МИНИСТЕРСТВО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ

ГБПОУ МО "Ногинский колледж"

Курсовой проект  
по МДК.09.01 Проектирование и разработка веб-приложений  
ПМ.09 Проектирование, разработка и оптимизация веб-приложений

Тема: Интернет-магазин

Разработчик: студент группы 3ИСПР  
Безуглов Даниил  
  
Руководитель проекта: преподаватель Степан Олегович  
  
Ногинск, 2024 г.

# СОДЕРЖАНИЕ

1. 1. Введение .......................... 3
2. 2. 1. Описание проекта .......................... 4
3. 3. 2. Архитектура и структура проекта .......................... 5
4. 4. 3. Используемые технологии .......................... 6
5. 5. 4. Основные функции .......................... 7
6. 6. 5. Установка и запуск .......................... 8
7. 7. 6. Заключение .......................... 9
8. 8. 7. Список использованных источников .......................... 10
9. 9. 8. Теоретическая часть .......................... 11
10. 10. 9. Тестирование функциональности интернет-магазина .......................... 12
11. 11. 10. Примеры кода .......................... 13
12. 12. 11. Структура базы данных .......................... 14
13. 13. Приложения .......................... 15

# Введение

Курсовой проект посвящён разработке серверной части веб-приложения "Интернет-магазин". Целью работы является закрепление знаний и развитие практических навыков по созданию современных веб-приложений с использованием фреймворка Django.

В ходе выполнения проекта были реализованы основные функции интернет-магазина: каталог товаров, корзина, система заказов, личный кабинет пользователя и административная панель.

Актуальность выбранной темы обусловлена растущим спросом на онлайн-торговлю и необходимостью создания эффективных и удобных платформ для продажи товаров. Современное состояние рынка показывает, что интернет-магазины становятся неотъемлемой частью бизнеса, что делает разработку таких систем особенно важной.

Введение также включает краткую характеристику области применения, обоснование актуальности выбранной темы и описание современного состояния систем на рынке.

# 1. Описание проекта

Проект представляет собой интернет-магазин, позволяющий пользователям просматривать каталог товаров, добавлять их в корзину, оформлять заказы и управлять своими заказами через личный кабинет. Для администраторов реализована отдельная панель управления товарами, категориями и заказами.

В рамках проекта были изучены различные аспекты разработки веб-приложений, включая работу с базами данных, создание пользовательского интерфейса и обеспечение безопасности.

# 2. Архитектура и структура проекта

Проект построен на фреймворке Django и состоит из двух основных модулей: основного модуля shop (настройки, маршруты, конфигурация) и приложения store (модели, представления, формы, шаблоны). Для хранения данных используется база данных SQLite3. Для загрузки тестовых данных предусмотрен отдельный скрипт.

Структура проекта:

internet-shop/  
├── manage.py # Скрипт управления Django  
├── requirements.txt # Зависимости проекта  
├── create\_test\_data.py # Скрипт для создания тестовых данных  
├── shop/ # Основной модуль проекта  
│ ├── settings.py # Настройки проекта  
│ ├── urls.py # Основные URL-маршруты  
│ ├── wsgi.py # WSGI-конфигурация  
│ └── asgi.py # ASGI-конфигурация  
└── store/ # Приложение магазина  
 ├── models.py # Модели данных  
 ├── views.py # Представления  
 ├── urls.py # URL-маршруты приложения  
 ├── forms.py # Формы  
 ├── cart.py # Класс корзины  
 └── templates/ # Шаблоны  
 └── store/  
 ├── base.html  
 ├── product/  
 ├── cart/  
 └── order/

# 3. Используемые технологии

- Python 3.13  
- Django 5.0.2  
- SQLite3  
- Bootstrap 5  
- Crispy Forms  
- Pillow (для работы с изображениями)

Выбор технологий был обусловлен их популярностью, надежностью и широкой поддержкой сообщества. Django предоставляет мощные инструменты для разработки веб-приложений, а Bootstrap обеспечивает современный и отзывчивый дизайн.

# 4. Основные функции

Каталог товаров: отображение, поиск, фильтрация по цене, детальная информация.

Корзина покупок: добавление, изменение количества, удаление товаров, подсчет стоимости.

Система заказов: оформление, история заказов, автоматическое списание товаров со склада.

Личный кабинет: просмотр истории заказов, редактирование профиля.

Административная панель: управление товарами, категориями, заказами и пользователями.

# 5. Установка и запуск

1. Клонировать репозиторий и перейти в папку проекта.

2. Создать виртуальное окружение и установить зависимости из requirements.txt.

3. Применить миграции и создать суперпользователя.

4. Загрузить тестовые данные с помощью create\_test\_data.py.

5. Запустить сервер разработки командой python manage.py runserver.

6. Открыть http://localhost:8000 в браузере.

# 6. Заключение

В ходе выполнения курсового проекта были закреплены знания по разработке серверных веб-приложений на Django. Реализованы все основные функции интернет-магазина, что позволяет использовать данный проект как основу для дальнейшего развития и внедрения новых возможностей.

Заключение также подводит итоги работы, обобщает ход решения и делает вывод о достижении поставленных целей.

# 7. Список использованных источников

[1] Официальная документация Django: https://docs.djangoproject.com/ru/5.0/

[2] Bootstrap: https://getbootstrap.com/

[3] Документация Python: https://docs.python.org/3/

[4] Примеры кода и статьи с сайта https://habr.com/

# 8. Теоретическая часть

## 8.1. История и развитие интернет-магазинов

Первые интернет-магазины появились в середине 1990-х годов, с развитием глобальной сети Интернет. Одним из первых и наиболее успешных проектов стал Amazon, основанный Джеффом Безосом в 1994 году. Изначально он предлагал только книги, но позже ассортимент расширился до миллионов товаров, включая электронику, одежду, товары для дома и многое другое.

Параллельно с Amazon активно развивался eBay, ориентированный на продажи между пользователями и аукционную модель. Эти компании стали флагманами нового направления — электронной коммерции.

С начала 2000-х годов интернет-торговля начала стремительно развиваться. Появление более надёжных платёжных систем (например, PayPal), широкое распространение Интернета и развитие логистических сетей сыграли ключевую роль в этом процессе. Постепенно электронная коммерция проникла в жизнь каждого человека, изменив не только модель покупок, но и общее представление о торговле.

Сегодня интернет-магазины представляют собой высокотехнологичные платформы, способные обрабатывать миллионы заказов ежедневно, обеспечивать персонализированные рекомендации, анализировать поведение пользователей и интегрироваться с социальными сетями, рекламными системами и маркетинговыми платформами.

## 8.2. Сравнение онлайн и оффлайн торговли

|  |  |  |
| --- | --- | --- |
| Критерий | Онлайн-магазин | Оффлайн-магазин |
| Доступность | 24/7, из любой точки мира | Зависит от времени работы и геолокации |
| Ассортимент | Огромный, масштабируется за счёт складов | Ограничен площадью торгового зала |
| Издержки | Ниже: нет аренды помещений, меньше персонала | Высокие: аренда, коммунальные, зарплаты |
| Удобство | Быстрый поиск, фильтры, доставка на дом | Визуальный осмотр товара, живое общение |
| Время покупки | Несколько кликов | Необходимо физически посетить магазин |
| Обратная связь | Отзывы, рейтинги, чат-боты | Только личное обращение к продавцу |

## 8.3. Архитектура клиент-серверных приложений

Современные интернет-магазины работают по архитектуре «клиент-сервер». Клиентская часть — это интерфейс, с которым взаимодействует пользователь (например, веб-страница в браузере), а серверная часть — это логика, обрабатывающая запросы и взаимодействующая с базой данных.

Основные компоненты:

- Клиент — браузер пользователя, отображает интерфейс.

- Сервер — приложение на Django, обрабатывает запросы, управляет логикой.

- База данных — хранит информацию о пользователях, товарах, заказах.

Типичный процесс:

1. Пользователь отправляет запрос (например, открыть страницу товара).

2. Сервер обрабатывает запрос, получает данные из базы.

3. Возвращает HTML-страницу или JSON-ответ.

4. Браузер отображает результат пользователю.

Такой подход обеспечивает масштабируемость и позволяет отделить логику от интерфейса, а также реализовывать разные клиенты: веб, мобильные приложения и т.д.

## 8.4. Безопасность и защита данных в интернет-магазинах

При разработке интернет-магазинов безопасность — критически важный аспект. Особенно это касается личных данных пользователей, паролей, истории заказов и, при необходимости, платёжной информации.

Основные меры защиты:

- Хэширование паролей (в Django используется PBKDF2).

- CSRF-защита — предотвращение подделки межсайтовых запросов.

- Валидация данных — фильтрация и проверка пользовательского ввода.

- Ограничение прав доступа — разграничение прав между пользователями и администраторами.

- Регулярные обновления — использование актуальных версий фреймворков и библиотек.

Также важно настроить безопасное хранение сессий, работу по HTTPS и использовать современные методы аутентификации.

## 8.5. Обзор платформ и фреймворков

|  |  |  |  |
| --- | --- | --- | --- |
| Платформа | Язык | Преимущества | Недостатки |
| Django | Python | Безопасность, структура, масштабируемость | Требует навыков программирования |
| Flask | Python | Простота, гибкость | Нет встроенной ORM, требует ручной сборки |
| Laravel | PHP | Мощный функционал из коробки | Сложность при масштабировании |
| Shopify | SaaS-платформа | Быстрый запуск, нет необходимости в коде | Ограниченная кастомизация, платный |
| WordPress + WooCommerce | PHP | Простота, расширения | Неудобна для крупных магазинов |

Django — оптимальный выбор для кастомных решений, где важна гибкость, безопасность и расширяемость.

## 8.6. Юзабилити и проектирование интерфейсов

Удобный интерфейс — ключевой фактор успеха интернет-магазина. Пользователь должен легко находить нужные товары, оформлять заказ и управлять своим профилем.

Принципы хорошего юзабилити:

- Простая навигация: меню, хлебные крошки, фильтры.

- Поиск и сортировка: по цене, названию, категории.

- Адаптивность: интерфейс одинаково удобен на ПК и телефоне.

- Обратная связь: уведомления, всплывающие подсказки.

- Минимализм: ничего лишнего, только важные действия.

Bootstrap 5, использованный в проекте, помогает быстро создавать адаптивный и современный интерфейс, следуя этим принципам.

# 9. Тестирование функциональности интернет-магазина

## 9.1. Виды тестирования

Тестирование проекта проводилось вручную, а также с использованием встроенных возможностей Django.

Основные виды тестирования:

- Функциональное — проверка корректности работы корзины, заказов, каталога.

- Интерфейсное — проверка отображения страниц на разных устройствах.

- Регрессионное — проверка стабильности после внесения изменений.

- Тестирование безопасности — базовая проверка CSRF, защиты от SQL-инъекций и XSS.

## 9.2. Примеры проверок

|  |  |  |
| --- | --- | --- |
| Проверяемый элемент | Ожидаемый результат | Статус |
| Добавление товара в корзину | Товар появляется в корзине, цена пересчитывается | ✅ Пройдено |
| Удаление товара | Товар удаляется, корзина обновляется | ✅ Пройдено |
| Оформление заказа | Создаётся запись в базе, отображается в «Мои заказы» | ✅ Пройдено |
| Авторизация | Переход к личному кабинету возможен только после входа | ✅ Пройдено |
| Админ-панель | Доступна только суперпользователю | ✅ Пройдено |

# 10. Примеры кода

## 10.1. Модель товара

```python  
class Product(models.Model):  
 name = models.CharField(max\_length=255)  
 description = models.TextField()  
 price = models.DecimalField(max\_digits=10, decimal\_places=2)  
 image = models.ImageField(upload\_to='products/')  
 category = models.ForeignKey(Category, on\_delete=models.CASCADE)  
  
 def \_\_str\_\_(self):  
 return self.name  
```

Пояснение: Модель Product описывает структуру товара: название, описание, цена, изображение и категория. Django автоматически создаёт таблицу в БД и обеспечивает взаимодействие с ней.

## 10.2. Представление каталога

```python  
def product\_list(request):  
 products = Product.objects.all()  
 return render(request, 'store/product/list.html', {'products': products})  
```

Пояснение: Функция product\_list получает список всех товаров из базы данных и передаёт их в шаблон для отображения. Это простейшее представление (view) в Django.

# 11. Структура базы данных

ERD (Entity-Relationship Diagram) - это визуальное представление структуры базы данных, которое показывает взаимосвязи между различными сущностями (таблицами) в системе. В нашем проекте используется реляционная база данных SQLite3, которая состоит из следующих основных таблиц:

|  |  |
| --- | --- |
| Таблица | Поля |
| User | id, username, password, email, first\_name, last\_name |
| Category | id, name |
| Product | id, name, description, price, image, category\_id (внешний ключ) |
| Order | id, user\_id (внешний ключ), created\_at, status |
| OrderItem | id, order\_id (внешний ключ), product\_id (внешний ключ), quantity, price |

🔐 В таблице User используются встроенные возможности Django (AbstractUser).

Связи между таблицами:

• Product связан с Category через внешний ключ category\_id (отношение "многие к одному")

• Order связан с User через внешний ключ user\_id (отношение "многие к одному")

• OrderItem связан с Order и Product через внешние ключи (отношение "многие к одному")

ERD-диаграмма:

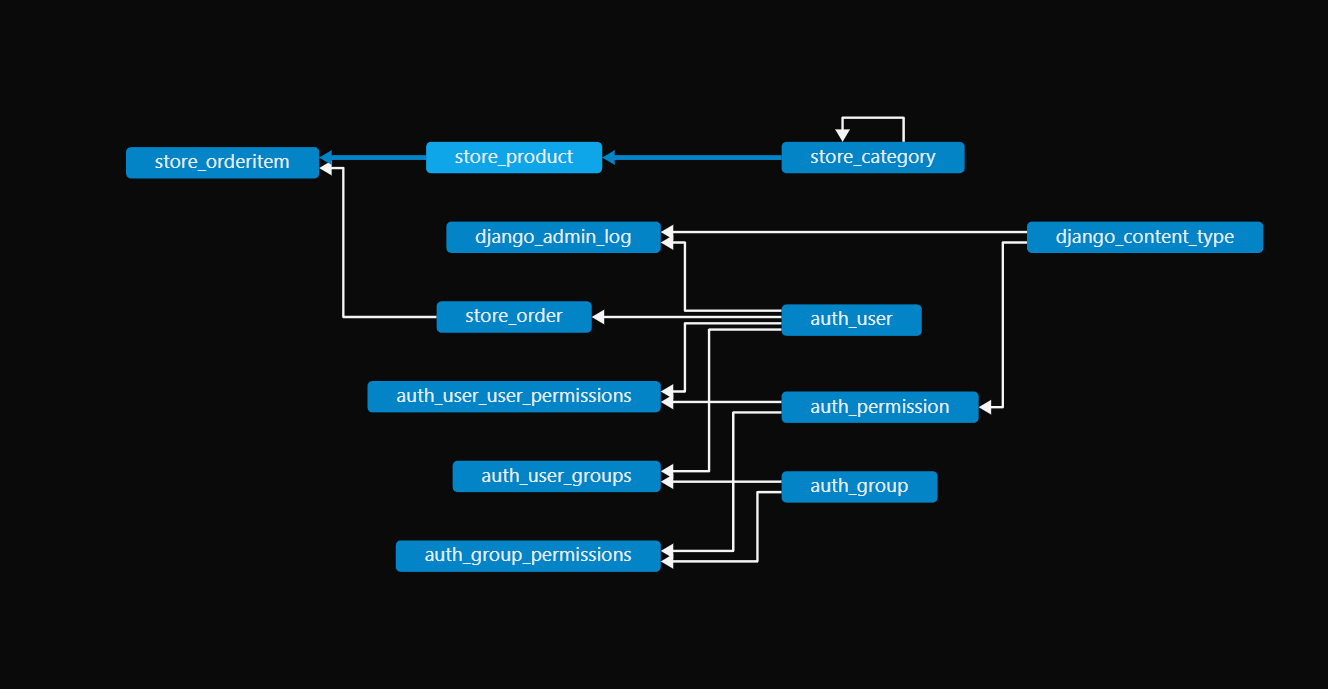
В приложении представлена ERD-диаграмма (рисунок 1), которая наглядно показывает:

• Все таблицы базы данных и их поля

• Связи между таблицами (один-ко-многим, многие-ко-многим)

• Первичные и внешние ключи

• Типы данных полей



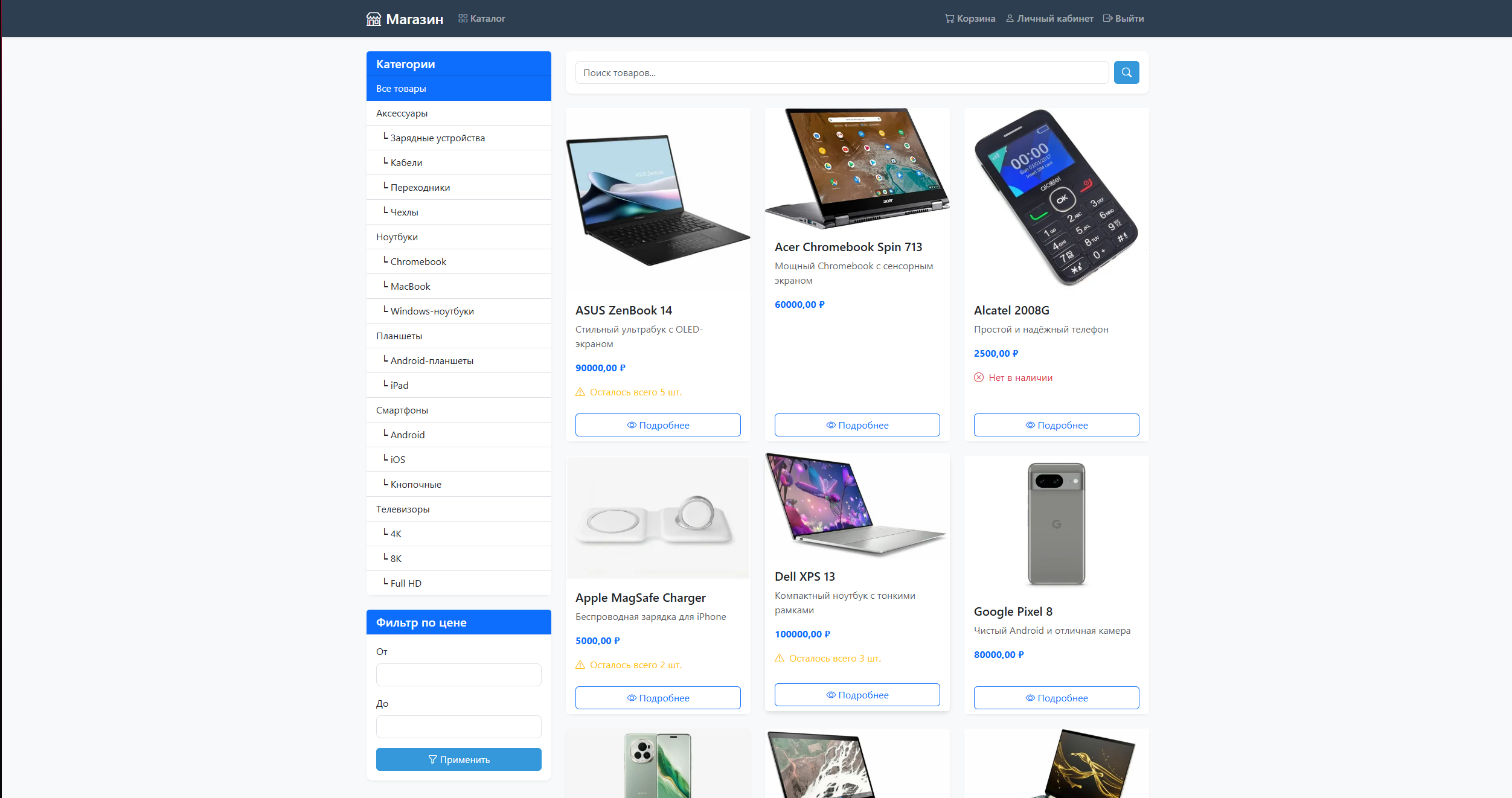
**Рисунок 1 - ERD-диаграмма базы данных интернет-магазина**

Диаграмма помогает лучше понять структуру базы данных и взаимосвязи между различными сущностями в системе.

# Приложения

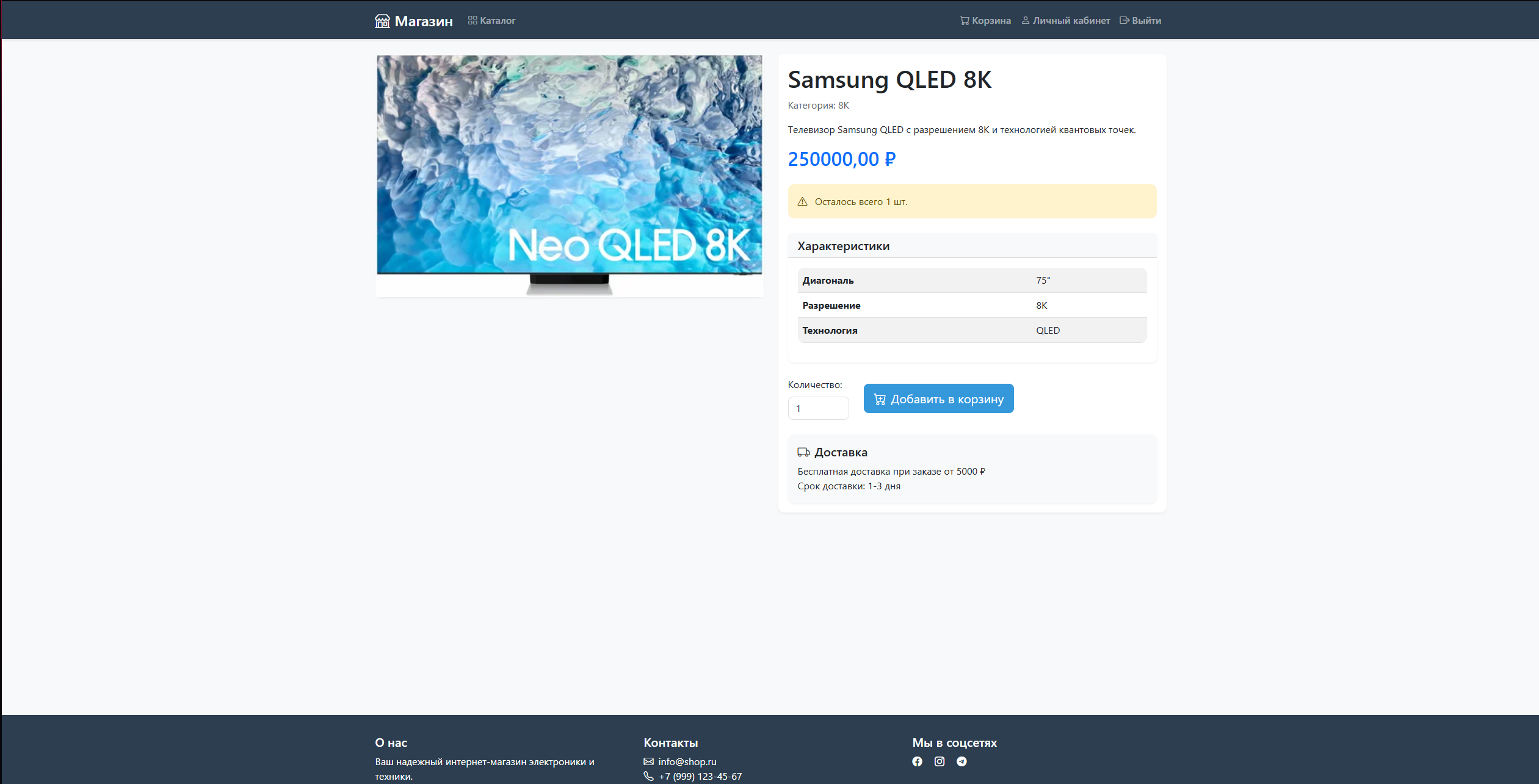
В данном разделе представлены скриншоты основных элементов интернет-магазина, демонстрирующие его функциональность и интерфейс.

## Рисунок 1 - Главная страница интернет-магазина



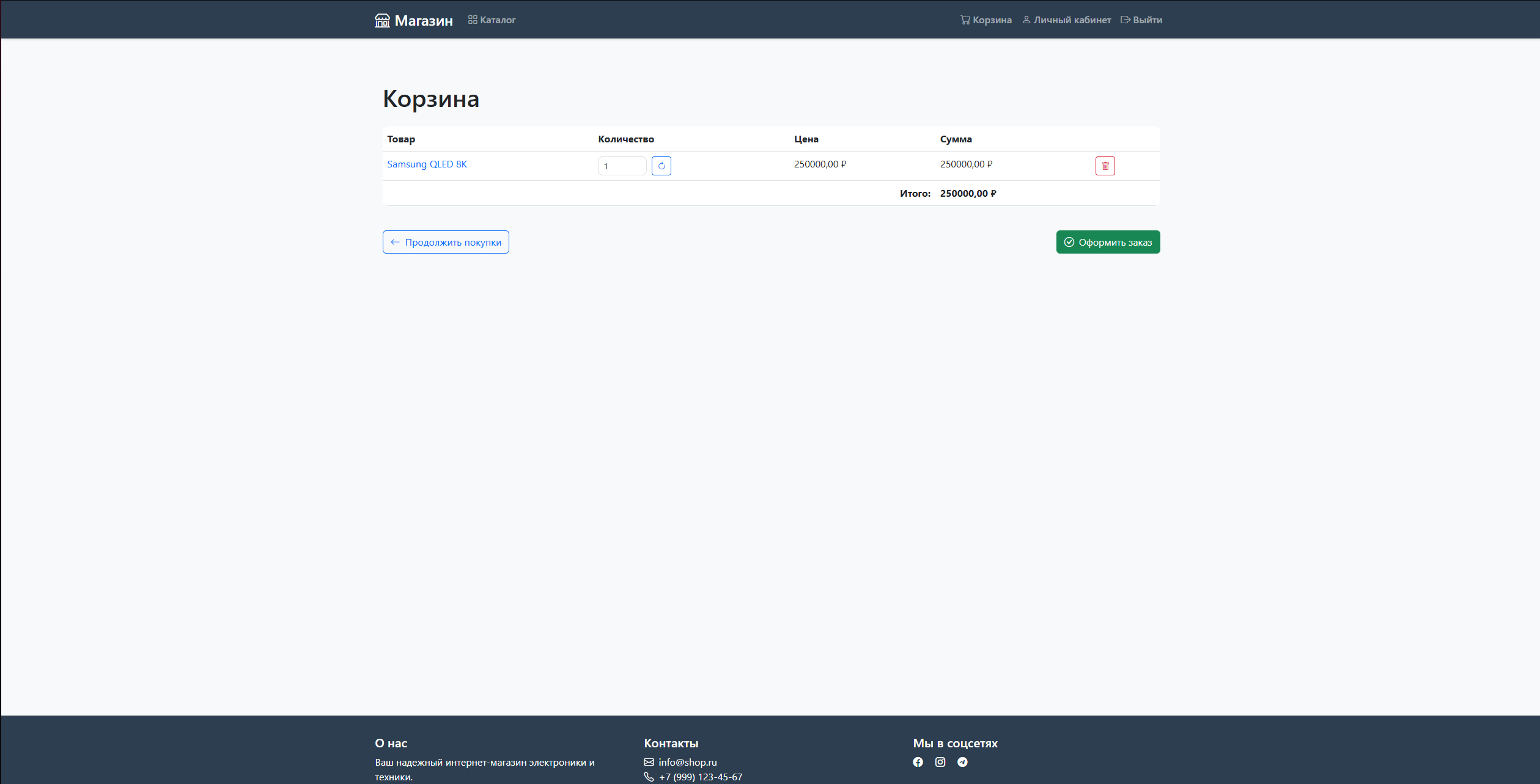
На главной странице отображается каталог товаров, навигационное меню и основные элементы интерфейса.

## Рисунок 2 - Страница товара



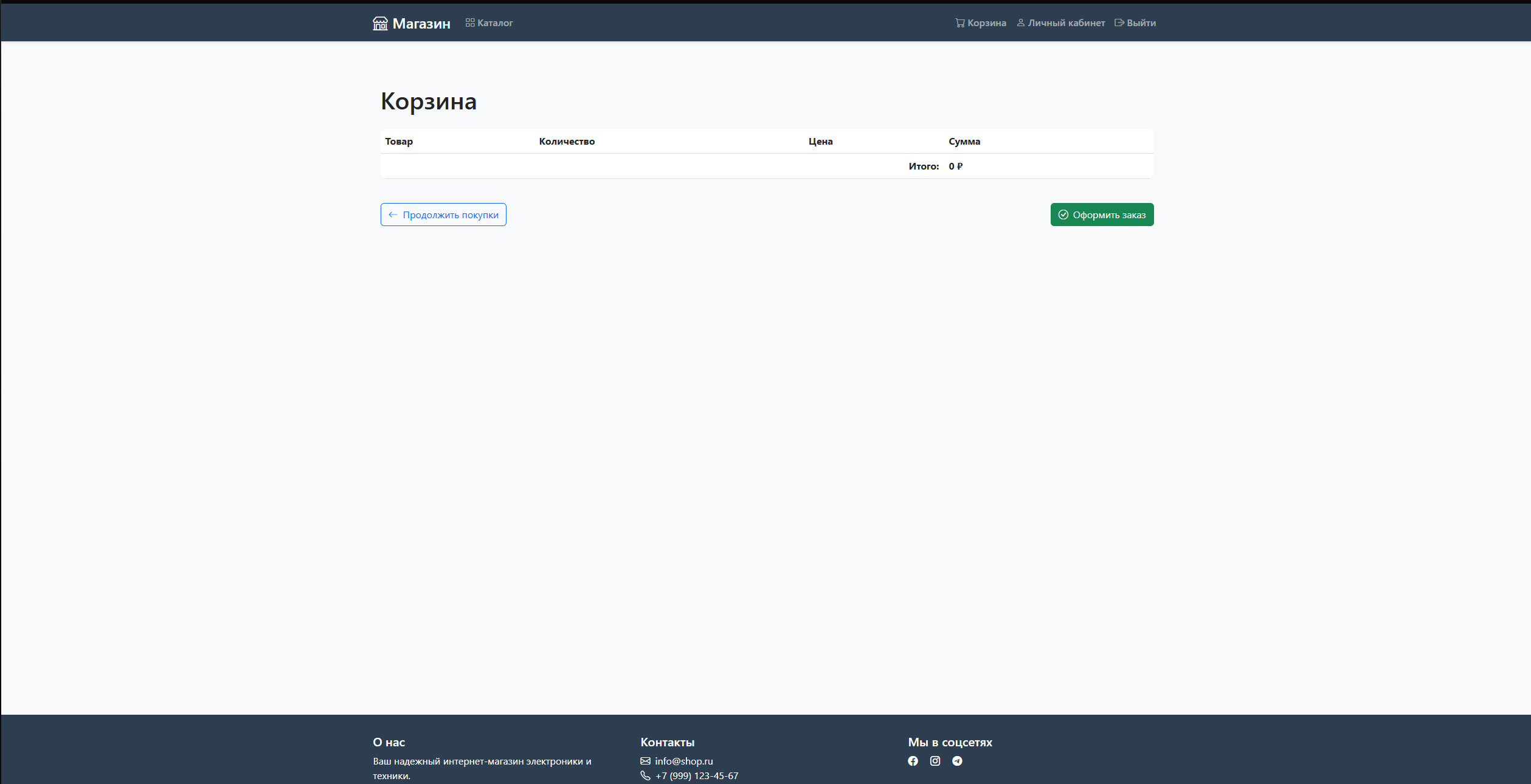
Детальная информация о товаре, включая описание, цену и кнопку добавления в корзину.

## Рисунок 3 - Корзина покупок



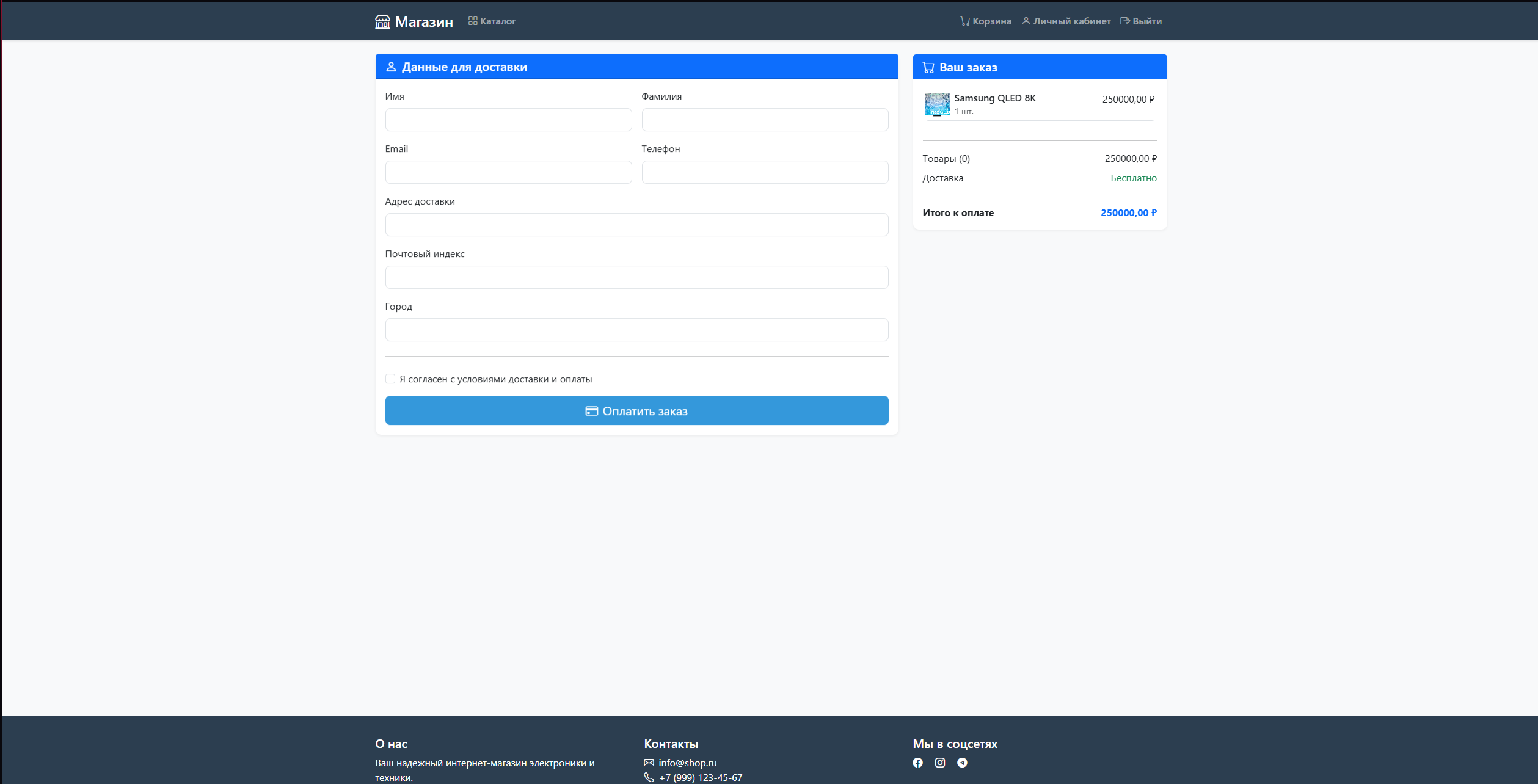
Страница корзины с списком выбранных товаров и возможностью изменения количества.

## Рисунок 4 - Пустая корзина



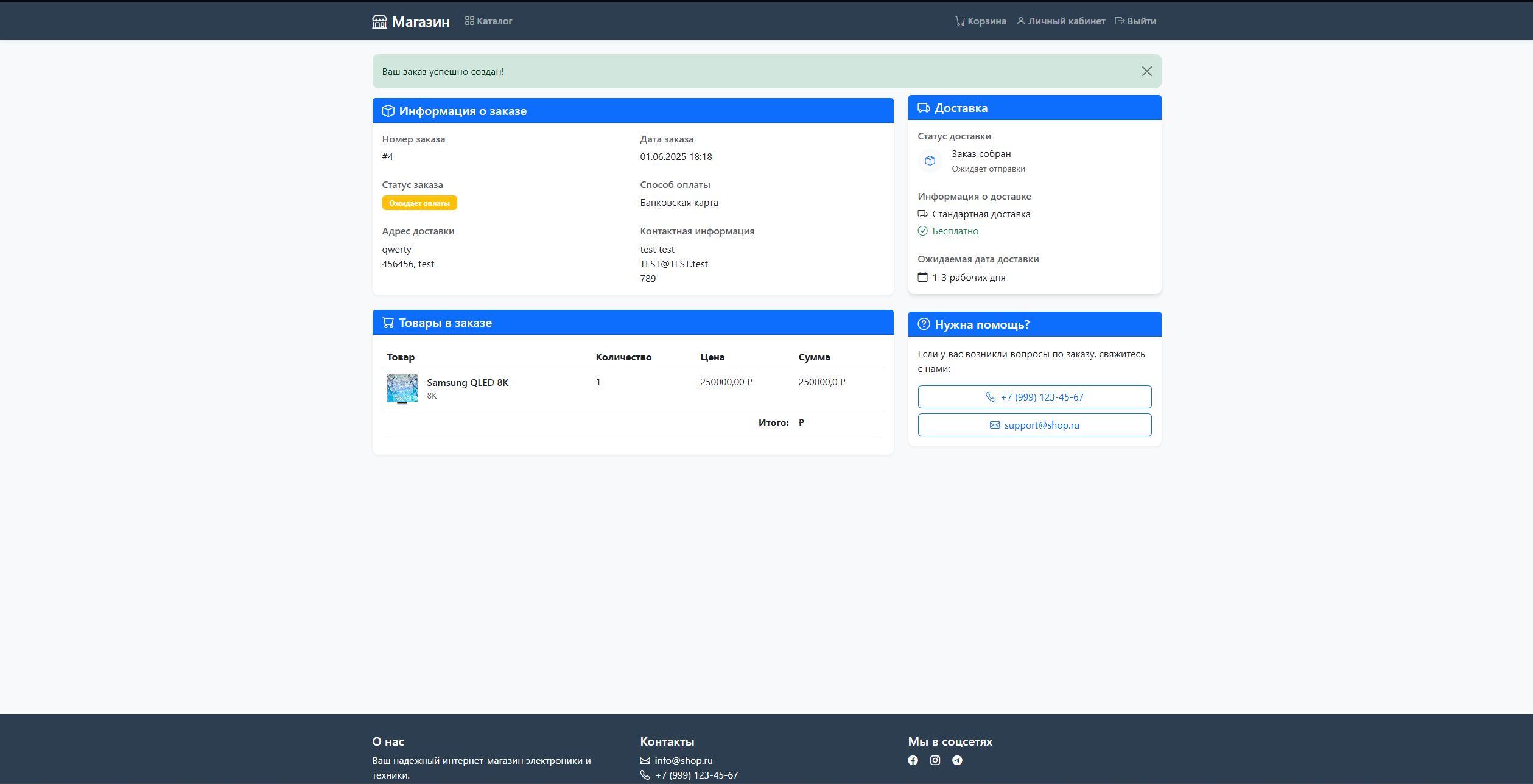
Отображение пустой корзины с предложением перейти к каталогу товаров.

## Рисунок 5 - Оформление заказа



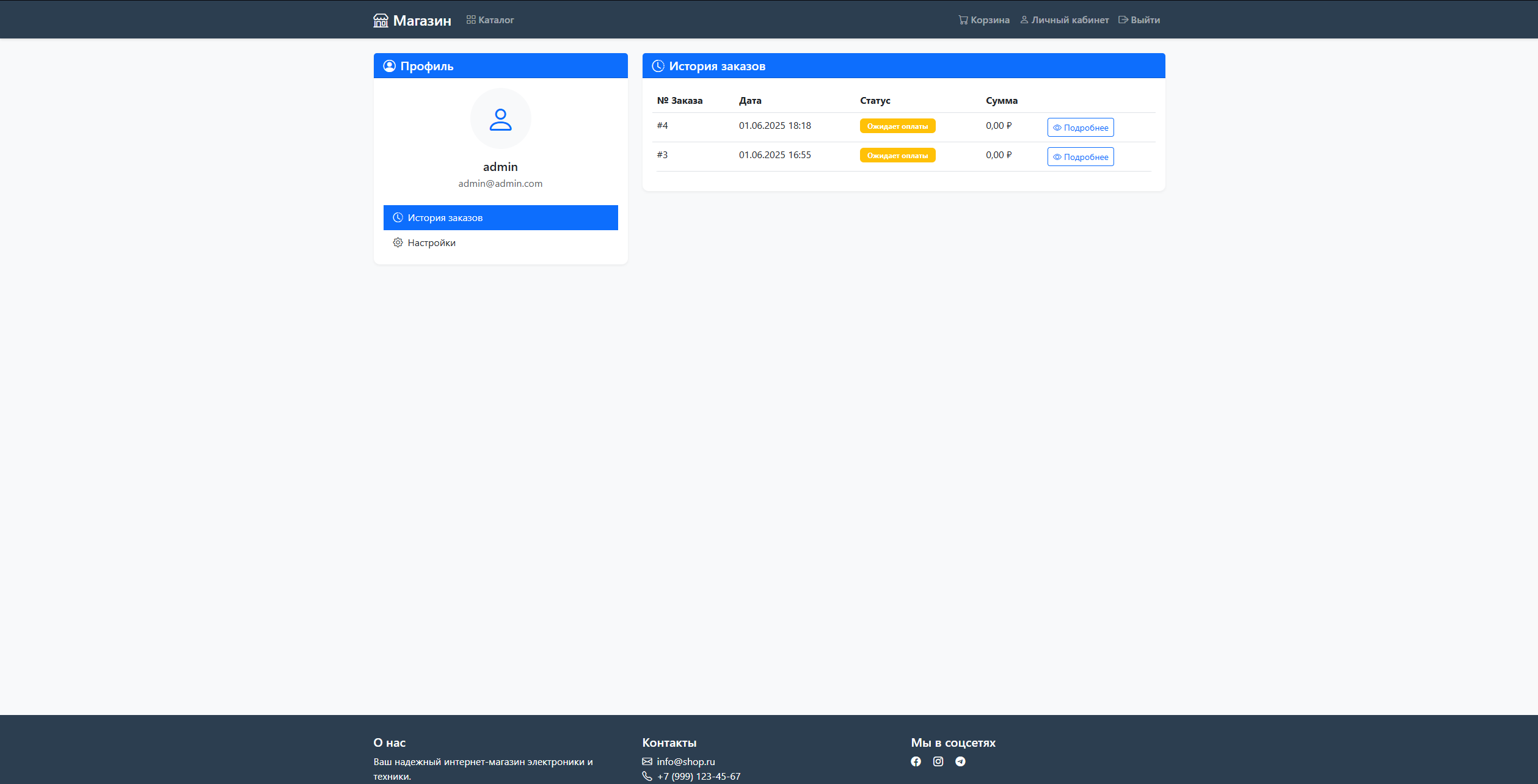
Форма оформления заказа с полями для ввода данных покупателя.

## Рисунок 6 - Страница заказа



Страница с детальной информацией о конкретном заказе.

## Рисунок 7 - Страница заказов



Страница со списком всех заказов пользователя.

## Рисунок 8 - ERD-диаграмма базы данных

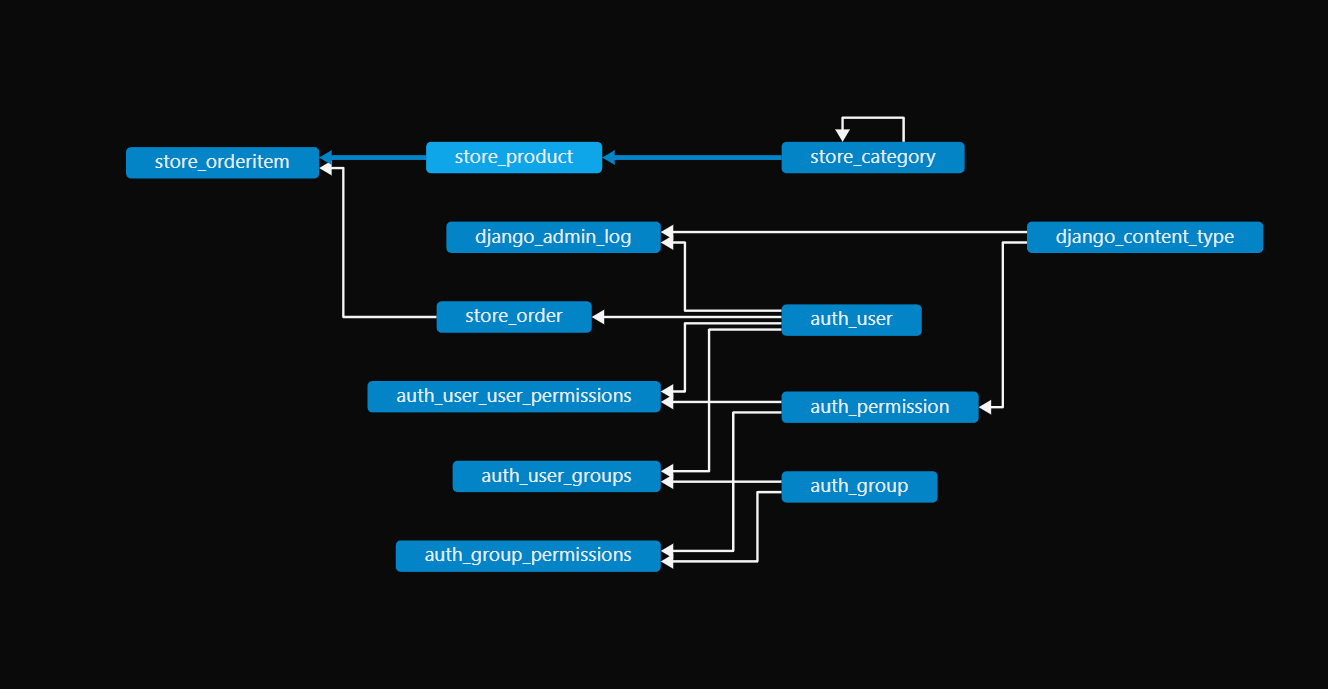


Схема базы данных, показывающая структуру таблиц и связи между ними.