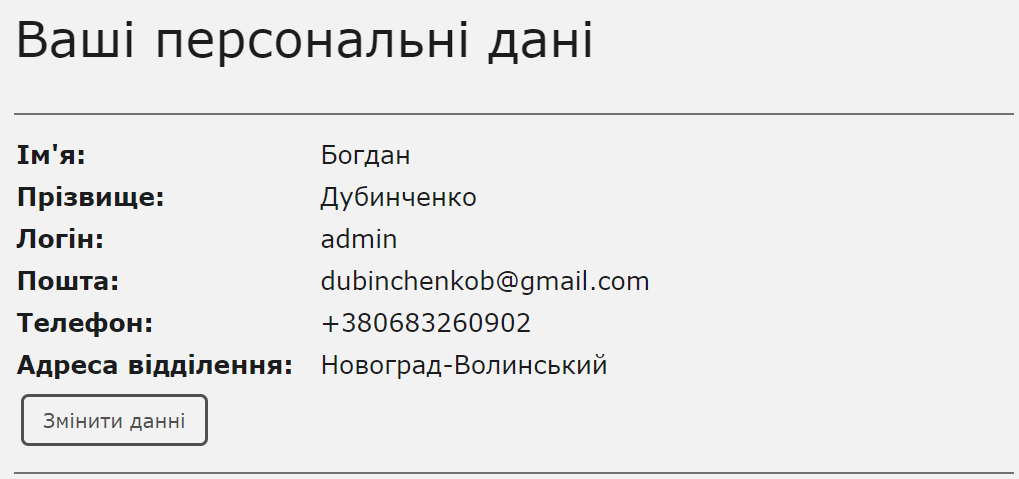


Рисунок 3.7 Сторінка персональних даних

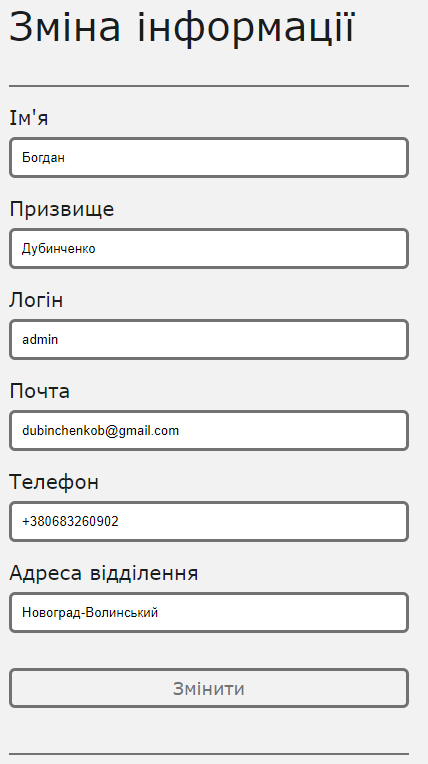


Рисунок 3.8 Сторінка зміни інформації

Далі перейдемо до можливостей користувачів з адмін ролями. У верхньому правому куті розміщена кнопка “Адмін панель”, яка відкриває сторінку з меню (Рис. 3.9). На даній сторінці адміністратор може переглядати та змінювати дані про користувачів (Рис. 3.10). Також має можливість створювати, редагувати, видаляти товари (Рис. 3.11) та категорії. Адмін має можливість надавати права адміна іншому користувачу.

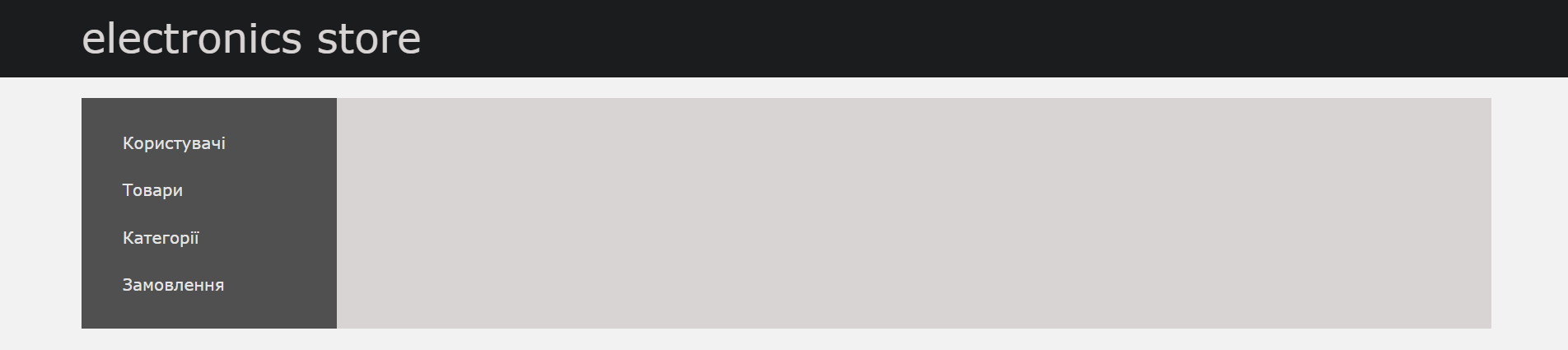


Рисунок 3.9 Сторінка адміна

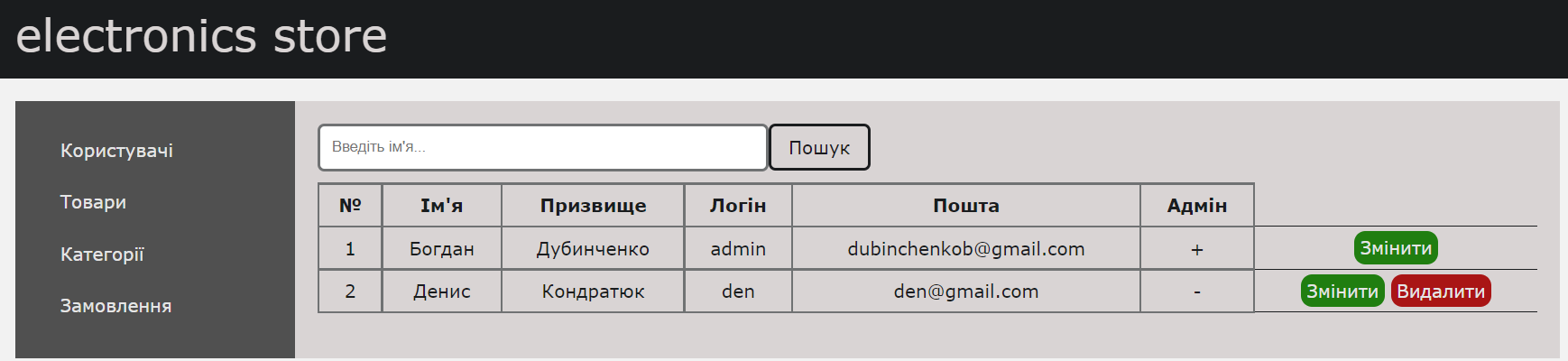


Рисунок 3.10 Сторінка користувачів

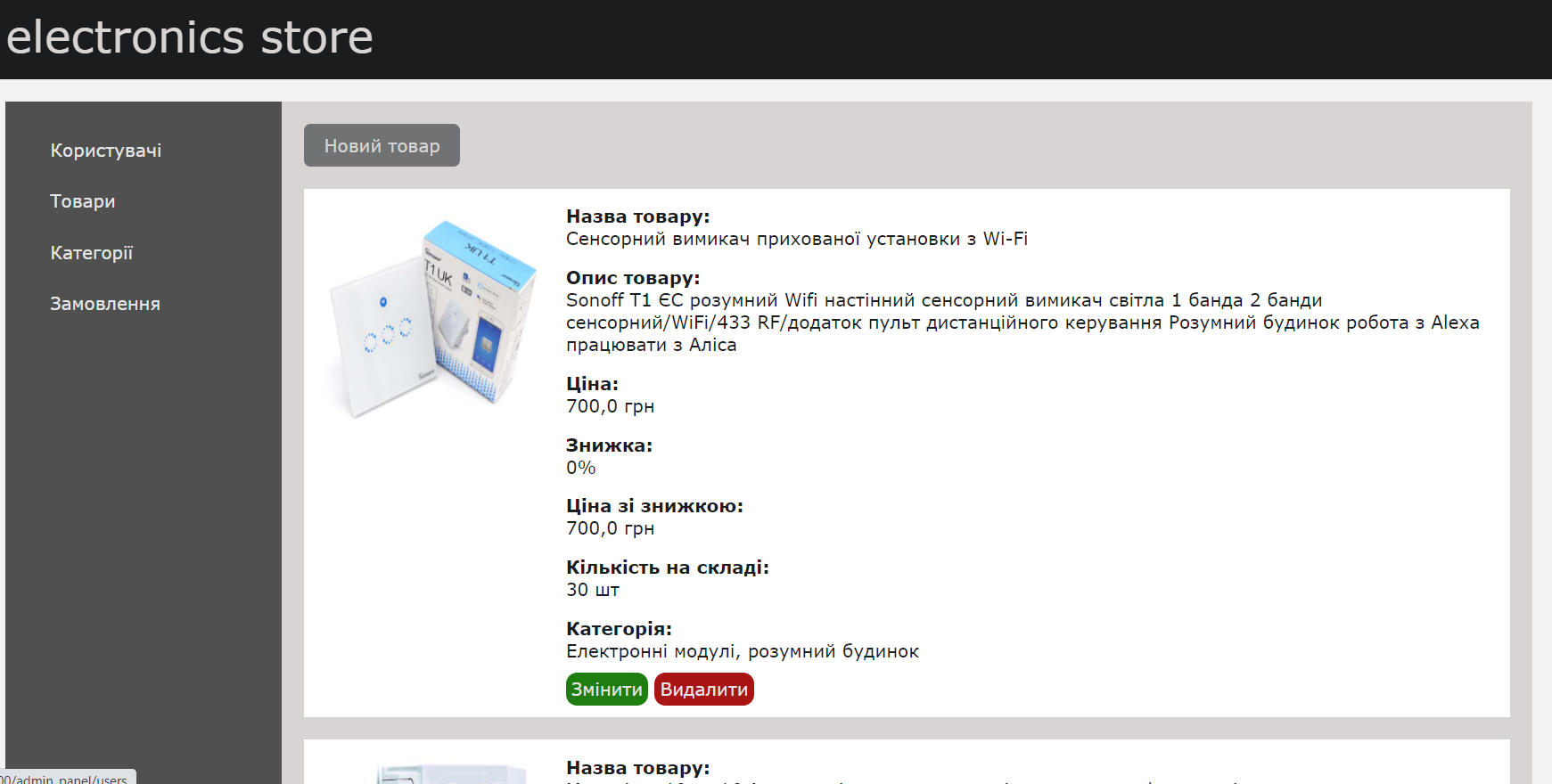


Рисунок 3.11 Сторінка товарів

Наприклад при створені нового товару (Рис. 3.12), користувач буде отримувати повідомлення про правильність введених даних та про успішність додавання товару.

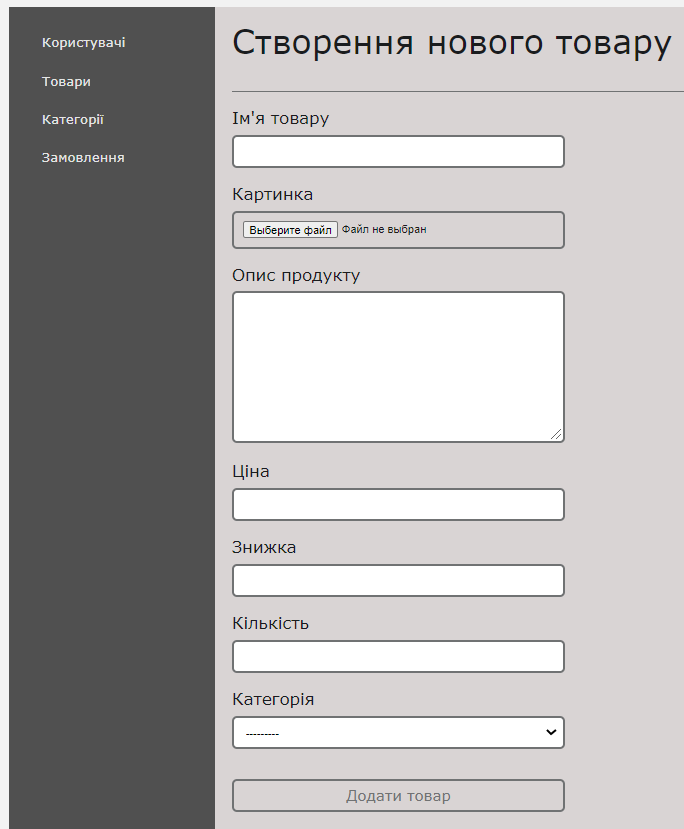


Рисунок 3.12 Створення нового товару

## **3.2 Тестування веб-додатку**

Веб-сайт відображає попередження про помилку при введені неправильних даних. Наприклад при спробі створення товару з від’ємною ціною, або знижкою, відображаються помилки про правильність заповнення полів (Рис. 3.13).

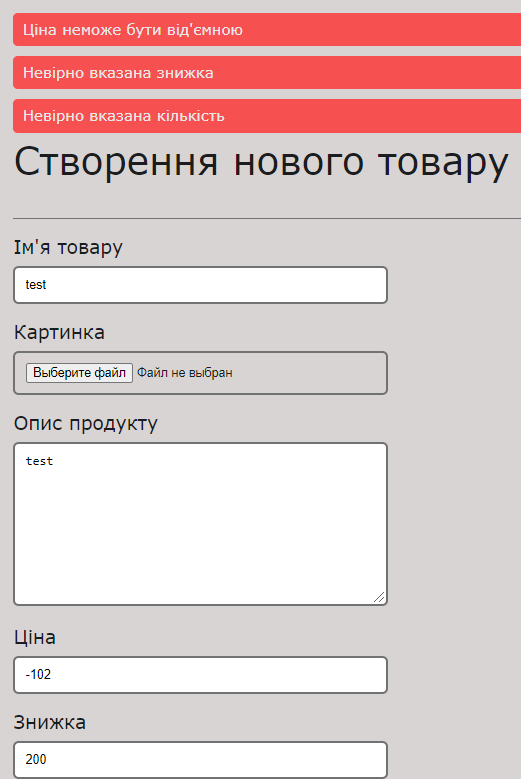


Рисунок 3.13 Повідомлення про неправильність даних

При вводі некоректної електронної пошти або паролю, з’являється помилка (Рис 3.14)

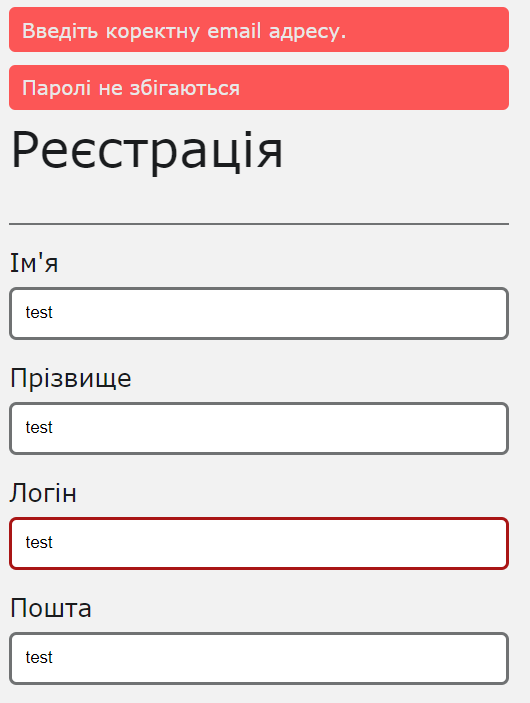


Рисунок 3.14 Повідомлення про неправильність паролю або пошти

При вводі даних які вже існують та не можуть повторюватися, з’являється помилка про існування таких даних (Рис. 3.15)

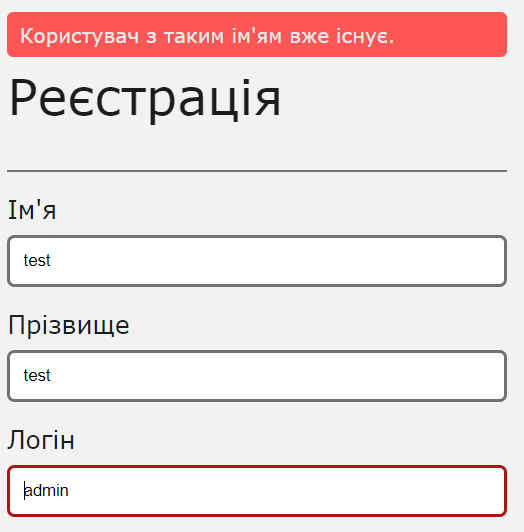


Рисунок 3.14 Помилка про дублювання

Подібна валідація реалізована на всіх формах.

## **Висновки до 3 розділу :**

Було детально описано роботу Web-сайту, користувацький інтерфейс, очікувані дії користувача та їх результати. Також було розглянуто основні можливості інтернет-магазину. Наведено приклади обробки помилок.

# **ВИСНОВКИ**

Під час написання даної курсової роботи було отримано навички роботи з розробки веб-додатку.

В першому розділі курсової роботи, було проаналізовано знайдені інтернет-магазини зі схожим функціоналом та інтерфейсом, та було сформовано уявлення про інтернет-магазин, що розроблюється. Було виявлено основні потреби для написання даного веб-додатку.

В другому розділі курсової роботи, було проведено розробку функціональних і загальних частин веб-додатку. При розробці функціонального алгоритму, розробили уявлення про методи розробки програмного коду. На заключному етапі, розроблено його. В результаті, отримано готовий програмний продукт, який виконує всі заплановані функції і задачі.

В третьому розділі курсової роботи було описано методи користування створеним інтернет-магазином, були пояснені і продемонстровані всі створені елементи інтерфейсу. Також було проведено тестування. В ході тестування проблем не виникало.

В результаті виконання курсової роботи отримано інтернет-магазин електроніки, який повністю відповідає запланованому функціоналу, умовам та вимогам, поставленим на початку проектування. Інтернет-магазин є зрозумілим та зручним в користуванні.

# **СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. Руководство по веб-фреймфорку Django. [Електронний ресурс] – 2018. Режим доступу: <https://metanit.com/python/django/>
2. Документація Django. [Електронний ресурс] – 2016. Режим доступу: <https://docs.djangoproject.com/en/4.0/>
3. Документація jQuery. [Електронний ресурс] – 2016. Режим доступу: <https://api.jquery.com/category/ajax/>
4. Дмитрий Кирсанов. “Веб-дизайн” [Електронний ресурс] - 2006. Режим доступу: <http://web-diz.com.ua/skachat/veb-dizayn--dmitriy-kirsanov-skachatbesplatno/>
5. Тим Кедлек. “Адаптивный дизайн. Делаем сайты для любых устройств” [Електронний ресурс] - 2013. Режим доступу: <https://www.ozon.ru/context/detail/id/21098804/>
6. Плюси і мінуси об’єктно-орієнтованого програмування [Електронний ресурс] - 2015. Режим доступу: <http://damp.biz/plyusi-i-minusi-obyektno-oriyentovanogoprogramuvannya/>

# **ДОДАТКИ**

# Додаток А

Схема системи інтернет-магазину електроніки:

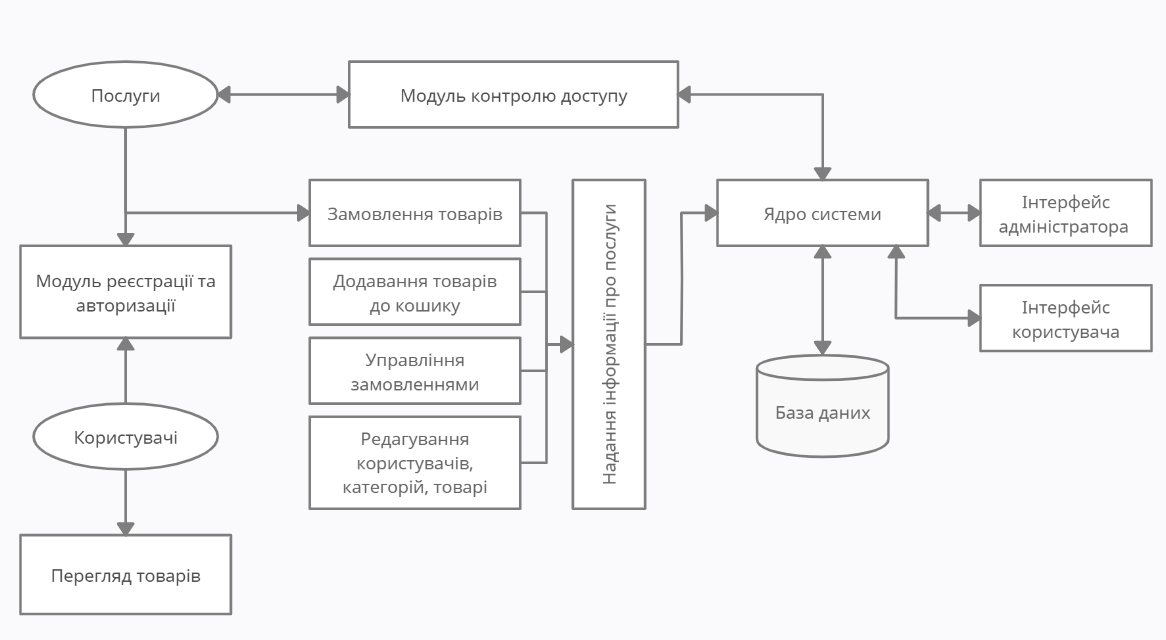


Рис А. Структура системи інтернет-магазину електроніки

# Додаток Б

Програмний код

Model.py:

from django.core.validators import RegexValidator  
from django.db import models  
from django.contrib.auth.models import User  
  
  
class Profile(models.Model):  
 user = models.OneToOneField(User, on\_delete=models.CASCADE)  
 phone\_regex = RegexValidator(regex=r'^(\+?(38))?[-\(]?\d{3}\)?-?\d{3}-?\d{2}-?\d{2}$', message="Номер телефону повинен бути в форматі '+380XXXXXXXXX'")  
 phone\_number = models.CharField(validators=[phone\_regex], max\_length=13, null=True, blank=True)  
 mail\_address = models.CharField(max\_length=100, null=True, blank=True)  
  
 def \_\_str\_\_(self):  
 return self.user.username  
  
 class Meta:  
 verbose\_name = 'Профіль'  
 verbose\_name\_plural = 'Профіль'

class ProductCategory(models.Model):  
 name = models.CharField('Назва категорії', max\_length=100)  
  
 def \_\_str\_\_(self):  
 return self.name  
  
 class Meta:  
 verbose\_name = 'Категорію'  
 verbose\_name\_plural = 'Категорії'  
  
  
class Product(models.Model):  
 name = models.CharField('Назва товару', max\_length=70)  
 image = models.ImageField(upload\_to="photo/%Y/%m/%d/", null=True, blank=True)  
 description = models.TextField('Опис продукту', null=True)  
 price = models.FloatField('Ціна')  
 discount = models.IntegerField('Знижка', max\_length=3)  
 discounted\_price = models.FloatField('Ціна зі знижкою', max\_length=20, default=0)  
 number = models.IntegerField('Кількість', max\_length=10)  
 categories = models.ForeignKey(ProductCategory, null=True, on\_delete=models.SET\_NULL)  
 time\_create = models.DateTimeField(auto\_now\_add=True)  
 time\_update = models.DateTimeField(auto\_now=True)  
  
 def \_\_str\_\_(self):  
 return self.name  
  
 class Meta: # змінюємо назву таблиці для адміна  
 verbose\_name = 'Тавар'  
 verbose\_name\_plural = 'Товари'  
  
  
class Basket(models.Model):  
 user = models.ForeignKey(User, on\_delete=models.CASCADE)  
 product = models.ForeignKey(Product, on\_delete=models.CASCADE)  
 quantity = models.PositiveIntegerField(default=0, max\_length=5)  
  
 def \_\_str\_\_(self):  
 return self.user.username  
  
 class Meta:  
 verbose\_name = 'Корзину'  
 verbose\_name\_plural = 'Корзина'  
  
  
class ReviewsProduct(models.Model):  
 user = models.ForeignKey(User, on\_delete=models.CASCADE)  
 product = models.ForeignKey(Product, on\_delete=models.CASCADE)  
 reviews = models.TextField('Відгук', max\_length=200)  
  
 def \_\_str\_\_(self):  
 return self.product  
  
 class Meta:  
 verbose\_name = 'Відгук'  
 verbose\_name\_plural = 'Відгуки'

class Order(models.Model):  
 user = models.ForeignKey(User, on\_delete=models.CASCADE)  
 order\_status = models.BooleanField()  
  
 def \_\_str\_\_(self):  
 return str(self.id)  
  
 class Meta:  
 verbose\_name = 'Замовлення'  
 verbose\_name\_plural = 'Замовлення'  
  
  
class OrderItems(models.Model):  
 order = models.ForeignKey(Order, on\_delete=models.CASCADE, related\_name='orderItems')  
 product = models.ForeignKey(Product, on\_delete=models.CASCADE)  
 quantity = models.PositiveIntegerField(default=0, max\_length=5)  
  
 def \_\_str\_\_(self):  
 return self.product.name  
  
 class Meta:  
 verbose\_name = 'Замовленні товари'  
 verbose\_name\_plural = 'Замовленні товари'

urls.py:

from django.urls import path  
from . import views  
  
urlpatterns = [  
 path('', views.index, name='home'),  
 path('product/<int:pk>/', views.product, name='product'),  
 path('product/<int:pk>/reviews/<int:pk\_review>', views.delete\_reviews),  
]

views.py:

from django.shortcuts import render, redirect  
from django.core import serializers  
from django.urls import reverse  
  
from .forms import ReviewsForm  
from .models import Product, ProductCategory, Basket, ReviewsProduct  
from django.http import JsonResponse, HttpResponseRedirect, HttpResponseNotFound  
  
  
def is\_ajax(request):  
 return request.META.get('HTTP\_X\_REQUESTED\_WITH') == 'XMLHttpRequest'  
  
  
def basket\_count(request):  
 basket\_count = 0  
 if request.user.is\_authenticated:  
 basket\_items = Basket.objects.filter(user=request.user)  
 for i in range(len(basket\_items)):  
 basket\_count += basket\_items[i].quantity

return basket\_count  
  
  
def index(request):  
 if is\_ajax(request):  
 try:  
 price\_\_from = int(request.GET.get("price\_\_from"))  
 price\_\_to = int(request.GET.get("price\_\_to"))  
 except ValueError:  
 price\_\_from = 0  
 price\_\_to = 99999  
  
 products = Product.objects.all()  
  
  
 if request.GET.getlist("category[]"):  
 products = products.filter(categories\_\_name\_\_in=request.GET.getlist("category[]"))  
  
 if request.GET.get("search"):  
 products = products.filter(name\_\_icontains=request.GET.get("search"))  
  
 if request.GET.get("price\_\_from") and request.GET.get("price\_\_to"):  
 products = products.filter(discounted\_price\_\_range=(price\_\_from, price\_\_to))  
  
 try:  
 products = serializers.serialize("json", products[:int(request.GET.get('count\_show')) + 1])  
 except ValueError:  
 return redirect('login')  
  
 return JsonResponse(products, safe=False)  
 else:  
 products = Product.objects.all()[:10]  
  
 categories = ProductCategory.objects.all()  
  
 context = {  
 'title': 'Головна сторінка сайту',  
 'products': products,  
 'categories': categories,  
 'basket\_count': basket\_count(request),  
 }  
  
 return render(request, 'main/index.html', context)  
  
  
def product(request, pk):  
 if request.method == "POST":  
 form = ReviewsForm(request.POST)  
  
 if form.is\_valid():  
 form = form.save(commit=False)  
 form.user = request.user  
 form.product = Product.objects.get(id=pk)  
 form.save()

return HttpResponseRedirect(reverse('product', args=(pk,)))  
  
 product = Product.objects.get(id=pk)  
 reviews = ReviewsProduct.objects.filter(product=product)  
  
 form = ReviewsForm()  
  
 context = {  
 'title': 'Головна сторінка сайту',  
 'product': product,  
 'basket\_count': basket\_count(request),  
 'reviews': reviews,  
 'form': form  
 }  
  
 return render(request, 'main/product.html', context)  
  
  
def delete\_reviews(request, pk, pk\_review):  
 if not request.user.is\_authenticated:  
 return redirect('home')  
  
 try:  
 reviews = ReviewsProduct.objects.get(id=pk\_review)  
 if request.user == reviews.user:  
 reviews.delete()  
 else:  
 return redirect('home')  
  
 return HttpResponseRedirect(reverse('product', args=(pk,)))  
  
 except ProductCategory.DoesNotExist:  
 return HttpResponseNotFound("<h2>Такого коментарю не існує</h2>")

forms.py:

from django.forms import ModelForm, TextInput  
from django import forms  
from .models import ReviewsProduct  
  
  
class ReviewsForm(ModelForm):  
 class Meta:  
 model = ReviewsProduct  
 fields = ['reviews']