



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ

Информатика и системы управления

КАФЕДРА

Программное обеспечение ЭВМ и информационные технологии

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОМУ ПРОЕКТУ

НА ТЕМУ:

Разработка интернет-магазина техники

Студент ИУ7-65Б

(Подпись, дата)

Мишин Ф.Р.
(И.О.Фамилия)

Руководитель курсового проекта

(Подпись, дата)

Леон А.Ф.
(И.О.Фамилия)

г. Москва 2020 г.

**Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ
Заведующий кафедрой ИУ 7

(Индекс)
И.В.Рудаков
(И.О.Фамилия)

« _____ » _____ 2020г.

З А Д А Н И Е **на выполнение курсового проекта**

по дисциплине Базы Данных

Студент группы ИУ7-65Б

Мишин Филипп Рафиг оглы
(Фамилия, имя, отчество)

Тема курсового проекта Разработка интернет-магазина техники

Направленность КП (учебный, исследовательский, практический, производственный, др.)
Учебный

Источник тематики (кафедра, предприятие, НИР) _____ Кафедра _____

График выполнения проекта: 25% к 4 нед., 50% к 7 нед., 75% к 11 нед., 100% к 14 нед.

Задание Разработать интернет-магазин техники. Спроектировать и реализовать базу данных приложения. Реализовать приложения для взаимодействия с базой данных. Реализовать авторизацию, возможность просматривать, фильтровать и сортировать товары, а также добавлять товары в корзину и оформлять заказ. Предусмотреть возможность добавления отзывов к товарам.

Оформление курсового проекта:

Расчетно-пояснительная записка на 20-30 листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

На защиту проекта должна быть представлена презентация, состоящая из 10-20 слайдов. На слайдах должны быть отражены: постановка задачи, использованные методы и алгоритмы, расчетные соотношения, структура комплекса программ, диаграмма классов, интерфейс, характеристики разработанного ПО.

Дата выдачи задания «19» марта 2020 г.

Руководитель курсового проекта

(Подпись, дата)

Деон А.Ф.
(И.О.Фамилия)

Студент

(Подпись, дата)

Мишин Ф.Р.
(И.О.Фамилия)

Содержание

Введение.....	4
1. Аналитический раздел	5
1.1 Формализация задачи	5
1.2 Реляционные базы данных	5
1.2.1 Структурная часть реляционной модели.....	6
1.2.2 Целостная часть реляционной модели.....	7
1.2.3 Манипуляционная часть реляционной модели.....	8
1.3 СУБД	8
Вывод.....	9
2. Конструкторский раздел.....	10
2.1 Диаграмма вариантов использования	10
2.2 ER диаграмма.....	12
2.3 Проектирование таблиц базы данных.....	13
2.4 Паттерн модель-представление-контроллер	18
2.4.1 Модели	18
2.4.2 Представления	19
2.4.3 Контроллер.....	20
2.5 Регистрация и аутентификация пользователей.....	20
Вывод.....	20
3. Технологический раздел.....	21
3.1 Выбор технологического стека.....	21
3.2 Реализация хранения данных.....	23
3.3 Реализация доступа к данным.....	26
3.4 Frontend-разработка	26
3.5 Интерфейс приложения	27
Вывод.....	33
Заключение	35
Список литературы	36

Введение

В настоящее время интернет-магазины стали также популярны, как и обычные магазины. Они имеют множество преимуществ: он доступен покупателю 24 часа в сутки, не привязан к какому-либо месту, покупателю легче товар, который ему нужен, используя фильтры и сортировку. Также популярность таких магазинов сильно возросла в текущих обстоятельствах, так как покупателю не нужно выходить из дома, чтобы совершить покупку.

Целью данной курсовой работы является разработка интернет-магазина техники.

Для достижения поставленной цели необходимо решить следующие задачи:

- проанализировать существующие СУБД;
- спроектировать базу данных для хранения и структурирования данных;
- реализовать приложение для взаимодействия с базой данных.

1. Аналитический раздел

В данном разделе выполняется постановка задачи, проводится анализ существующих СУБД и технологий.

1.1 Формализация задачи

В ходе выполнения данной курсовой работы должно быть спроектировано и реализовано клиент-серверное приложение с поддержкой следующего функционала:

- предоставляет доступ к списку всех товаров;
- просмотр информации о конкретном товаре;
- возможность добавить в корзину выбранные товары и оформлять заказ;
- фильтрация товаров по параметрам, зависящим от категории товара, сортировка по нескольким параметрам;
- регистрация и авторизация пользователей;
- возможность оставлять отзывы о товарах.

1.2 Реляционные базы данных

База данных представляет собой совокупность определенным образом организованных данных, которые хранятся в памяти вычислительной системы и отображают состояние объектов и их взаимосвязи в рассматриваемой предметной области.

Реляционная база данных – это набор отношений, имена которых совпадают с именами схем отношений в схеме базы данных.

Реляционная модель данных включает следующие компоненты:

- структурный (данные в базе данных представляют собой набор отношений);

- целостностный (отношения (таблицы) отвечают определенным условиям целостности);
- манипуляционный (манипулирования отношениями осуществляется средствами реляционной алгебры и/или реляционного исчисления). Кроме того, в состав реляционной модели данных включают теорию нормализации.

1.2.1 Структурная часть реляционной модели

Структурная часть реляционной модели описывает, из каких объектов состоит реляционная модель. Основной структурой данных, используемой в реляционной модели, являются нормализованные «парные» отношения.

Домен можно рассматривать как подмножество значений некоторого типа данных, имеющих определенный смысл. Домен характеризуется следующими свойствами:

- домен имеет уникальное имя (в пределах базы данных);
- домен определен на некотором типе данных или на другом домене;
- домен может иметь некоторое логическое условие, позволяющее описать подмножество данных, допустимых для данного домена;
- домен несет определенную смысловую нагрузку.

Атрибут отношения – это пара вида <имя_атрибута, имя_домена >. Имена атрибутов должны быть уникальны в пределах отношения. Часто имена атрибутов отношения совпадают с именами соответствующих доменов.

Схема отношения – это именованное множество упорядоченных пар <имя_атрибута, имя_домена>. Степенью или «арностью» схемы

отношения является мощность этого множества. Схема базы данных в реляционной модели – это множество именованных схем отношений.

Кортеж, соответствующий данной схеме отношения, – это множество упорядоченных пар <имя_атрибута, значение_атрибута>, которое содержит одно вхождение каждого имени атрибута, принадлежащего схеме отношения. Значение атрибута должно быть допустимым значением домена, на котором определен данный атрибут. Степень или «арность» кортежа совпадает с «арностью» соответствующей схемы отношения.

1.2.2 Целостная часть реляционной модели

В целостностной части реляционной модели фиксируются два базовых требования целостности, которые должны выполняться для любых отношений в любых реляционных базах данных. Это целостность сущностей и ссылочная целостность (или целостность внешних ключей).

Требование целостности сущностей заключается в следующем: каждый кортеж любого отношения должен отличаться от любого другого кортежа этого отношения (т.е. любое отношение должно обладать потенциальным ключом). Поддержание целостности сущностей обеспечивается средствами СУБД. Это осуществляется с помощью двух ограничений:

- при добавлении записей в таблицу проверяется уникальность их первичных ключей;
- не допускается изменение значений атрибутов, входящих в первичный ключ.

Требование ссылочной целостности состоит в следующем:

- для каждого значения внешнего ключа, появляющегося в дочернем отношении, в родительском отношении должен найтись кортеж с таким же значением первичного ключа.

Как правило, поддержание ссылочной целостности также возлагается на СУБД.

1.2.3 Манипуляционная часть реляционной модели

Манипуляционная часть реляционной модели описывает два эквивалентных способа манипулирования реляционными данными – реляционную алгебру и реляционное исчисление. Принципиальное различие между реляционной алгеброй и реляционным исчислением заключается в следующем: реляционная алгебра в явном виде предоставляет набор операций, а реляционное исчисление представляет систему обозначений для определения требуемого отношения в терминах данных отношений. Формулировка запроса в терминах реляционной алгебры носит предписывающий характер, а в терминах реляционного исчисления – описательный характер. Говоря неформально, реляционная алгебра носит процедурный характер (пусть на очень высоком уровне), а реляционное исчисление – непроцедурный характер.

1.3 СУБД

Система управления базами данных (СУБД) - приложение, обеспечивающее создание, хранение, обновление и поиск информации в базах данных.

Основные функции СУБД:

- непосредственное управление данными во внешней памяти;
- управление буферами оперативной памяти;
- управление транзакциями;
- журнализация;
- поддержка языков БД.

Вывод

В данном разделе была приведена формализация задачи, основные принципы реляционных баз данных. В качестве СУБД был выбран SQLite, а в качестве фреймворка Django.

2. Конструкторский раздел

В данном разделе будет рассмотрено проектирование ПО, представлены диаграмма вариантов использования, ER диаграмма и диаграмма базы данных. Рассмотрена регистрация и аутентификация пользователей.

2.1 Диаграмма вариантов использования

На рисунке 1 представлена Use Case диаграмма с двумя видами акторов.

Администратор: взаимодействует с базой данной, создает, редактирует и удаляет записи, добавляет новых пользователей.

Пользователь: использует приложение для составления списка интересующих его вакансий.

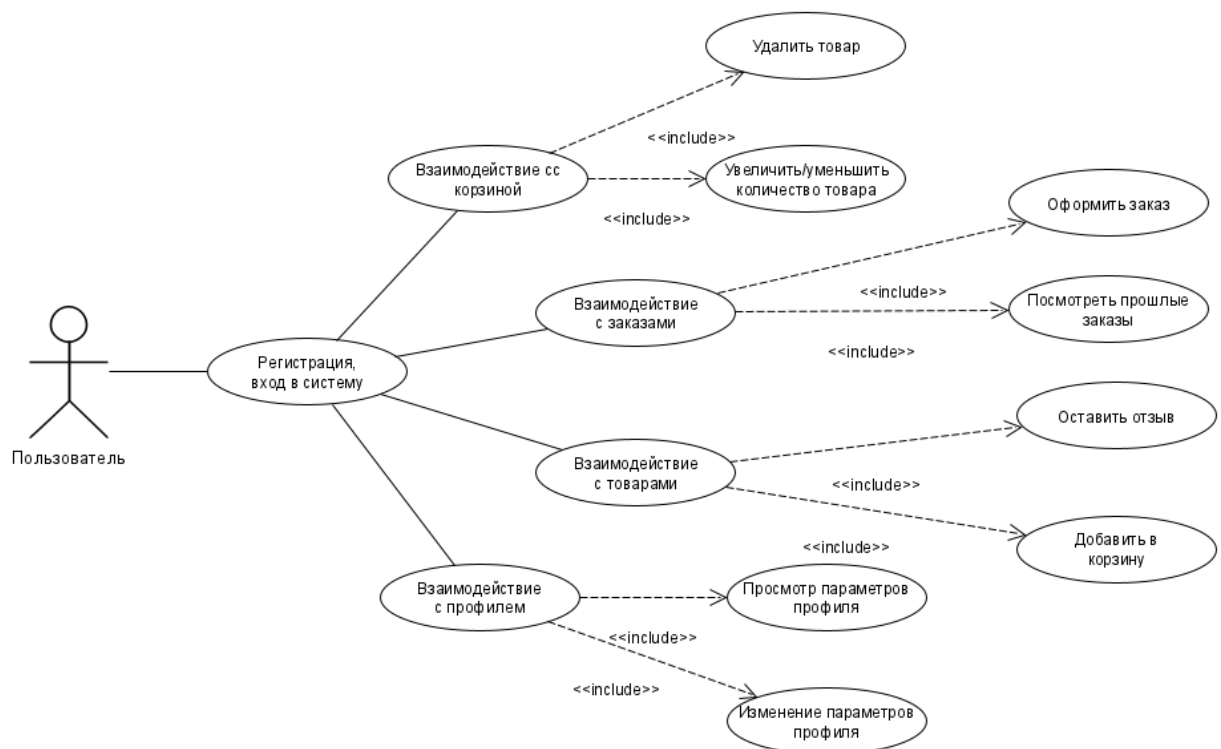




Рисунок 1. Диаграмма вариантов использования

2.2 ER диаграмма

На рисунке 2 представлена ER диаграмма приложения.

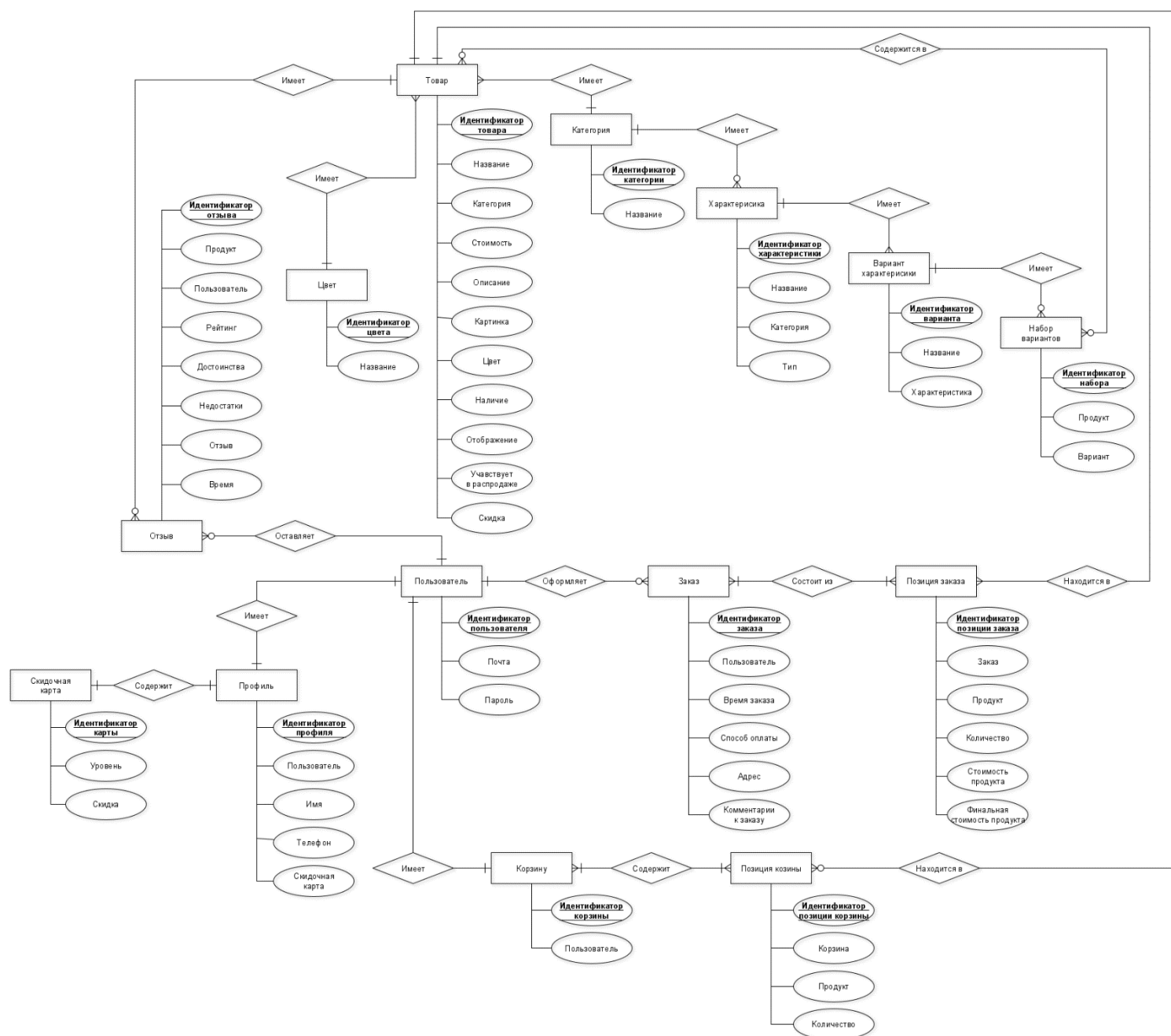


Рисунок 2. ER диаграмма

2.3 Проектирование таблиц базы данных

База данных приложения содержит следующие таблицы:

- таблица пользователей User;
- таблица профилей Profile;
- таблица скидочных карт LoyaltyCard;
- таблица товаров Product;
- таблица цветов Color;
- таблица категорий Category;
- таблица заказов Order;
- таблица позиций заказа OrderItem;
- таблица корзин Cart;
- таблица позиций корзины CartItem;
- таблица отзывов Review
- таблица характеристик товаров Feature
- таблица вариантов характеристики FeatureVariant
- таблица набора вариантов характеристик FeatureSet

Таблица User

Содержит информацию о пользователях сайта.

Таблица включает следующие поля:

- id – целочисленное поле, идентификатор пользователя;
- email – символьное поле, адрес электронной почты пользователя;
- password – символьное поле, пароль пользователя.

Таблица Profile

Содержит информацию о профиле пользователя.

Таблица включает следующие поля:

- id – целочисленное поле, идентификатор профиля;
- name – символьное поле, имя пользователя;

- phone – символьное поле, номер телефона пользователя;
- loyalty_card – целочисленное поле, идентификатор скидочной карты.

Таблица LoyaltyCard

Содержит информацию о скидочных картах.

Таблица включает следующие поля:

- id – целочисленное поле, идентификатор карты;
- name – символьное поле, уровень карты;
- discount – целочисленное поле, размер скидки по карте.

Таблица Product

Содержит информацию о товарах.

Таблица включает следующие поля:

- id – целочисленное поле, идентификатор товара;
- name – символьное поле, название товара;
- category – целочисленное поле, идентификатор категории;
- cost - целочисленное поле, стоимость товара;
- description – текстовое поле, описание товара;
- image – символьное поле, расположение изображения товара;
- color - целочисленное поле, идентификатор цвета;
- discount - целочисленное поле, размер скидки;
- on_sale – логическое поле, определяет, участвует ли товар в распродаже;
- displayed - логическое поле, определяет, отображается ли товар на сайте;
- in_stock - логическое поле, определяет, есть ли товар в наличии.

Таблица Color

Содержит информацию о цветах товаров.

Таблица включает следующие поля:

- id – целочисленное поле, идентификатор цвета;
- name – символьное поле, название цвета.

Таблица Category

Содержит информацию о категории товара.

Таблица включает следующие поля:

- id – целочисленное поле, идентификатор категории;
- name – символьное поле, название категории.

Таблица Order

Содержит информацию о заказах.

Таблица включает следующие поля:

- id – целочисленное поле, идентификатор заказа;
- user – целочисленное поле, идентификатор пользователя;
- order_time – поле даты, дата заказа;
- payment_method – символьное поле, способ оплаты;
- address – символьное поле, адрес доставки;
- notes – символьное поле, комментарии к заказу.

Таблица OrderItem

Содержит информацию о позициях заказа.

Таблица включает следующие поля:

- id – целочисленное поле, идентификатор позиции заказа;
- order – целочисленное поле, идентификатор заказа;
- product – целочисленное поле, идентификатор товара;
- quantity – целочисленное поле, количество товара;

- `item_cost` – поле дробного числа, цена товара;
- `Item_cost_final` – поле дробного типа, цена товара с примененной скидочной картой.

Таблица Cart

Содержит информацию о корзине.

Таблица включает следующие поля:

- `id` – целочисленное поле, идентификатор корзины;
- `user` – целочисленное поле, идентификатор пользователя;

Таблица CartItem

Содержит информацию о позициях корзины.

Таблица включает следующие поля:

- `id` – целочисленное поле, идентификатор позиции корзины;
- `cart` – целочисленное поле, идентификатор корзины;
- `product` – целочисленное поле, идентификатор товара;
- `quantity` – целочисленное поле, количество товара;

Таблица Review

Содержит информацию о отзывах пользователей о товаре.

Таблица включает следующие поля:

- `id` – целочисленное поле, идентификатор отзыва;
- `user` – целочисленное поле, идентификатор пользователя;
- `time` – поле даты, дата написания отзыва;
- `rating` – целочисленное поле, оценка товара;
- `advantages` – текстовое поле, плюсы товара;
- `disadvantages` – текстовое поле, минусы товара;
- `review` – текстовое поле, отзыв о товаре.

Таблица Feature

Содержит информацию о характеристиках категории продуктов.

Таблица включает следующие поля:

- id – целочисленное поле, идентификатор характеристики;
- category – целочисленное поле, идентификатор категории;
- name – символьное поле, название характеристики;
- type – символьное поле, тип характеристики – множественный выбор или одиночный.

Таблица FeatureVariant

Содержит информацию о вариантах характеристики.

Таблица включает следующие поля:

- id – целочисленное поле, идентификатор варианта;
- feature – целочисленное поле, идентификатор характеристики;
- name – символьное поле, название варианта.

Таблица FeatureSet

Содержит информацию о наборе вариантов характеристик у товаров.

Таблица включает следующие поля:

- id – целочисленное поле, идентификатор набора;
- product – целочисленное поле, идентификатор товара;
- feature_variant – целочисленное поле, идентификатор варианта.

На рисунке 3 представлена диаграмма базы данных.

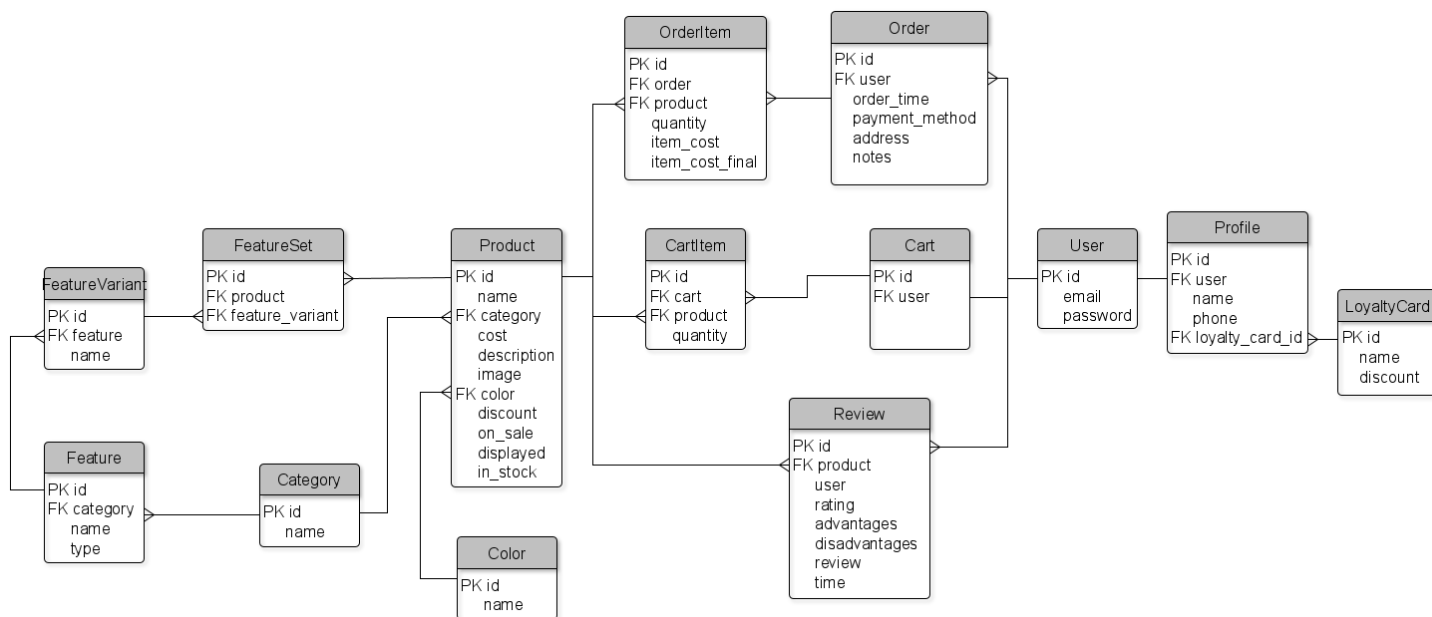


Рисунок 3. Диаграмма базы данных

2.4 Паттерн модель-представление-контроллер

Шаблон проектирования MVC предполагает разделение данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: Модель (Model), Представление (View) и Контроллер (Controller) – таким образом, что модификация каждого компонента может осуществляться независимо.

Основное преимущество такого подхода заключается в свободе объединения этих компонентов. Следовательно, каждая отдельная часть приложения, созданного с помощью Django, имеет одно назначение и может быть изменена независимо, т.е., без влияния на остальные компоненты.

2.4.1 Модели

В Django модели отображают информацию о данных. Они содержат поля и поведение данных, одна модель соответствует одной таблице в базе данных.

В данной работе будет использована технология ORM.

ORM (англ. Object-Relational Mapping, рус. объектно-реляционное отображение) — технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных». ORM позволяет удобно интегрировать модели в приложения с объектно-ориентированным стилем программирования.

2.4.2 Представления

Представление — это компонент системы, нужный для отображения пользователю модели. В Django этим занимаются представления (views) и шаблоны (templates).

В данной работе можно выделить следующие представления:

- страница со списком всех товаров;
- страница с товарами выбранной категории;
- страница с подробной информацией о выбранном товаре;
- страница с профилем пользователя;
- страница с предыдущими заказами пользователя;
- страница с подробной информацией о заказе;
- страница с аутентификацией;
- страница с регистрацией;
- страница с корзиной;
- страница оформления заказа;
- страница создания отзыва.

2.4.3 Контроллер

Контроллер определяет представление в зависимости от указаний пользователя. Он перенаправляет данные от пользователя к системе и наоборот. Использует модель и представление для реализации необходимого действия.

В Django контроллеры обеспечивают обработку HTTP запросов GET и POST. Контроллер есть у каждого представления, так как именно он отправляет пользователю запрашиваемую HTML-страницу.

2.5 Регистрация и аутентификация пользователей

Регистрация пользователя в приложении является добавлением в базу данных (таблица User) записи, содержащей необходимую информацию для аутентификации. Для этого пользователь вводит соответствующие данные в поля регистрационной формы.

Django предоставляет набор базовых инструментов для реализации web-приложения. В этот функционал включена и реализация аутентификации пользователя.

Вывод

В данном разделе была рассмотрена архитектура приложения, представлены диаграммы приложения. Был рассмотрен механизм регистрации и аутентификации пользователей.

3. Технологический раздел

В данной части приведены листинги классов для оформления таблиц базы данных, доступ к данным и frontend-разработка, а также рассмотрена интерфейс приложения.

3.1 Выбор технологического стека

В качестве языка программирования был выбран Python. Т.к. он поддерживает разные парадигмы программирование, а также обладает большим количеством фреймворков и библиотек, в том числе для доступа к различным СУБД.

Основными критериями выбора СУБД являлись поддержка реляционной модели данных (т.к. заранее известны типы хранимых данных), наличие ORM и простота тестирования.

Для проекта были рассмотрены две самые популярные СУБД: PostgreSQL и SQLite.

SQLite является компактной встраиваемой БД с интуитивным интерфейсом и удобна в тестировании. Вся база состоит из одного файла, что позволяет очень легко ее переносить с машины на машину.

Реляционная СУБД PostgreSQL ориентируется на полное соответствие стандартам и расширяемость. Помимо того, PostgreSQL при простых операциях чтения может быть значительно медленнее своего конкурента.

В связи с этим в качестве СУБД в данной работе был выбран SQLite.

СУБД SQLite поддерживается множеством фреймворков, которые содержат в себе необходимые методы обращения к базе данных.

В качестве web-framework был выбран Django, который предоставляет все необходимые инструменты для написания как frontend, так и backend частей для полноценного запуска приложения.

Django — свободный фреймворк для веб-приложений на языке Python, использующий шаблон проектирования MVC.

Сайт на Django строится из одного или нескольких приложений, которые рекомендуется делать отчуждаемыми и подключаемыми. Это одно из существенных архитектурных отличий этого фреймворка от некоторых других. Также, в отличие от других фреймворков, обработчики URL в Django конфигурируются явно при помощи регулярных выражений.

Для работы с базой данных Django использует собственный ORM, в котором модель данных описывается классами Python, и по ней генерируется схема базы данных.

К основным преимуществам фреймворка Django, которые повлияли на его выбор, стали:

- **быстрота:** Django был разработан, чтобы помочь разработчикам создать приложение настолько быстро, насколько это возможно. Это включает в себя формирование идеи, разработку и выпуск проекта, где Django экономит время и ресурсы на каждом из этих этапов;
- **полная комплектация:** Django работает с десятками дополнительных функций, которые заметно помогают с аутентификацией пользователя, картами сайта, администрированием содержимого, RSS и многим другим. Данные аспекты помогают осуществить каждый этап веб разработки;
- **безопасность:** работая в Django, вы получаете защиту от ошибок, связанных с безопасностью и ставящих под угрозу проект;
- **масштабируемость:** фреймворк Django наилучшим образом подходит для работы с самыми высокими трафиками. Следовательно, логично, что великое множество загруженных сайтов используют Django для удовлетворения требований, связанных с трафиком.

Для реализации проекта были выбраны следующие технологии:

- СУБД SQLite;
- Язык программирования python;
- framework Django.

Выбранные инструменты совместимы друг с другом и позволяют выполнить все поставленные задачи.

3.2 Реализация хранения данных

В листингах 1-14 представлены реализованные модели.

Листинг 1. Модель MyUser

```
class MyUser(AbstractBaseUser):
    email = models.EmailField(
        verbose_name='email address',
        max_length=255,
        unique=True,
    )
    is_active = models.BooleanField(default=True)
    is_admin = models.BooleanField(default=False)

    objects = MyUserManager()

    USERNAME_FIELD = 'email'
    REQUIRED_FIELDS = []
```

Листинг 2. Модель Profile

```
class Profile(models.Model):
    user = models.OneToOneField(settings.AUTH_USER_MODEL,
        on_delete=models.CASCADE)
    name = models.CharField(max_length=50, default=None, blank=True,
        null=True)
    loyalty_card = models.ForeignKey(LoyaltyCard, on_delete=models.SET_NULL,
        null=True, default=6)
```

Листинг 3. Модель LoyaltyCard

```
class LoyaltyCard(models.Model):
    name = models.CharField(max_length=50)
    discount = models.FloatField()
```

Листинг 4. Модель Order

```
class Order(models.Model):
    CARD = 'card'
```

```

CASH = 'cash'
PAYMENT_CHOICES = [
    (CARD, 'Карта'),
    (CASH, 'Наличные')
]

user = models.ForeignKey(settings.AUTH_USER_MODEL,
on_delete=models.CASCADE)
order_time = models.DateTimeField(auto_now=True)
payment_method = models.CharField(
    max_length=4,
    choices=PAYMENT_CHOICES,
    default=CASH
)
address = models.CharField(max_length=250, default='')
notes = models.TextField(blank=True, null=True)

```

Листинг 5. Модель OrderItem

```

class OrderItem(models.Model):
    order = models.ForeignKey(Order, on_delete=models.CASCADE)
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    quantity = models.IntegerField(default=1)
    item_cost = models.FloatField()
    item_cost_final = models.FloatField()

```

Листинг 6. Модель Cart

```

class Cart(models.Model):
    user = models.OneToOneField(settings.AUTH_USER_MODEL,
on_delete=models.CASCADE)

```

Листинг 7. Модель CartItem

```

class CartItem(models.Model):
    cart = models.ForeignKey(Cart, on_delete=models.CASCADE)
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    quantity = models.IntegerField(default=1)

```

Листинг 8. Модель Review

```

class Review(models.Model):
    user = models.ForeignKey(settings.AUTH_USER_MODEL,
on_delete=models.CASCADE)
    product = models.ForeignKey('shop.Product', on_delete=models.CASCADE)
    rating = models.IntegerField()
    advantages = models.TextField()
    disadvantages = models.TextField()
    review = models.TextField()
    date_posted = models.DateTimeField(auto_now=True)

```

Листинг 9. Модель Product

```

class Product(models.Model):
    name = models.CharField(max_length=50)

```



```

category = models.ForeignKey(Category, on_delete=models.CASCADE)
color = models.ForeignKey(Color, on_delete=models.CASCADE)
cost = models.IntegerField()
description = models.TextField()
image = models.ImageField(default='product_default.jpg',
upload_to='product_pics')
displayed = models.BooleanField(default=True)
in_stock = models.BooleanField(default=True)
on_sale = models.BooleanField(default=False)
discount = models.IntegerField(default=0)

```

Листинг 10. Модель Color

```

class Color(models.Model):
    name = models.CharField(max_length=50)

```

Листинг 11. Модель Category

```

class Category(models.Model):
    name = models.CharField(max_length=50)

```

Листинг 12. Модель Feature

```

class Feature(models.Model):
    category = models.ForeignKey(Category, on_delete=models.CASCADE)
    name = models.CharField(max_length=50)

    CHECKBOX = 'checkbox'
    RADIOBUTTON = 'radiobutton'
    TYPES = [
        (CHECKBOX, 'Множественный выбор'),
        (RADIOBUTTON, 'Одиночный выбор')
    ]

    type = models.CharField(
        max_length=11,
        choices=TYPES,
        default=CHECKBOX
    )

```

Листинг 13. Модель FeatureVariant

```

class FeatureVariant(models.Model):
    feature = models.ForeignKey(Feature, on_delete=models.CASCADE)
    name = models.CharField(max_length=50)

```

Листинг 14. Модель FeatureSet

```

class FeatureSet(models.Model):
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    feature_variant = models.ForeignKey(FeatureVariant,
on_delete=models.CASCADE)

```

3.3 Реализация доступа к данным

Чтобы обеспечить доступ к данным, необходимо создать форму, позволяющую добавлять и изменять записи в таблицах.

Центр данного механизма — класс `Form`, который описывает структуру объекта, его поведение и представление.

Реализация формы для доступа к данным представлена в листинге 7.

Листинг 7. Форма для регистрации

```
class UserCreationForm(forms.ModelForm):
    password1 = forms.CharField(label='Password', widget=forms.PasswordInput)
    password2 = forms.CharField(label='Password confirmation',
                                widget=forms.PasswordInput)

    class Meta:
        model = MyUser
        fields = ('email',)

    def clean_password2(self):
        password1 = self.cleaned_data.get("password1")
        password2 = self.cleaned_data.get("password2")
        if password1 and password2 and password1 != password2:
            raise forms.ValidationError("Passwords don't match")
        return password2

    def save(self, commit=True):
        user = super().save(commit=False)
        user.set_password(self.cleaned_data["password1"])
        if commit:
            user.save()
        return user
```

3.4 Frontend-разработка

Дизайн сайта настроен с помощью технологии Bootstrap.

Bootstrap — это инструмент с открытым исходным кодом для разработки web-приложений с помощью HTML, CSS и JS. Включает в себя HTML- и CSS-шаблоны оформления для типографики, веб-форм, кнопок, меток, блоков навигации и прочих компонентов веб-интерфейса, включая JavaScript-расширения.

Django предоставляет инструмент шаблонизатора, который дает возможность вносить динамические данные в html с backend. С помощью

шаблонизатора есть возможность проверять данные, изменяя элементы страницы в зависимости от результата проверки.

При рендеринге шаблона переменные в двойных фигурных скобках будут заменяться на вычисленные значения.

Например, в шаблоне профиля, представленном в листинге 8, шаблонизатор `{{ form|crispy }}` вставляет на страницу созданные формы пользователя и профиля.

Листинг 8. Шаблон профиля

```
<form method="POST" enctype="multipart/form-data">
  {% csrf_token %}
  <fieldset class="form-group">
    <legend class="border-bottom mb-4">Информация о профиле</legend>
    {{ u_form|crispy }}
    {{ p_form|crispy }}
  </fieldset>
  <div class="form-group">
    <button class="btn btn-outline-info" type="submit">Сохранить
профиль</button>
  </div>
</form>
```

3.5 Интерфейс приложения

В зависимости от того, выполнил ли пользователь аутентификацию, интерфейс выглядит по-разному. Зарегистрированному пользователю доступна возможность открыть свой профиль и посмотреть корзину. Незарегистрированный пользователь не может этого увидеть до тех пор, пока не выполнит вход в систему.

На рисунках 4 и 5 представлен интерфейс зарегистрированного и незарегистрированного пользователей соответственно.

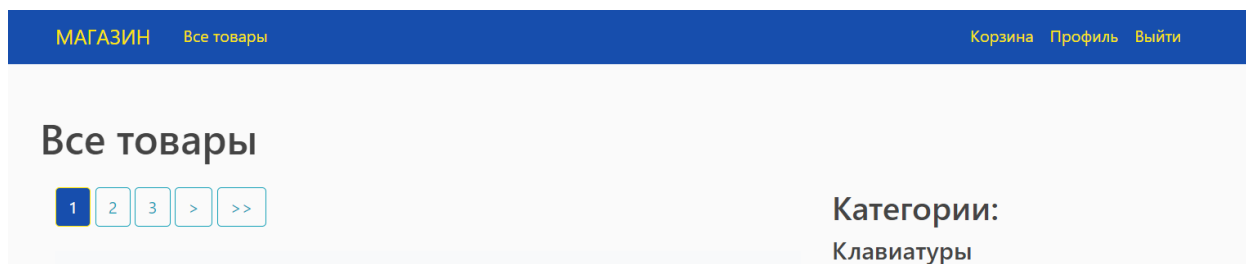


Рисунок 4. Интерфейс зарегистрированного пользователя

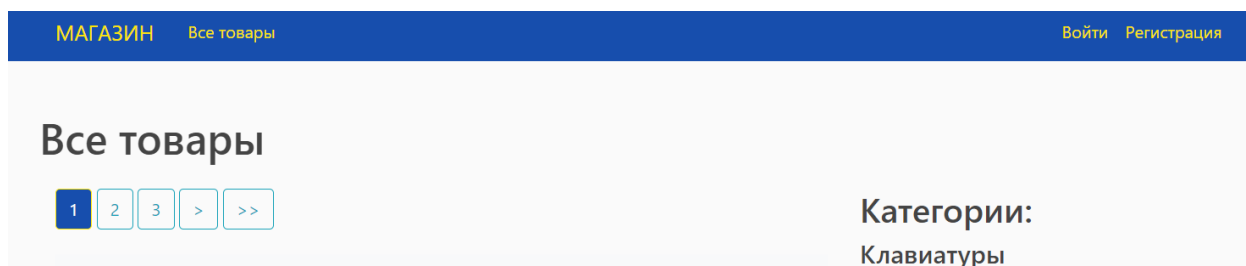


Рисунок 5. Интерфейс незарегистрированного пользователя

На рисунках 6 и 7 представлены страницы аутентификации и регистрации пользователей.

The image shows a login form titled 'Log In'. It contains two input fields: 'Email address*' and 'Password*'. Below the 'Password*' field is a 'Log In' button. At the bottom of the form, there is a link that says 'Need an Account? Sign Up Now'.

Рисунок 6. Страница аутентификации

Join Today

Email address*

Password*

Password confirmation*

[Sign Up](#)

Already Have an Account? [Sign In](#)

Рисунок 7. Страница регистрации

На рисунках 8 и 9 представлен интерфейс поиска товаров в категории , используя фильтры и сортировку.

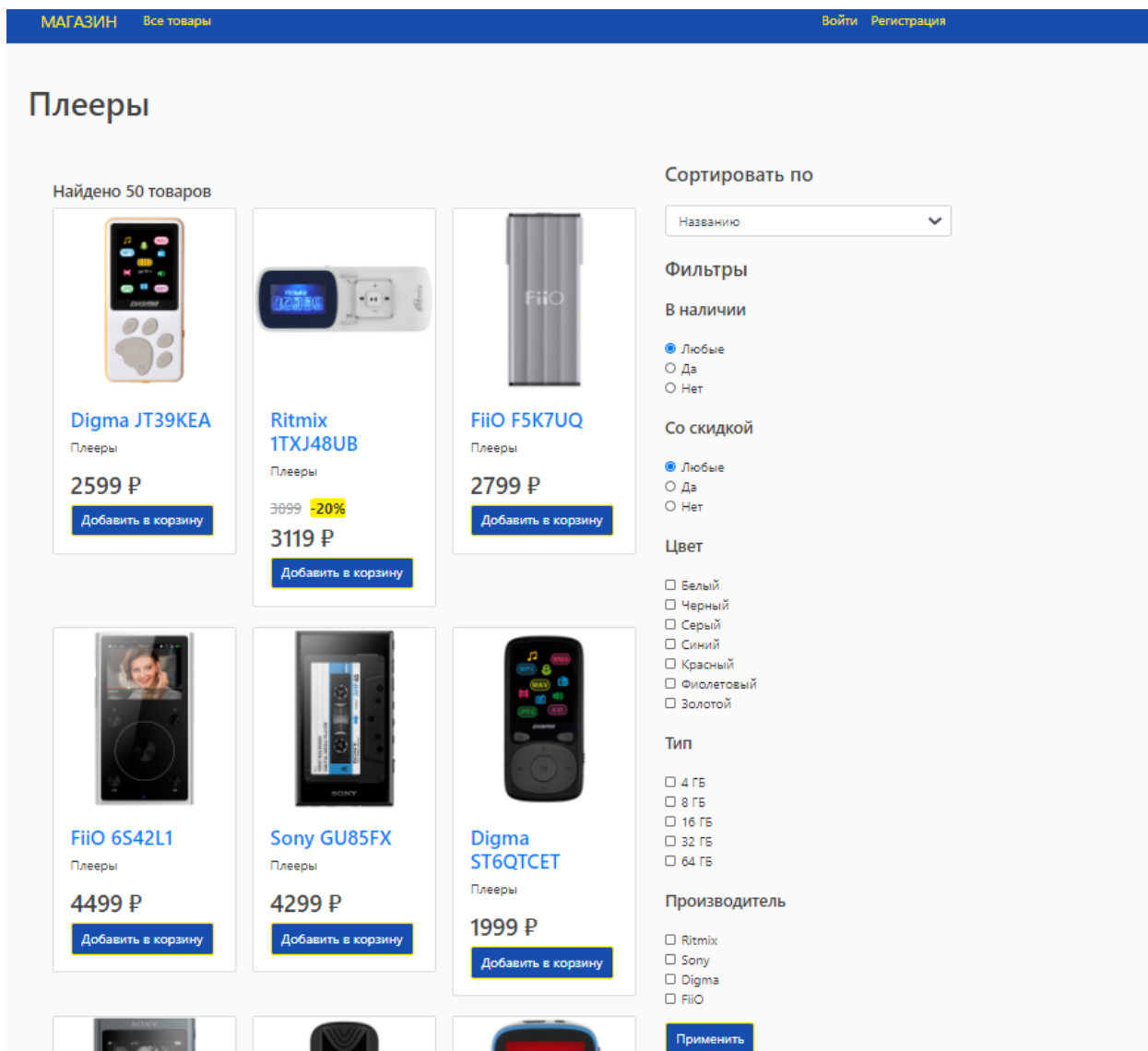


Рисунок 8. Страница до введения параметров

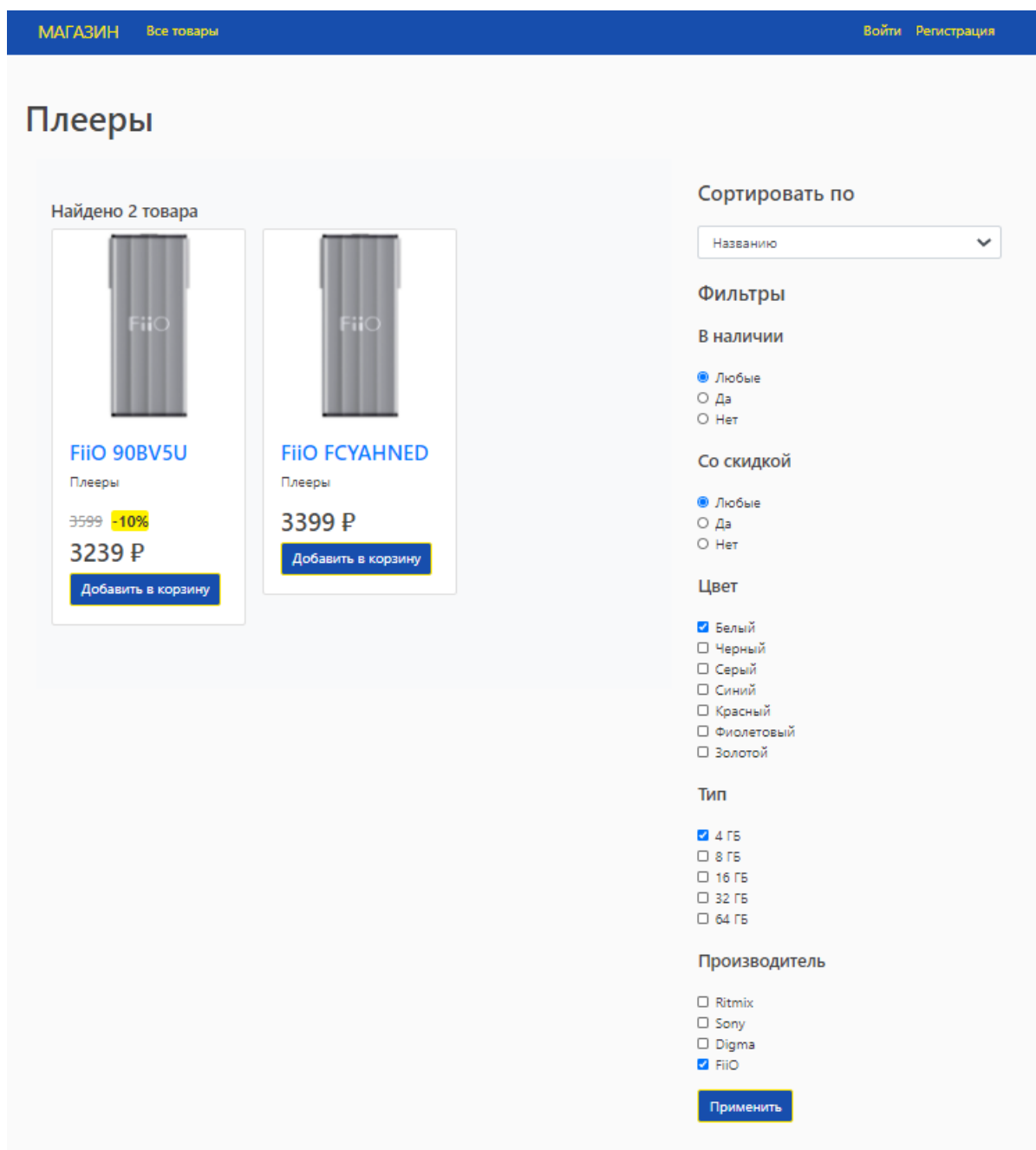





Рисунок 9. Страница со списком найденных товаров

На рисунке 10 представлен интерфейс корзины.

Товары в Вашей корзине

	A4Tech 9VY6BK8S Количество: 1	2079.2 Р
<div>- + Удалить</div>		
	Microsoft 3RIVIZE6 Количество: 1	3499.0 Р
<div>- + Удалить</div>		
	Brother 2X31V91 Количество: 1	10009.3 Р
<div>- + Удалить</div>		

Информация о заказе


В корзине 3 товара

Стоимость: 15587.5 Р

[Оформить заказ](#)

Рисунок 10. Корзина с несколькими добавленными товарами

На рисунке 11 представлена страница с информацией о товаре.



A4Tech 9VY6BK8S

Мыши

2599 **-20%**

2079 Р

[Перейти в корзину](#)

[Изменить отзыв](#)

[Удалить отзыв](#)

Подробнее о товаре

Цвет	Красный
Производитель	A4Tech
Проводные	Да

Отзывы

new_acc@mail.ru

☆☆☆☆☆

Плюсы: Удобно нажимать на кнопки.

Минусы: Этой мышью не накормить змею.

Отзыв: Покупайте, если вам нужна мышь для компьютера!!!!

29.09.20

Рисунок 11. Подробная информация о товаре и отзывы пользователей.

Вывод

Были рассмотрены листинги реализованных классов для оформления таблиц базы данных, доступ к данным и frontend-разработка. Был рассмотрен интерфейс приложения и его основные функции.

Заключение

В ходе проделанной работы были проанализированы основные принципы реляционных баз данных и реляционные СУБД.

Спроектирована база данных, состоящая из нескольких сущностей.

С помощью выбранных технологий было реализовано приложение для взаимодействия с базой данных. В ходе работы мы научились создавать интернет-магазин и работать с ним.

Список литературы

1. Документация Python 3 [Электронный ресурс]. – Режим доступа: URL: <https://docs.python.org/3/> (Дата Обращения - 30.05.2020)
2. Паттерн MVC [Электронный ресурс]. - Режим доступа: URL: https://professorweb.ru/my/WPF/documents_WPF/level36/36_3.php (дата обращения: 31.05.2020).
3. Документация к Django [Электронный ресурс]. - Режим доступа: URL: <https://docs.djangoproject.com/en/3.0/> (дата обращения: 31.05.2020).
4. Документация к Bootstrap [Электронный ресурс]. - Режим доступа: URL: <https://getbootstrap.com/docs/4.5/getting-started/introduction/> (дата обращения: 31.05.2020).