

МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

Факультет Компьютерных наук

Кафедра информационных технологий управления

Онлайн каталог кондитерских изделий «Sweet World»
Курсовая работа по дисциплине
«Технологии программирования»

09.03.02 Информационные системы и технологии
Информационные системы и технологии в управлении
предприятием

Зав. кафедрой _____ д.т.н., профессор М.Г.Матвеев __. __20__

Обучающийся _____ Д.С. Агабабян, 3 курс, д/о

Обучающийся _____ А.В. Гранкина, 3 курс, д/о

Руководитель _____ В.С. Тарасов, ст. преподаватель

Руководитель _____ К.В. Зенин, преподаватель

Содержание

Содержание	2
Введение	4
1 Постановка задачи.....	5
1.1 Требования к разрабатываемой системе	5
1.1.1 Функциональные требования	5
1.1.2 Нефункциональные требования	6
1.2 Требования к архитектуре	6
1.3 Задачи, решаемые в процессе разработки.....	7
2 Анализ предметной области	9
2.1 Терминология (гlossарий) предметной области	9
2.2 Обзор аналогов.....	11
2.2.1 Объединенные кондитеры	11
2.2.2 Конфил.....	12
2.2.3 Славянка	14
2.3 Диаграммы, иллюстрирующие работу системы.....	15
2.3.1 Диаграмма прецедентов (Use case).....	15
2.3.2 Диаграмма последовательности (Sequence diagram)	18
2.3.3 Диаграмма состояний (Statechart diagram)	20
2.3.4 Диаграмма деятельности (Activity diagram).....	21
2.3.5 Диаграмма классов (Class diagram)	22
2.3.6 Диаграмма объектов (Object diagram)	23
2.3.7 Диаграмма сотрудничества (Collaboration diagram)	24
2.3.8 Диаграмма развертывания (Deployment diagram).....	25
2.3.9 Диаграмма IDEF0	25

2.3.10 ER-диаграмма	28
3 Реализация	29
3.1 Средства реализации	29
3.1.1 Средства реализации серверной части приложения	29
3.1.2 Средства реализации клиентской части приложения	30
3.2 Реализация серверной (backend) части приложения	32
3.3 Реализация клиентской (frontend) части приложения	36
3.4 Навигация по приложению	40
3.4.1 Для неавторизованного пользователя	40
3.4.2 Для авторизованного пользователя	44
3.4.3 Для администратора	46
4 Тестирование	49
4.1 Модульное тестирование (unit-тесты).....	49
4.2 Дымовое тестирование.....	50
4.3 Тестирование пользовательского интерфейса.....	53
Заключение	55
Список использованной литературы.....	56

Введение

В настоящее время Интернет-технологии стали неотъемлемой частью современного мира, ведь именно они позволяют получать доступ к большому количеству информации, не выходя из дома. В связи с этим все больше компаний и индивидуальных предпринимателей начинают использовать Интернет для продажи своих товаров и услуг. В этом контексте онлайн-каталоги играют важную роль, предоставляя пользователям возможность ознакомиться с ассортиментом товаров и услуг, которые они могут приобрести, и предоставляя детальную информацию о разнообразии продуктов и услуг, а также их ценах и уникальных характеристиках.

Одной из наиболее популярных областей, где используются интернет-каталоги, является сфера кондитерской продукции. Они помогают людям ознакомиться с товарами и выбрать подходящие по вкусу и стилю, часто содержат подробные описания продуктов и изображения, что позволяет покупателям сделать осознанный выбор. Кондитерские каталоги также обеспечивают привлекательную визуальную презентацию продукции, что может заинтересовать покупателя и способствует принятию решения о покупке, а также отражают изменения во вкусах и предпочтениях потребителей, позволяют следить за новыми тенденциями в кондитерском искусстве и обеспечивают удобство и простоту в ведении бизнеса.

Кондитерские каталоги важны для людей, которые любят сладости и хотят наслаждаться свежими и качественными продуктами. Они помогают сохранять культуру и традиции кондитерского мастерства, а также являются важным элементом гастрономической культуры нашего общества.

В данной курсовой работе был реализован интернет-каталог кондитерских изделий, который будет предоставлять пользователям подробную информацию о продуктах.

1 Постановка задачи

Данный проект предназначен для обеспечения пользователей способностью просмотра каталога кондитерских изделий по популярным товарам от ведущих предприятий России, в результате чего заказчик сможет иметь возможность получать прибыль с рекламы данных товаров.

Помимо этого, целью данного проекта является разработка сайта с определенной выборкой продукции для людей с разными моделями пищевого поведения, что позволит расширить границы целевой аудитории.

1.1 Требования к разрабатываемой системе

1.1.1 Функциональные требования

К разрабатываемому приложению выдвигаются следующие функциональные требования:

- Осуществление просмотра продукции по категориям (кондитерским компаниям).
- Получение информации о пищевой и энергетической ценности товаров.
- Осуществление просмотра кондитерских товаров для диабетиков с низким содержанием сахара.
- Осуществление просмотра кондитерских товаров для худеющих людей с отсутствием глютена.
- Добавление кондитерских изделий в список «Избранное» авторизованным пользователем.
- Очистка списка «Избранное» по каждому конкретному товару авторизованным пользователем.

- Возможность добавления своего отзыва к конкретной продукции авторизованным пользователем.
- Возможность просмотра комментариев других пользователей.
- Добавление удаление, просмотр и редактирование товаров, категорий (кондитерских компаний), комментариев и записей пользователей администратором.
- Редактирование прав пользователей администратором.

1.1.2 Нефункциональные требования

К разрабатываемому приложению выдвигаются следующие нефункциональные требования:

- Приложение должно отвечать на запросы пользователей в течение нескольких секунд.
- Приложение должно обладать интерфейсом, выполненном в едином стиле со всем необходимым набором функций, чтобы с ним могли работать пользователи различных возрастных и культурных групп.
- Приложение должно использовать современные технологии и инструменты разработки.

1.2 Требования к архитектуре

Список требований к архитектуре:

- Приложение должно быть построено с использованием протоколов HTTP.
- Для хранения информации необходимо использовать реляционную базу данных.

- Клиентская часть приложения должна быть написана с использованием технологий frontend разработки, таких как HTML, CSS, JavaScript.
- Серверная часть приложения должна быть написана с использованием технологий backend разработки, таких как Python и Django [1] на основе архитектурного паттерна MVC. Выбор фреймворка Django объясняется тем, что он включает в себя большое количество готового функционала. И, как правило, проекты, написанные на данном фреймворке, обладают быстрой загрузкой, могут хранить огромные данные на сервере и по умолчанию создают панель администратора для редактирования информации на сайте.

1.3 Задачи, решаемые в процессе разработки

Процесс организации данного веб-приложения построен на основе гибкой методологии Kanban.

В процессе разработки интернет-каталога кондитерских изделий будут решаться следующие задачи:

- Анализ предметной области: необходимо изучить особенности работы интернет-каталога кондитерских изделий.
- Проектирование базы данных: на основе полученных требований необходимо разработать структуру базы данных, которая будет использоваться в приложении.
- Разработка серверной части приложения: на этом этапе необходимо разработать серверную часть приложения, которая будет отвечать за обработку запросов клиента и взаимодействие с базой данных. Для этого используется фреймворк Django.

- Разработка клиентской части приложения: клиентская часть приложения должна быть написана с использованием современных технологий frontend разработки, таких как HTML, CSS, JavaScript.
- Тестирование и отладка: на этом этапе необходимо провести тестирование и отладку приложения, чтобы убедиться, что оно соответствует требованиям, определенным в начале проекта.

2 Анализ предметной области

2.1 Терминология (гlossарий) предметной области

Веб-приложение — клиент-серверное приложение, в котором клиент взаимодействует с веб-сервером при помощи браузера.

Фреймворк — программная платформа, определяющая структуру программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.

Клиент (клиентская сторона) — сайт, который предоставляет пользователю взаимодействовать со всей системой.

Сервер (серверная часть) — компьютер, обслуживающий другие устройства (клиентов) и предоставляющий им свои ресурсы для выполнения определенных задач.

Backend — логика работы сайта, внутренняя часть продукта, которая находится на сервере и скрыта от пользователя.

Frontend — презентационная часть информационной или программной системы, ее пользовательский интерфейс и связанные с ним компоненты.

GitHub — веб-сервис для хостинга IT-проектов и их совместной разработки.

Неавторизованный пользователь — пользователь, не прошедший авторизацию или не зарегистрированный в системе.

Авторизованный пользователь — пользователь, прошедший авторизацию в системе.

MVC — схема разделения данных приложения и управляющей логики на три отдельных компонента: модель, представление и контроллер - таким

образом, что модификация каждого компонента может осуществляться независимо.

CSS — формальный язык описания внешнего вида документа, написанного с использованием языка разметки (HTML, XHTML, XML).

HTML — стандартизированный язык гипертекстовой разметки веб-страниц в браузере.

JavaScript — язык программирования высокого уровня, который используется для написания frontend- и backend-частей сайтов, а также мобильных приложений.

PostgreSQL — реляционная база данных с открытым кодом.

React — JavaScript-библиотека для создания пользовательских интерфейсов [2].

Сериализация — это процесс преобразования объектов Python в поток байтов, который может быть сохранен в файле или передан по сети.

Десериализация — это процесс получения потока байтов и преобразования его в объект Python.

Docker — это открытая платформа для разработки, доставки и работы с приложениями. С ее помощью можно создавать и развертывать приложения в легковесных, подключаемых контейнерах, которые могут быть запущены и масштабированы на любой платформе.

2.2 Обзор аналогов

2.2.1 Объединенные кондитеры

Холдинг «Объединенные кондитеры» является крупнейшим кондитерским предприятием в Восточной Европе, объединяя 19 российских фабрик. По результатам международного рейтинга «Global Top-100» «Объединенные кондитеры» входят в 20-ку крупнейших мировых производителей по объемам продаж.

Предприятиями производятся все виды кондитерских изделий: шоколад, конфеты в коробках, весовые конфеты, карамель, ирис, зефир, вафли, торты, мармелад и восточные сладости. Интерфейс приложения представлен на Рисунке 1.

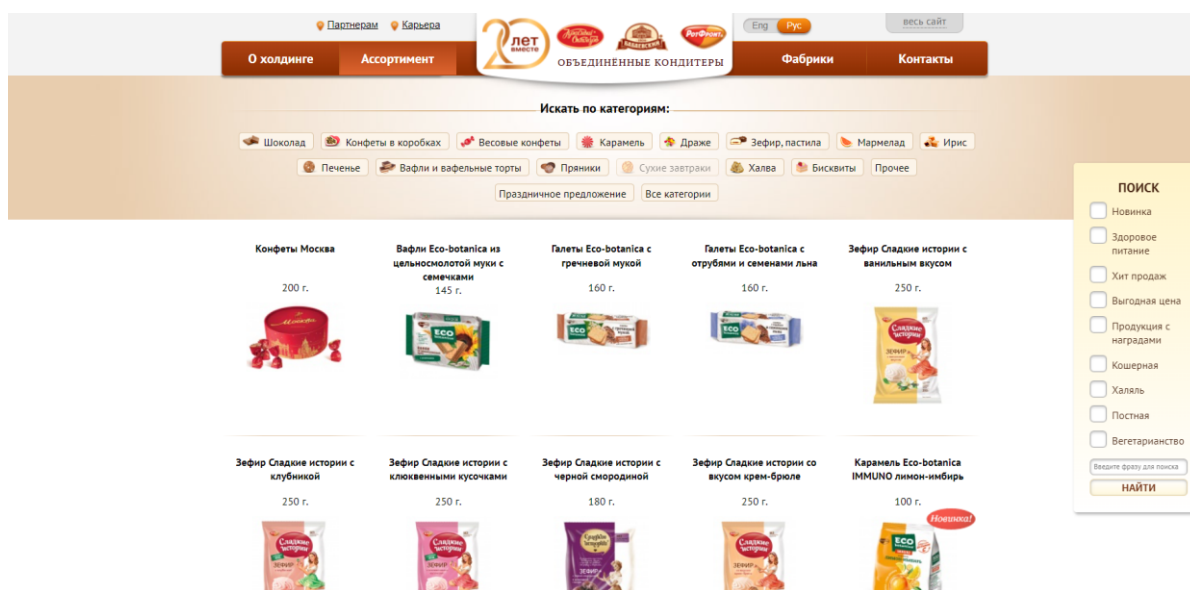


Рисунок 1 - Внешний вид каталога «Объединенные кондитеры»

Каталог «Объединенные кондитеры» обладает следующим рядом преимуществ:

- Содержит широкий ассортимент кондитерских изделий от различных фабрик для людей с разными моделями пищевого поведения.

- Содержит информацию об энергетической и пищевой ценности продуктов.
- Имеется форма для поиска товара при помощи фильтров.
- Содержит много дополнительной информации о фабриках, в том числе и где приобрести продукцию.

И в свою очередь следующим рядом недостатков:

- На сайте отсутствуют комментарии пользователей для конкретных товаров.
- Нет возможности добавлять кондитерские изделия в список избранных товаров.
- Если пользователь находится на карточке товара, то для возврата к каталогу ему приходится проделывать большое количество действий, так как происходит автоматический переход на главную страницу, что снижает просматриваемость сайта.

2.2.2 Конфил

В настоящее время АОр "НП "Конфил" является крупнейшим в Нижнем Поволжье кондитерским предприятием и входит в число двадцати крупнейших кондитерских фабрик России. Годовой объем продаж предприятия составляет около 10000 тонн при ассортименте более 300 наименований кондитерских изделий. Интерфейс приложения представлен на Рисунке 2.

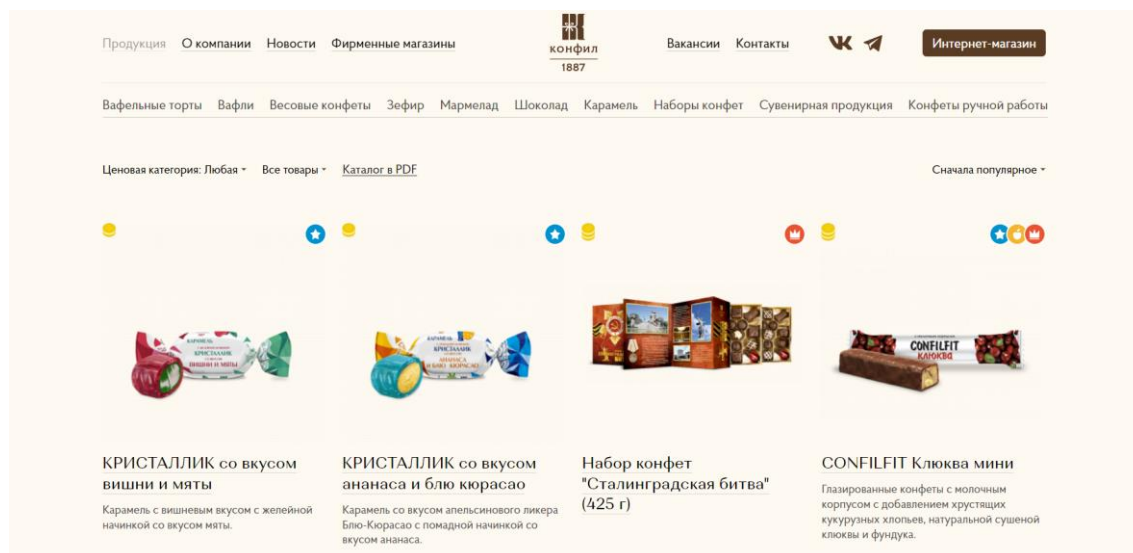


Рисунок 2 - Внешний вид каталога «Конфил»

Каталог «Конфил» обладает следующим рядом преимуществ:

- Содержит широкий ассортимент кондитерских изделий.
- Содержит подробную информацию об энергетической и пищевой ценности продуктов.
- Содержит много дополнительной информации о фабрике, в том числе и где приобрести продукцию.
- Есть фильтр разделения товара по ценовой категории.

И в свою очередь следующим рядом недостатков:

- Отсутствует строка поиска конкретных товаров.
- На сайте отсутствуют комментарии пользователей для конкретных товаров.
- Нет возможности добавлять кондитерские изделия в список избранных товаров.
- Продукция не предусмотрена для людей с разными моделями пищевого поведения.

2.2.3 Славянка

Группа компаний «Славянка» – крупный кондитерский холдинг с большим собственным производством, который входит в топ-50 мирового кондитерского рейтинга, и топ-500 в рейтинге предприятий России. Интерфейс приложения представлен на Рисунке 3.

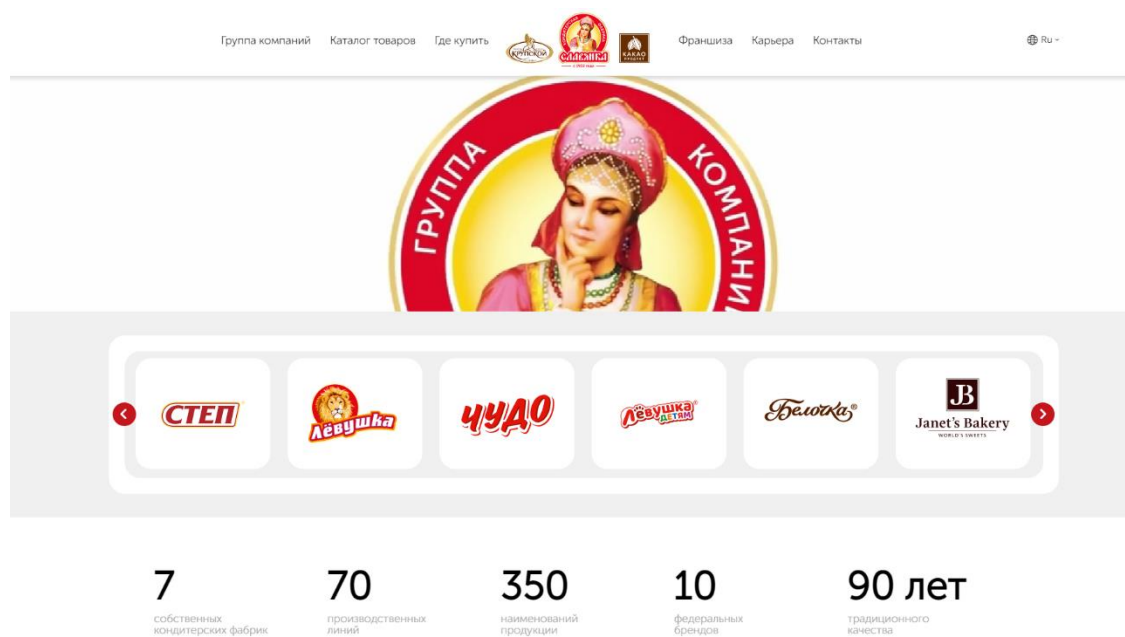


Рисунок 3 - Внешний вид каталога «Славянка»

Каталог «Славянка» обладает следующим рядом преимуществ:

- Содержит широкий ассортимент кондитерских изделий от различных фабрик для людей с разными моделями пищевого поведения.
- Содержит информацию об энергетической ценности продуктов.
- Имеется строка поиска.
- Содержит много дополнительной информации о фабриках, в том числе и где приобрести продукцию.

И в свою очередь следующим рядом недостатков:

- Не содержит информацию о пищевой ценности продуктов.
- На сайте отсутствуют комментарии пользователей для конкретных товаров.
- Нет возможности добавлять кондитерские изделия в список избранных товаров.

2.3 Диаграммы, иллюстрирующие работу системы

2.3.1 Диаграмма прецендентов (Use case)

Диаграмма прецендентов (Use case) представлена для трех типов акторов: неавторизованного пользователя, авторизованного пользователя и администратора. У каждого из них своя модель поведения, которую можно проследить на Рисунках 4-6.

Неавторизованный пользователь может:

- Зарегистрироваться в системе.
- Авторизоваться в системе.
- Осуществлять поиск товаров по наименованию.
- Просматривать категории товаров каталога.
- Просматривать товары по выбранной категории.
- Просматривать комментарии по выбранному товару.

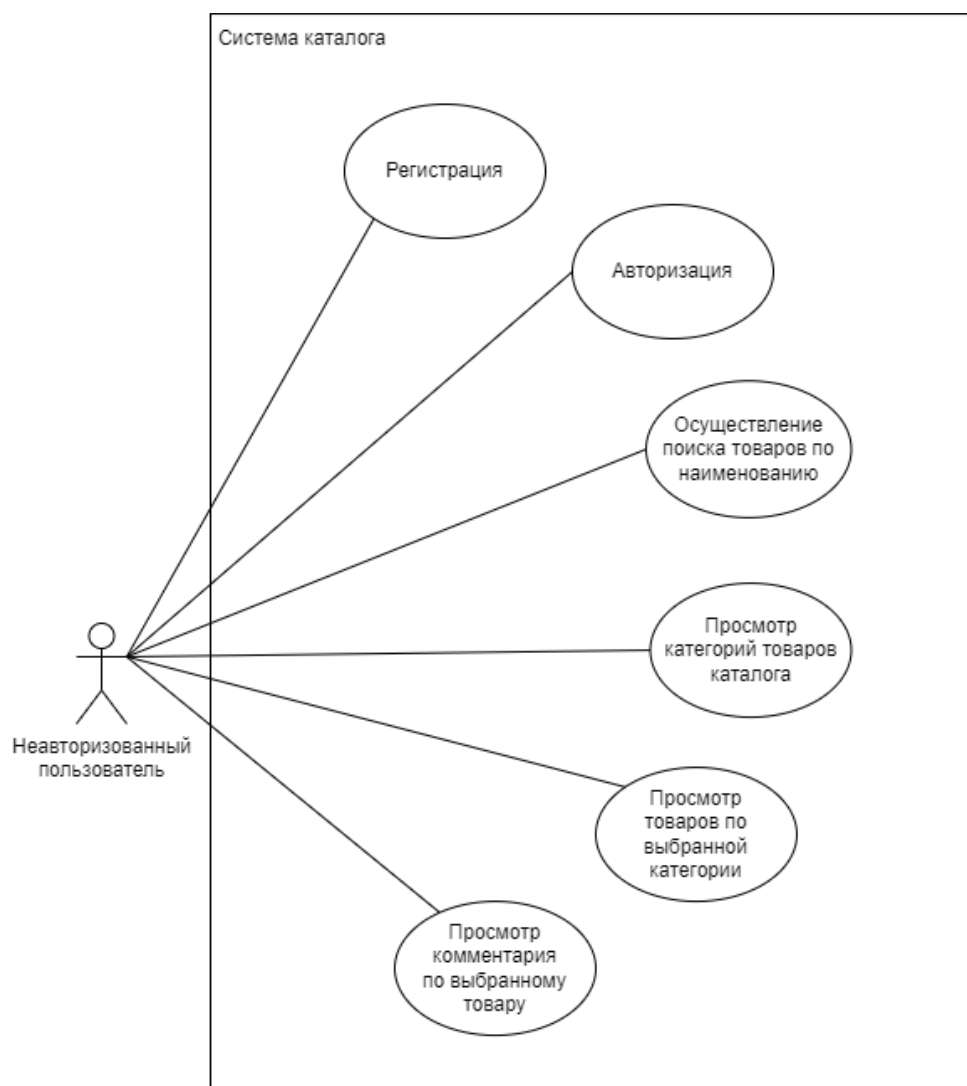


Рисунок 4 - Диаграмма прецедентов (Use case) для неавторизованного пользователя

Авторизованный пользователь помимо функций, доступных неавторизованному пользователю, может:

- Взаимодействовать со списком избранного: просматривать, добавлять и удалять товары.
- Взаимодействовать с комментариями, где помимо просмотра становятся доступны функции добавления и удаления комментариев.
- Осуществлять выход из системы.



Рисунок 5 - Диаграмма прецедентов (Use case) для авторизованного пользователя

Также существуют следующие функции, доступные администратору:

- Управлять информацией о товарах: добавление, удаление и редактирование.
- Управлять информацией о категориях: добавление, удаление и редактирование.
- Управлять информацией о пользователях: добавление, удаление, редактирование, просмотр данных и списка избранного.

— Управлять информацией о комментариях: просмотр, добавление, удаление и редактирование.

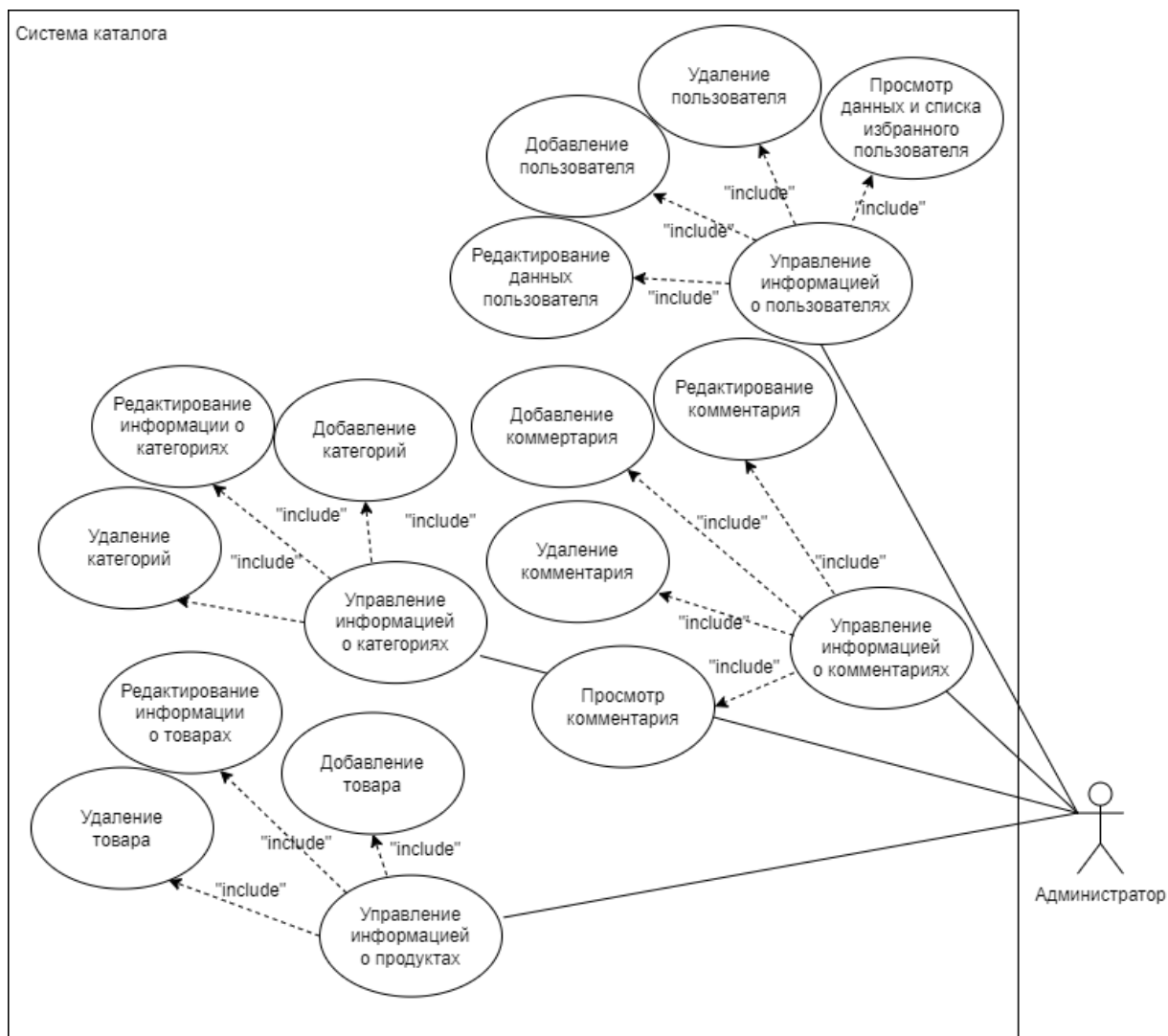


Рисунок 6 - Диаграмма прецедентов (Use case) для администратора

2.3.2 Диаграмма последовательности (Sequence diagram)

Существует также диаграмма последовательностей (Рисунок 7), на которой для некоторого набора объектов на единой временной оси показан жизненный цикл объекта и взаимодействие актеров информационной системы в рамках прецедента [3]. Участником данной системы является пользователь, а объектами – клиент, сервер и база данных.

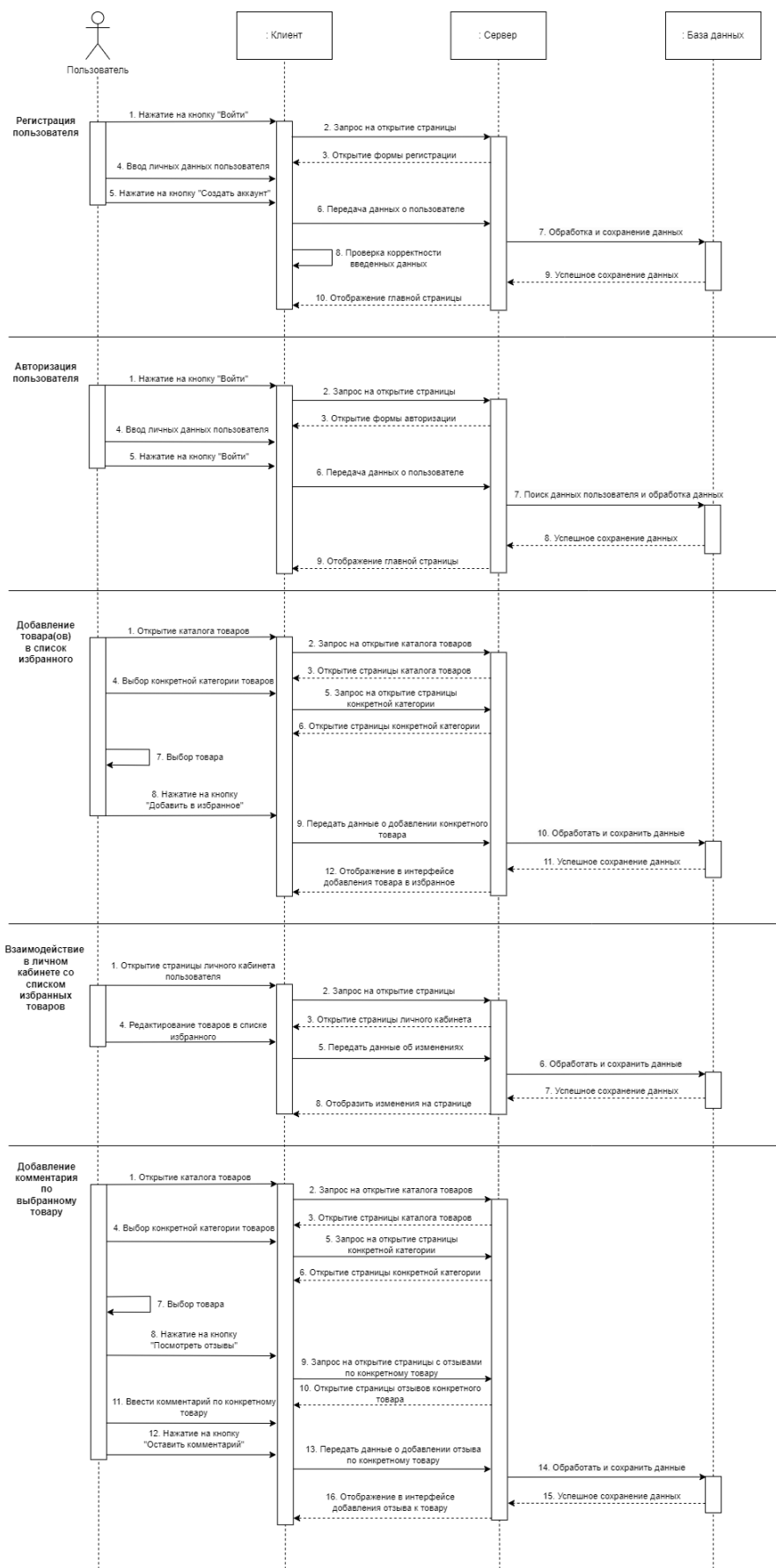


Рисунок 7 - Диаграмма последовательности

2.3.3 Диаграмма состояний (Statechart diagram)

Диаграмма состояний (Рисунок 8) отражает внутренние состояния объекта в течение его жизненного цикла от момента создания до разрушения [3]. На данной диаграмме рассмотрены состояния от момента входа в систему до полного выхода из нее.

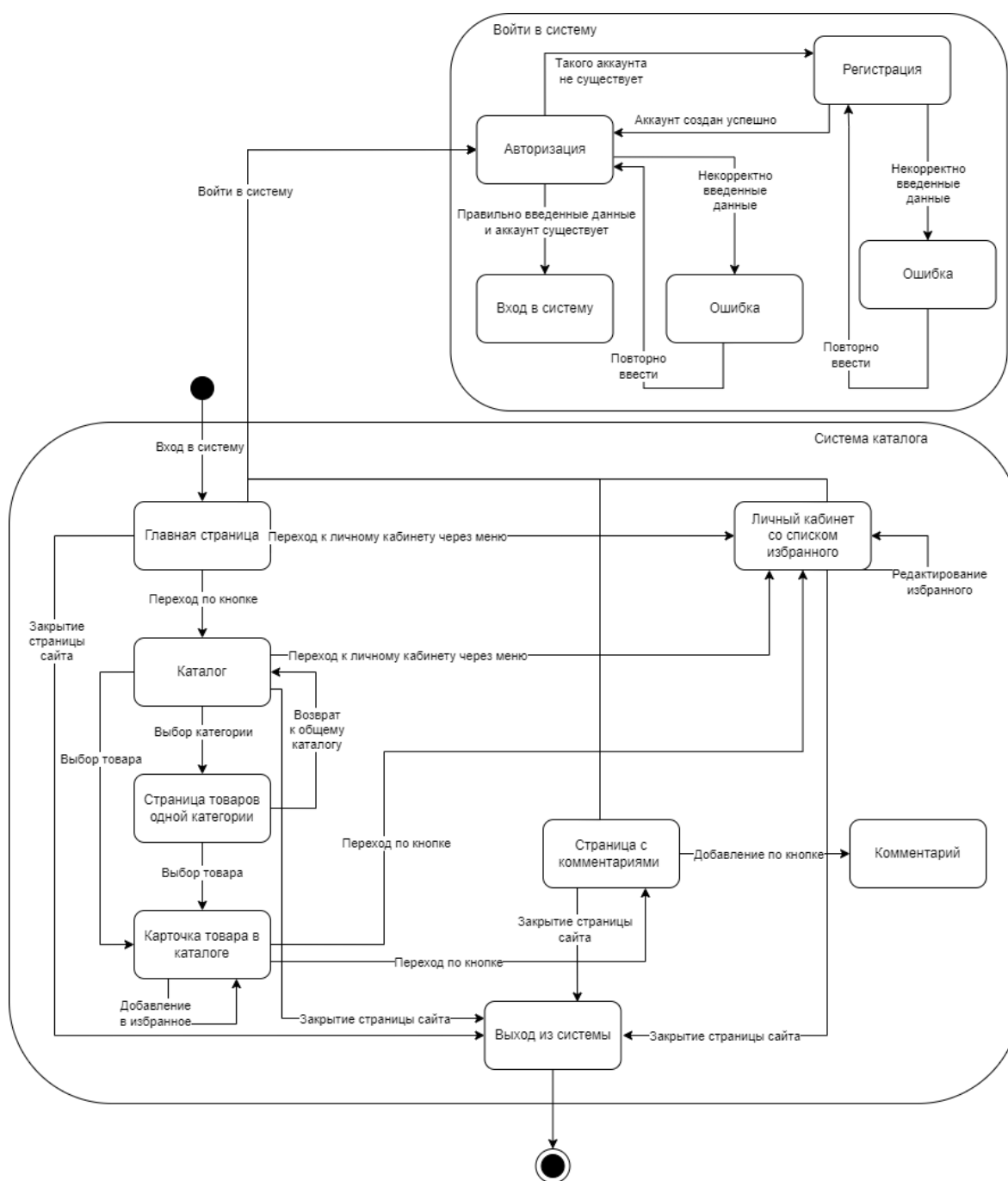


Рисунок 8 - Диаграмма состояний

2.3.4 Диаграмма деятельности (Activity diagram)

Диаграмма деятельности (Рисунок 9) представляет собой диаграмму, на которой показаны действия, состояния которых описаны на диаграмме состояний. Она описывает действия системы или людей, выполняющих действия, и последовательный поток этих действий [3].

В данном случае рассмотрен путь действий пользователя.

Диаграмма показывает, что пользователь, находясь в неавторизованной зоне системы не может заходить на свой профиль, добавлять товары в избранное и комментировать продукцию.

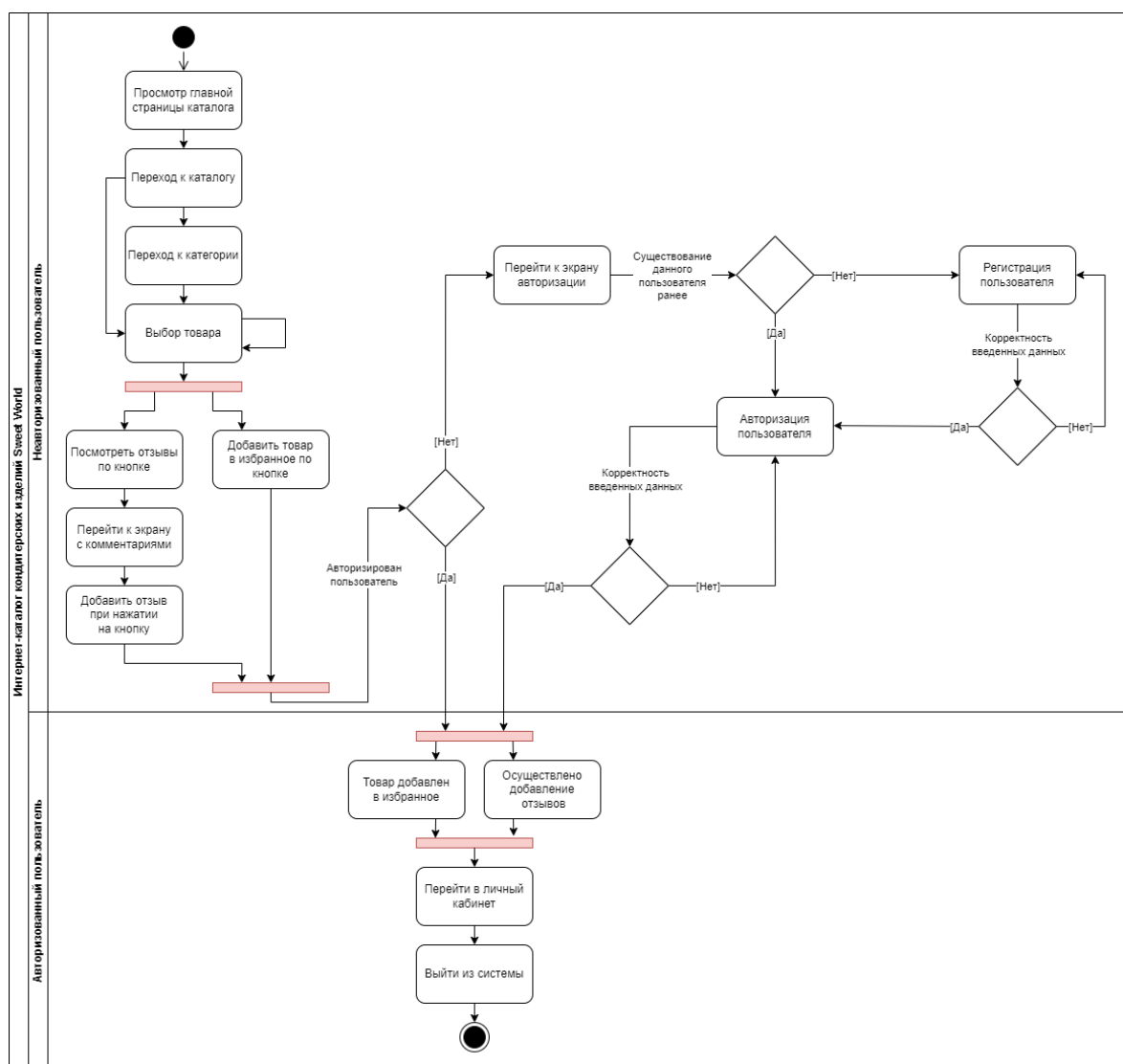


Рисунок 9 - Диаграмма деятельности (активности)

2.3.5 Диаграмма классов (Class diagram)

Диаграмма классов (Рисунок 10) демонстрирует общую структуру иерархии классов системы, их коопераций, атрибутов, методов, интерфейсов и взаимосвязей между ними. В данной системе рассмотрены следующие классы:

- Класс «Пользователь».
- Класс «Продукт».
- Класс «Категории продуктов».
- Класс «Комментарии».

У каждого из классов существуют свои атрибуты.

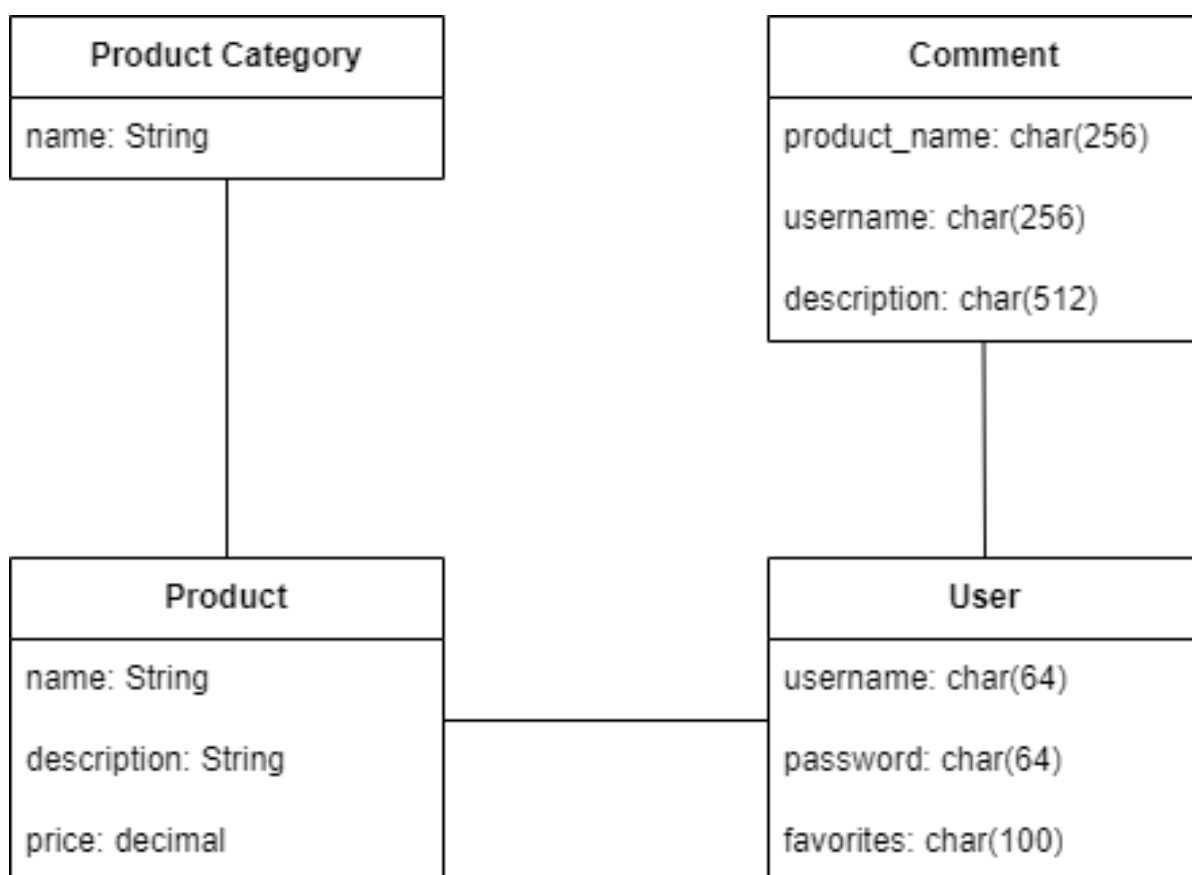


Рисунок 10 - Диаграмма классов

2.3.6 Диаграмма объектов (Object diagram)

По подобию диаграммы классов была выполнена диаграмма объектов. (Рисунок 11).

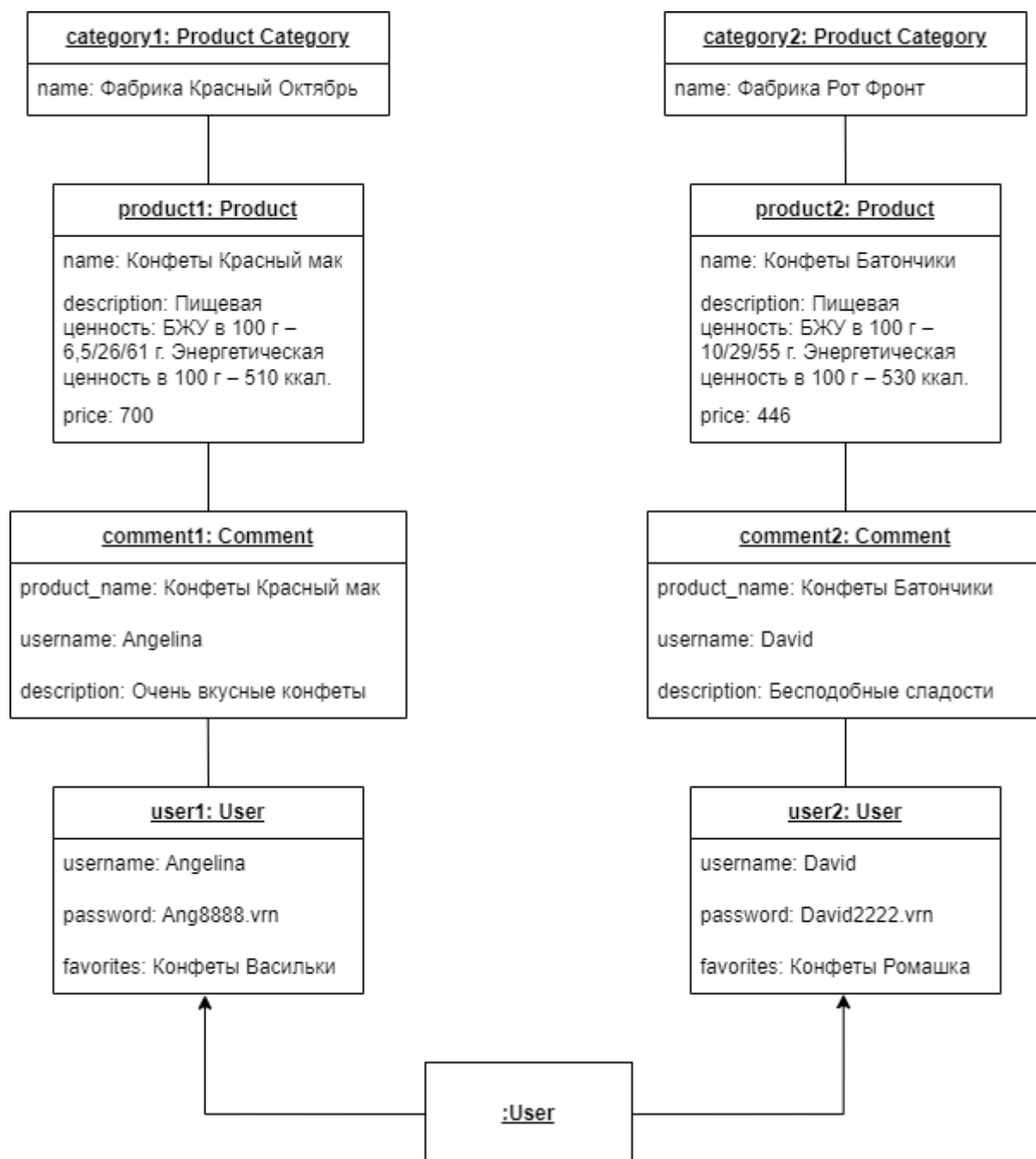


Рисунок 11 - Диаграмма объектов

2.3.7 Диаграмма сотрудничества (Collaboration diagram)

Диаграмма сотрудничества (Рисунки 12-15) — это вид диаграммы взаимодействия, в котором основное внимание сосредоточено на структуре взаимосвязей объектов, принимающих и отправляющих сообщения [4].

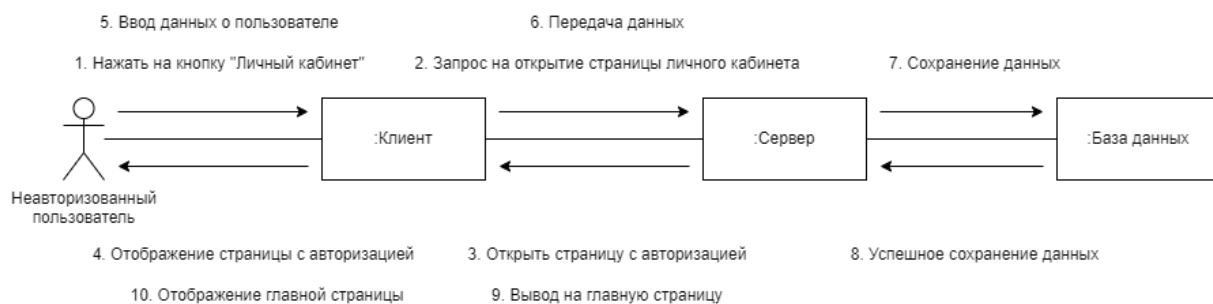


Рисунок 12 - Диаграмма сотрудничества при авторизации

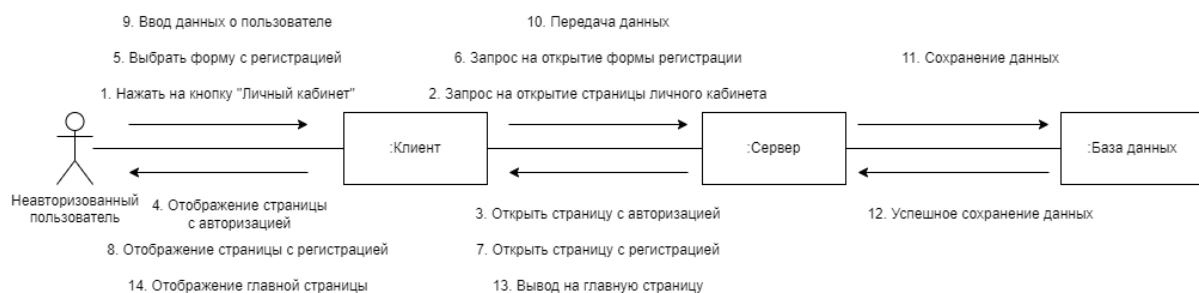


Рисунок 13 - Диаграмма сотрудничества при регистрации

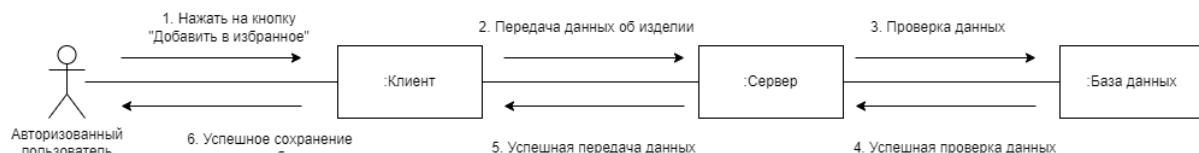


Рисунок 14 - Диаграмма сотрудничества при добавлении в избранное

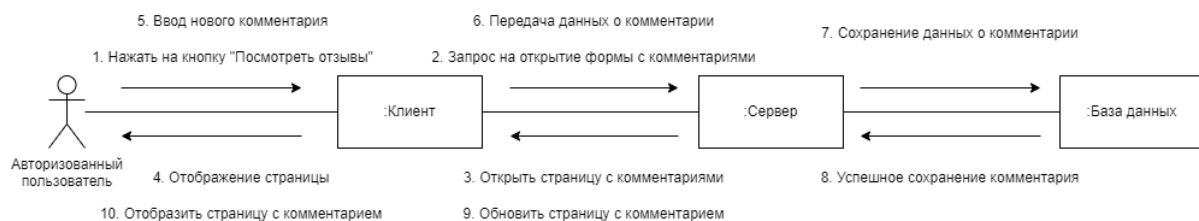


Рисунок 15 - Диаграмма сотрудничества при создании комментария

2.3.8 Диаграмма развертывания (Deployment diagram)

Диаграмма развертывания (Рисунок 16) предназначена для представления общей конфигурации или топологии распределенной программной системы [3].

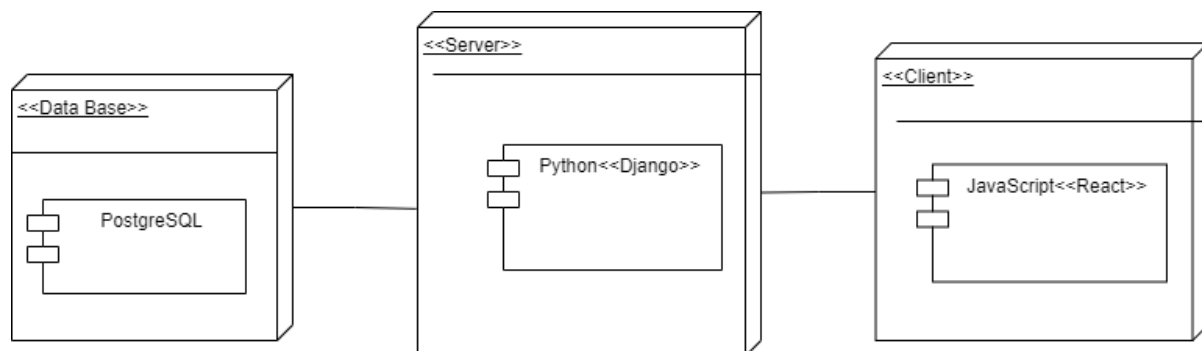


Рисунок 16 - Диаграмма развертывания

2.3.9 Диаграмма IDEF0

IDEF0 используется для создания функциональной модели, отображающей структуру и функции системы, а также потоки информации и материальные объекты, связывающие эти функции.

На Рисунке 17 представлена контекстная диаграмма системы. На вход системе поступает пользователь. Работу системы регулирует законодательство РФ. Как ресурсы, необходимые для работы системы, в неё поступают администратор и сайт. На выходе системы мы имеем удовлетворённого пользователя, список комментариев и избранного. Далее представлена декомпозиция диаграммы по уровням (Рисунки 18-20).

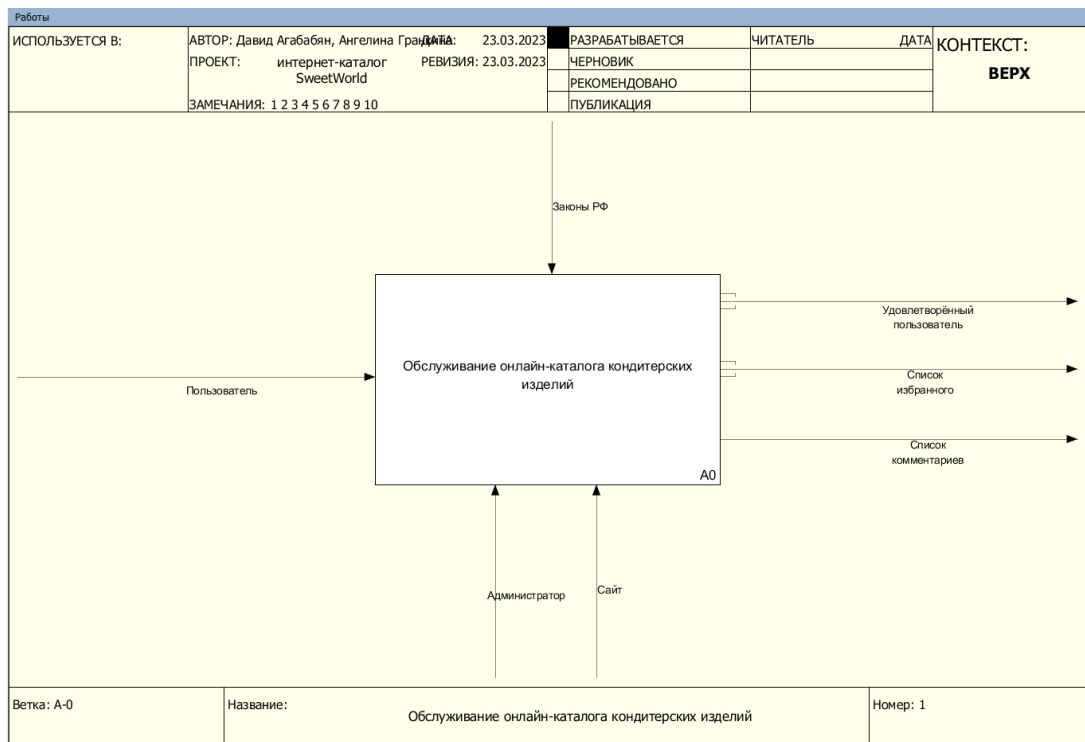


Рисунок 17 - Контекстная диаграмма системы

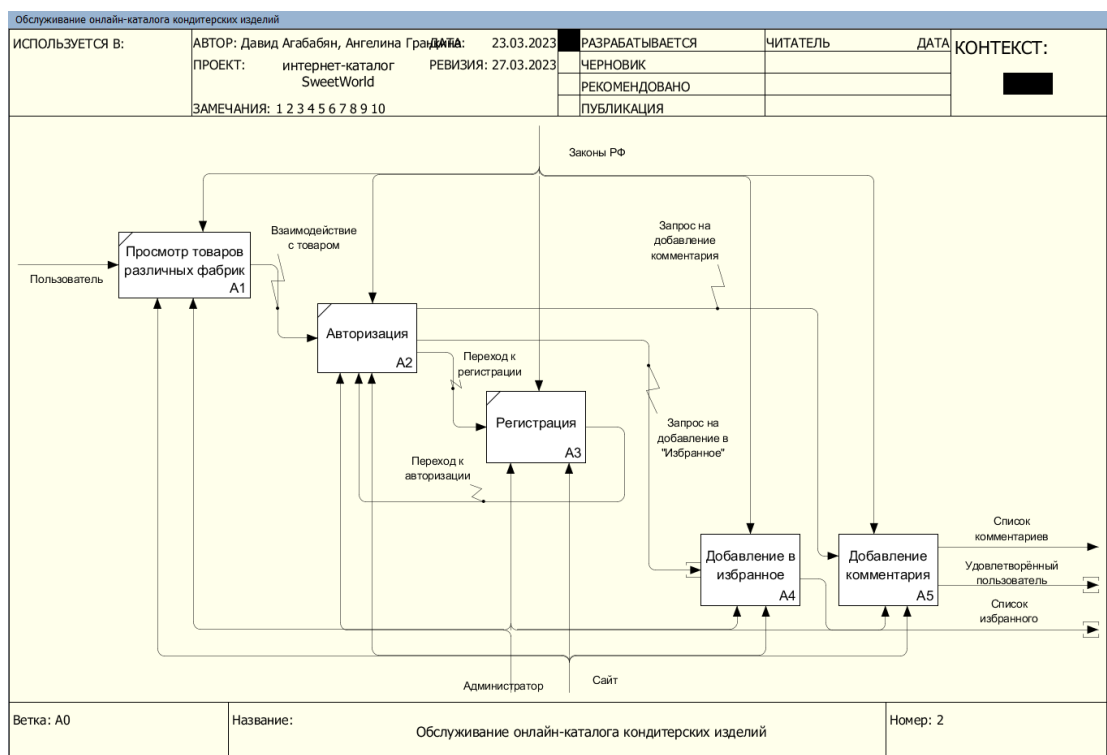


Рисунок 18 - Декомпозиция обслуживания онлайн-каталога кондитерских изделий

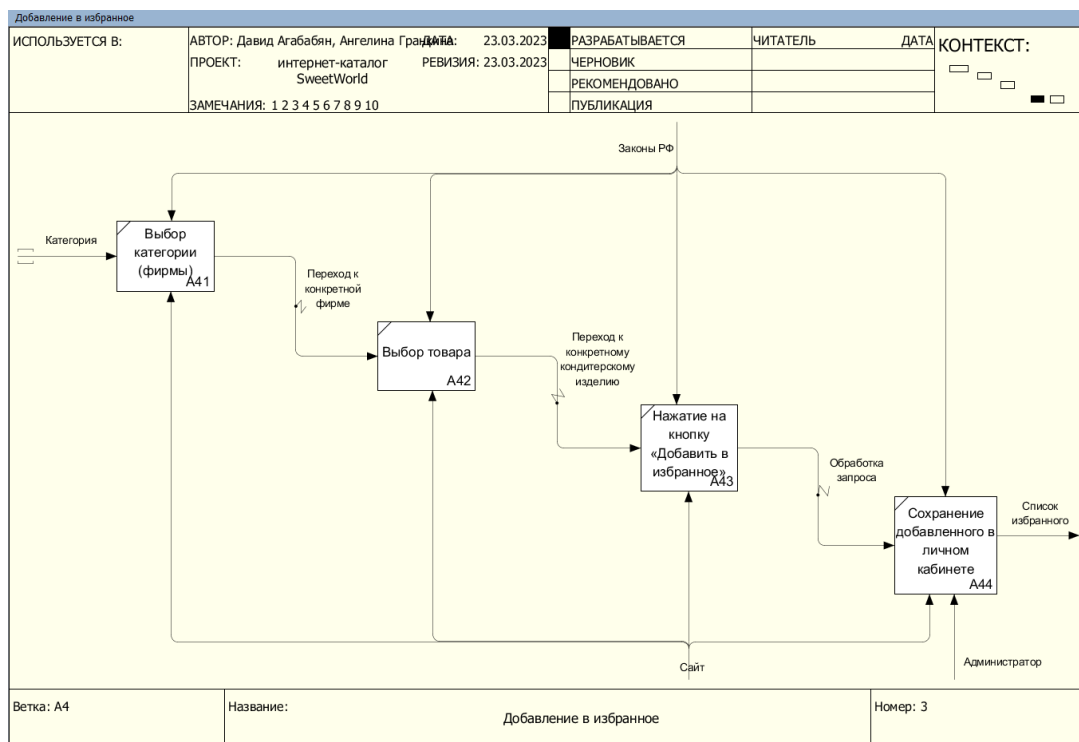


Рисунок 19 - Декомпозиция добавления в избранное

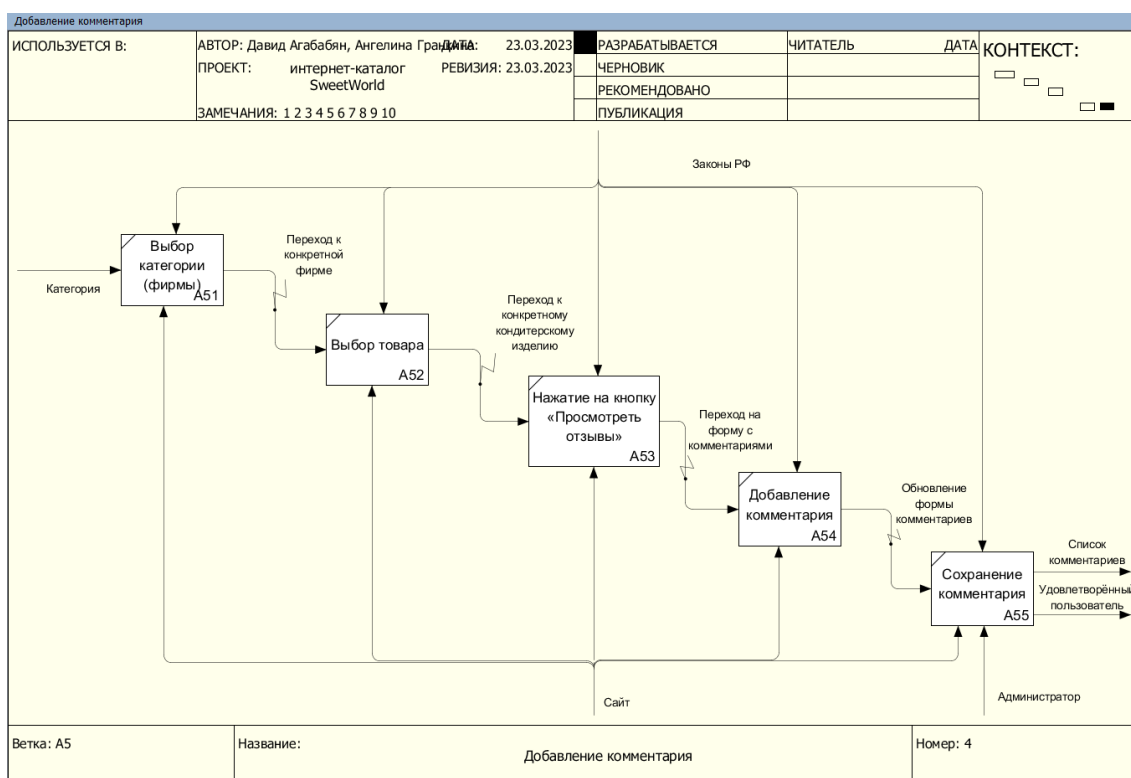


Рисунок 20 - Декомпозиция добавления комментария

2.3.10 ER-диаграмма

ER-диаграмма — это графическое представление модели данных, которая используется для описания концептуальной структуры базы данных. В такой диаграмме есть сущности, которые представляют объекты, с которыми работает система, и связи между сущностями, описывающие их взаимодействия. ER-диаграмма помогает наглядно описать структуру базы данных и увидеть связи между ее элементами (Рисунок 21).

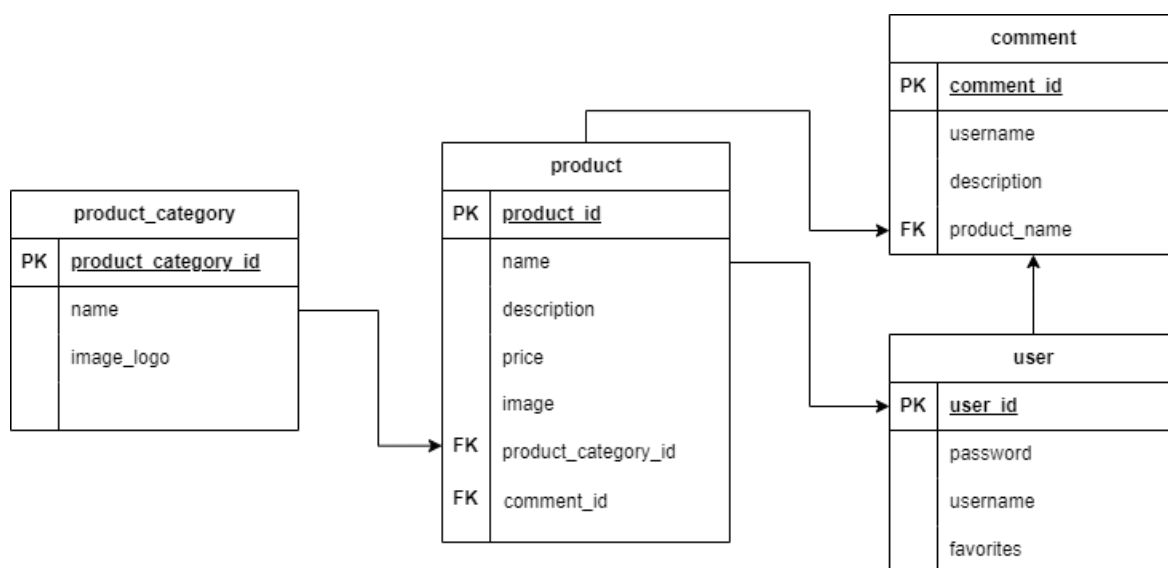


Рисунок 21 - ER-диаграмма

3 Реализация

3.1 Средства реализации

3.1.1 Средства реализации серверной части приложения

Для разработки серверной (backend) части приложения был выбран следующий стек технологий:

- Django — это высокоуровневый веб-фреймворк на языке Python. Он предоставляет набор инструментов и библиотек для упрощения и автоматизации различных аспектов веб-разработки, таких как работа с базами данных, обработка форм, работа с шаблонами, управление сессиями пользователей и многие другие функции. Он является открытым и бесплатным инструментом, доступным каждому разработчику.
- PostgreSQL это объектно-реляционная система управления базами данных (СУБД), которая предоставляет мощные средства для хранения, организации и манипулирования данными. PostgreSQL является свободным и открытым программным обеспечением и использует SQL для работы с данными. Он предлагает расширяемость, высокую надежность, многофункциональность, поддержку геопространственных данных, JSON-данных и многое другое. PostgreSQL активно развивается и используется в крупных проектах по всему миру.
- DB Browser for SQLite — это пользовательский интерфейс (GUI) для работы с базами данных SQLite. SQLite является легковесной реляционной базой данных, часто используемой в мобильных приложениях, десктопных приложениях и веб-приложениях.

— Swagger — инструмент для документирования и тестирования API. Он позволяет создавать интерактивную документацию для вебсервисов, что упрощает их использование и интеграцию. Swagger автоматически генерирует документацию на основе аннотаций и комментариев в коде, что позволяет разработчикам сосредоточиться на написании логики приложения, а не на создании и поддержке документации. Благодаря Swagger, разработчики могут изучить доступные параметры, модели данных и примеры запросов и ответов.

3.1.2 Средства реализации клиентской части приложения

Для разработки клиентской (frontend) части приложения был выбран следующий стек технологий:

- JavaScript — это один из наиболее популярных языков программирования, который используется для разработки веб-приложений, игр, мобильных приложений и других приложений. JavaScript является интерпретируемым языком скриптов, то есть он выполняется в среде браузера, что позволяет создавать динамические и интерактивные веб-сайты, а также управлять внешним поведением страницы. Помимо этого, он обладает следующими преимуществами: широкая поддержка, гибкость, доступ к различным библиотекам и фреймворкам.
- React.js — это open-source JavaScript библиотека для построения пользовательских интерфейсов, созданная Facebook. React позволяет разрабатывать компоненты UI в виде функций или классов, которые могут быть многократно использованы, и которые просты в поддержке. React используется для создания одностраничных приложений (SPA), мобильных приложений,

административных дашбордов и других веб-приложений. React реализует виртуальный DOM, который дает библиотеке многие преимущества перед другими фреймворками и библиотеками для разработки пользовательских интерфейсов, такие как скорость, универсальность, простота и удобство тестирования.

— CSS (Cascading Style Sheets) — это язык стилей, используемый для задания внешнего вида веб-страниц. CSS позволяет разработчикам отделить визуальное представление веб-сайта от содержания, тем самым обеспечивая большую гибкость и управляемость веб-ресурсов. С помощью CSS можно задавать шрифты, цвета, расположение, размеры элементов, оформление фона и др. CSS является основным инструментом для создания визуального дизайна веб-сайтов и позволяет создавать уникальные дизайны и согласовывать внешний вид контента на всем сайте.

— HTML (HyperText Markup Language) — это язык разметки, используемый для создания веб-страниц. С помощью HTML разработчики определяют структуру и содержание веб-страниц, позволяя браузеру правильно интерпретировать и отображать контент для пользователей. HTML использует теги для форматирования текста, вставки изображений, оформления списков, аудио и видео, ссылок и многого другого. Он обладает следующими преимуществами: читаемость для разработчиков, возможность создания структурированного контента с использованием семантических элементов, доступность и адаптивности для различных устройств и браузеров. HTML является основным языком для создания веб-страниц и работает в сочетании с CSS для определения внешнего вида и JavaScript для добавления интерактивности.

3.2 Реализация серверной (backend) части приложения

Серверная (backend) часть приложения была написана на языке Python с использованием фреймворка Django. Структура проекта представляет собой корневую папку `tp_shop` и дополнительные — `comments`, `products`, `productscategory` и `users` (Рисунок 22).

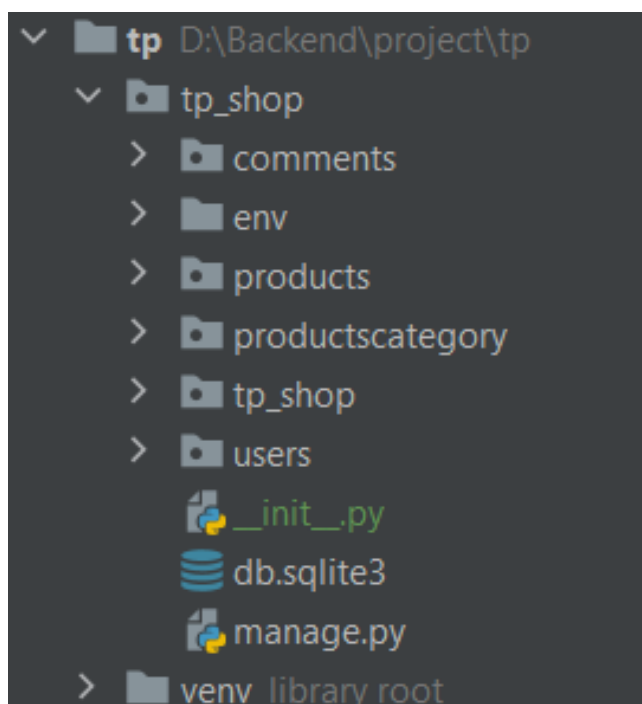


Рисунок 22 - Структура проекта

В корневой папке (Рисунок 23) особое внимание стоит обратить на файл `settings.py`, который содержит настройки проекта. Он содержит множество параметров для настройки работоспособности приложения.

- `DEBUG` - параметр, отвечающий за включение/выключение режима отладки.
- `SECRET_KEY` - секретный ключ приложения, который применяется для генерации токенов, а также для шифрования паролей пользователей.

- DATABASES - настройки подключения к базе данных, в которой хранятся данные приложения.
- INSTALLED_APPS - список приложений, установленных в проекте. Здесь указываются все приложения, которые будут использованы в проекте.
- MIDDLEWARE - список всех промежуточных компонентов, используемых в Django для обработки запросов и ответов между сервером и клиентом.
- TEMPLATES - настройки для генерации HTML-шаблонов, используемых для визуализации данных пользователя в БД.
- AUTH_PASSWORD_VALIDATORS - список компонентов, используемых для проверки безопасности паролей пользователей.

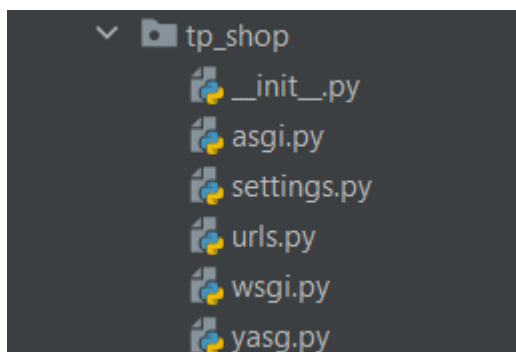


Рисунок 23 - Структура корневой папки проекта

При этом фреймворк Django подразумевает разделение приложения на несколько основных компонентов (Рисунок 24).

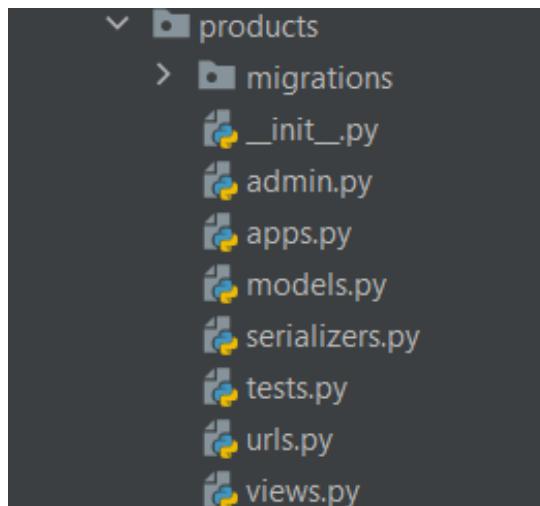


Рисунок 24 - Основные компоненты Django на примере product

- Инициализация (`__init__.py`) — это специальный файл в Python, определяющий пакет. В Django он используется для определения пакета для приложения.
- Административная панель (`admin.py`) — это административная панель Django, которая обеспечивает автоматическую генерацию форм и предоставляет удобный интерфейс для работы с данными.
- Конфигурационный файл (`apps.py`) — это Python файл, который определяет основные настройки для приложения. Он содержит метаданные, относящиеся к данному приложению, и позволяет настроить приложение в соответствии с требованиями и лучше контролировать его работу.
- Модели данных (`models.py`) — это классы Python, которые определяют структуру базы данных и могут быть использованы для создания или обновления схемы базы данных. Модели могут содержать поля для хранения данных (текстовые, числовые, даты, файлы и др.), а также методы для работы с этими данными.
- Сериализация (`serializers.py`) — файл, обеспечивающий сериализацию и десериализацию данных, передаваемых через

приложение. Он используется в Django для работы с данными, передаваемыми через HTTP в API-модулях. Файл содержит классы сериализаторов, которые определяют, какие поля модели должны быть преобразованы в JSON, XML и т.д. Они также могут обеспечивать валидацию данных во время десериализации.

— Тесты (`test.py`) — это файл в приложении, который содержит модульные тесты для этого приложения. Эти тесты используются, чтобы убедиться, что функциональные возможности приложения работают должным образом, и выявить любые ошибки или ошибки до того, как приложение будет развернуто в рабочей среде.

— URL-адресация (`urls.py`) — это механизм маршрутизации запросов на определенные представления. Django использует файл `urls.py` для определения соответствующей представлению URL-адреса.

— Представления (`views.py`) — это функции Python, которые обрабатывают запросы от клиента и возвращают HTTP-ответы. Представления могут включать в себя логику приложения, обработку данных из модели, взаимодействие с другими системами.

Серверная часть приложения Django также может использовать множество дополнительных библиотек и компонентов.

База данных была развернута на виртуальной машине с помощью системы контейнеризации Docker. Такие данные как хост, порт, имя базы данных, имя пользователя и пароль задавались при помощи файла `docker-compose.yml`. Эти данные будут использоваться для настройки подключения к базе данных из приложения.

Сервер был развернут также на виртуальной машине с помощью системы контейнеризации Docker. Этот процесс был выполнен с

использования системы контроля версий Git. После загрузки потребовалось настроить переменные окружения, представляющие собой конфиденциальные настройки, такие как данные аутентификации для базы данных. Клиентская часть была загружена в другом Docker-контейнере.

Для описания спецификации API использовался Swagger [5]. Для того чтобы интегрировать его в проект, нужно было внести изменения в код, включая добавление необходимой зависимости (`drf_yasg`), добавить ее в `INSTALLED_APPS` и описать спецификации всех методов.

3.3 Реализация клиентской (frontend) части приложения

Клиентская часть приложения (frontend) разработана на языке JavaScript с использованием фреймворка React.js.

Одной из особенностей разработки приложения на React является декомпозиция приложения на независимые компоненты. Каждый компонент отвечает за определенный функциональный блок приложения и представляет собой функцию, которая возвращает HTML-код. Кроме того, компоненты могут содержать переменные и другие функции, что позволяет дополнительно упростить код и повысить его читабельность. Приложение загружается на страницу по мере необходимости, что уменьшает время загрузки страницы и повышает производительность приложения в целом.

Основной HTML файл называется `index.html`, именно на него загружаются компоненты, которые будут выведены в браузере. В нём вызывается `index.js`, в котором находится точка входа React-приложения, и вызывается основной компонент `App.js`, который будет меняться в зависимости от действий пользователя. Все остальные компоненты вызываются по мере необходимости.

Структура проекта на React включает в себя определенные основные элементы (Рисунок 25).

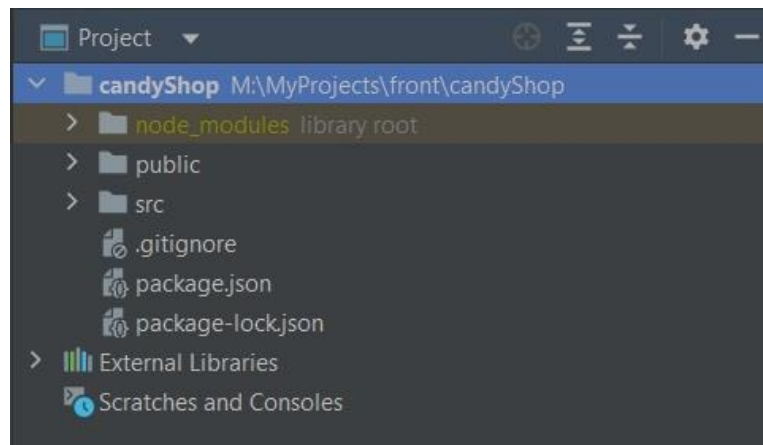


Рисунок 25 - Основные элементы React-проекта

- Основная папка проекта `src/`, содержащая исходный код.
- Папка `public/`, содержащая файлы, которые будут доступны публично.
- Файл `package.json`, содержащий информацию о проекте, а также список зависимостей для установки.
- Файл `package-lock.json` с информацией о текущей версии зависимостей.

В свою очередь в папке `src/` (Рисунок 26) хранятся следующие разделы приложения.

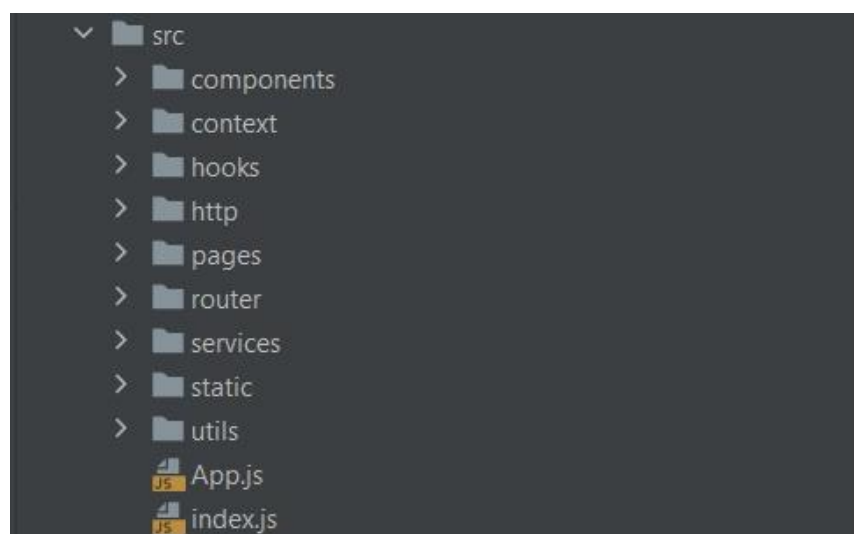


Рисунок 26 - Структура папки `src/` проекта

- Папка `components/`, содержащая все компоненты приложения.
- Папка `context/`, необходимая для хранения файлов, содержащих в себе контексты для глобализации данных и доступа к ним без передачи значений между компонентами, содержит в себе `AuthContext` для хранения данных о пользователе.
- Папка `hooks/`, содержащая в себе определенных `hooks`, использующиеся в приложении.
- Папка `http/`, содержащая в себе объект `axios` со стандартными настройками для обращения к REST API.
- Папка `pages/`, содержащая существующие страницы нашего приложения.
- Папка `router/`, хранящая в себе списки всех `routes`, используемых в приложении.
- Папка `services/`, совершающая обращение к серверу через API.
- Папка `static/`, содержащая таблицы стилей для компонентов и приложения в целом.
- Папка `utils/`, содержащая утилиты, которые используются в приложении.

Компоненты могут состоять из других компонентов. Поэтому в приложении помимо компонентов, описывающих конкретные страницы, есть компоненты, из которых состоят эти страницы. Компонентами приложения являются следующие разделы (Рисунок 27).

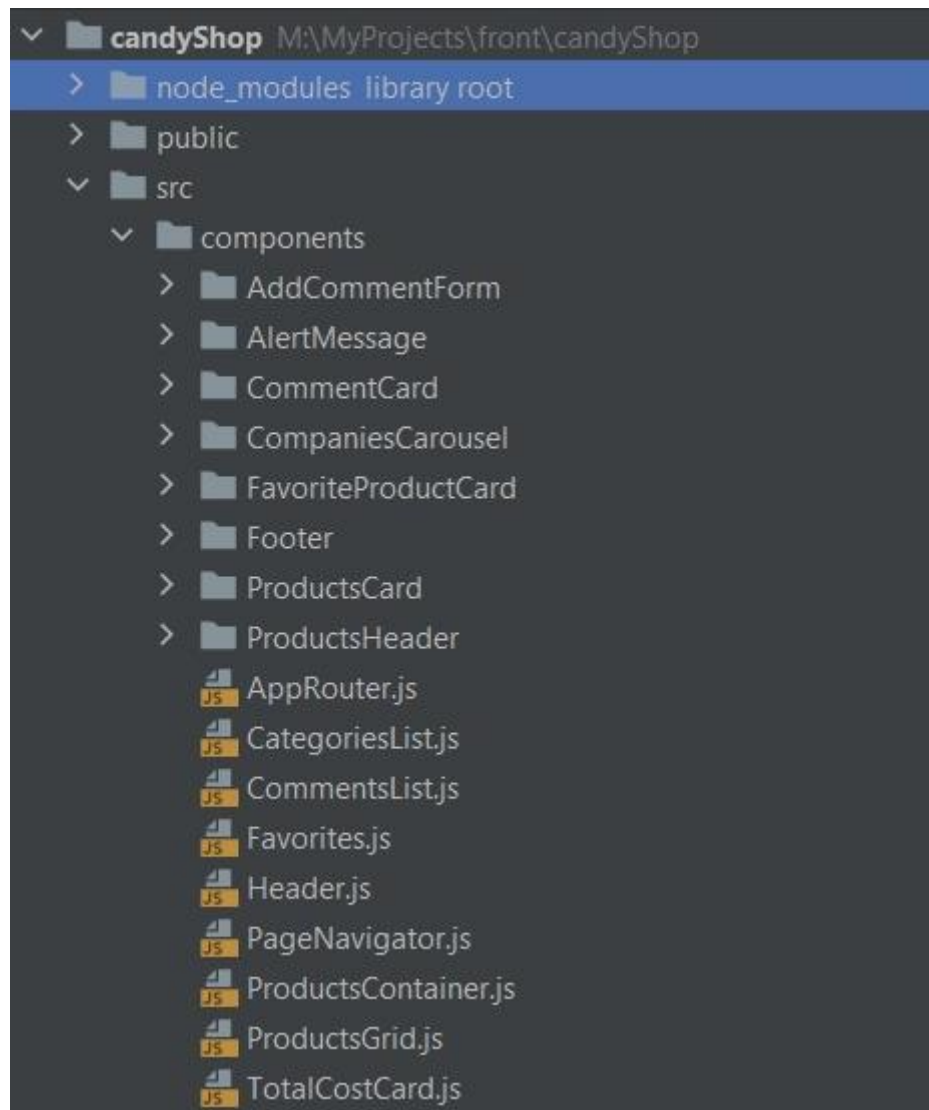


Рисунок 27 - Компоненты приложения

- Папка AddCommentForm – добавление комментария.
- Папка AlertMessage – сообщение при добавлении товара в список «Избранное».
- Папка CommentCard – комментарий.
- Папка CompaniesCarousel – карусель логотипов ведущих фабрик России.
- Папка FavoriteProductCard – товары в списке «Избранного».
- Папка Footer – подвал веб-приложения.

- Папка `ProductsCard` – карточка товара.
- Папка `ProductsHeader.js` – заголовок в каталоге.
- `AppRouter.js` — возвращает в себе объекты типа `Route`, хранящие в себе значения из папки `routes`.
- `CategoriesList.js` – список категорий (фабрик) каталога.
- `CommentsList.js` – список комментариев.
- `Favorites.js` – список избранных товаров.
- `Header.js` – шапка веб-приложения.
- `PageNavigator.js` – пагинация сайта.
- `ProductsContainer.js` — компонент каталога, возвращающий в себе компоненты, необходимые на странице каталога
- `ProductsGrid.js` — компонент, возвращающий карточки товара.
- `TotalCostCard.js` – итоговая стоимость в списке избранных товаров.

3.4 Навигация по приложению

3.4.1 Для неавторизованного пользователя

Первое, что видит пользователь — главный экран (Рисунок 28).

Далее у пользователя возникает выбор: он может войти в личный кабинет при помощи авторизации (Рисунок 29), если нет аккаунта – при помощи регистрации (Рисунок 30) или продолжить пользоваться сайтом в режиме неавторизованного пользователя, но с ограниченным функционалом (без возможности осуществлять добавление товаров в избранное и написания комментариев).

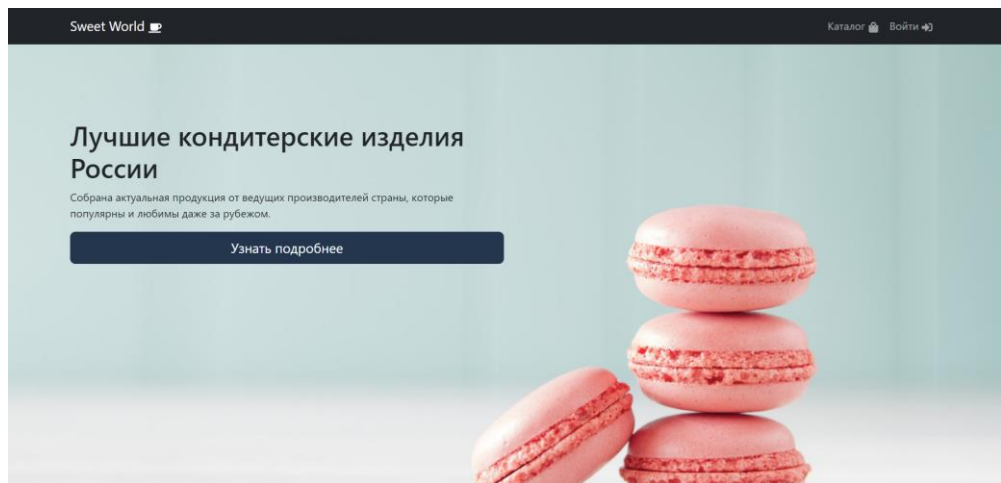


Рисунок 28 - Главная страница каталога

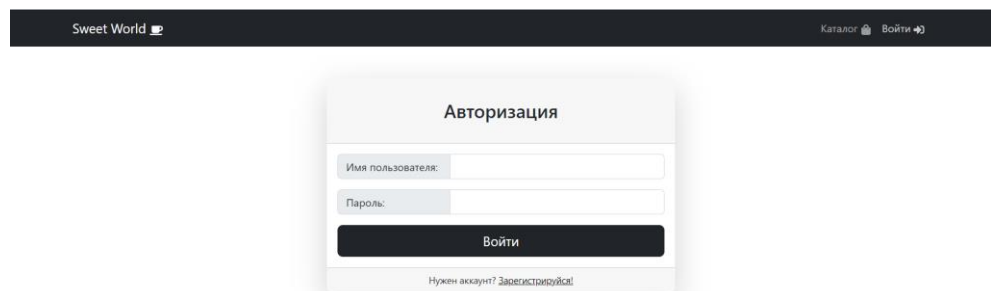


Рисунок 29 - Страница авторизации

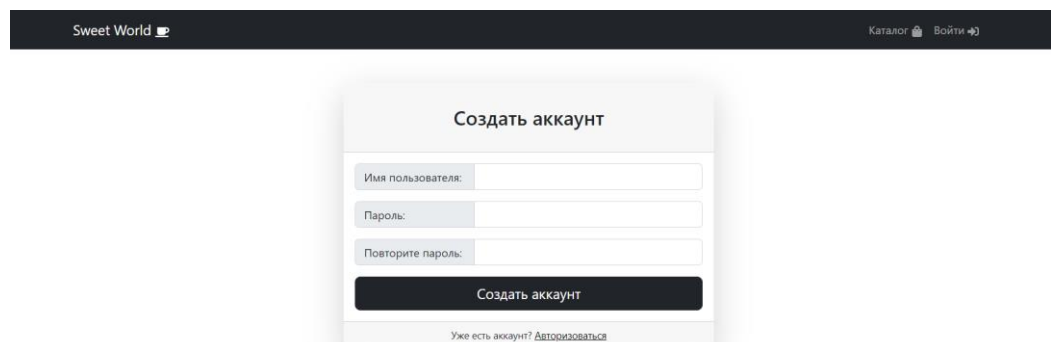


Рисунок 30 - Страница регистрации

Неавторизованному пользователю доступен просмотр каталога товаров, где можно осуществить поиск необходимого кондитерского изделия по наименованию (Рисунок 31).

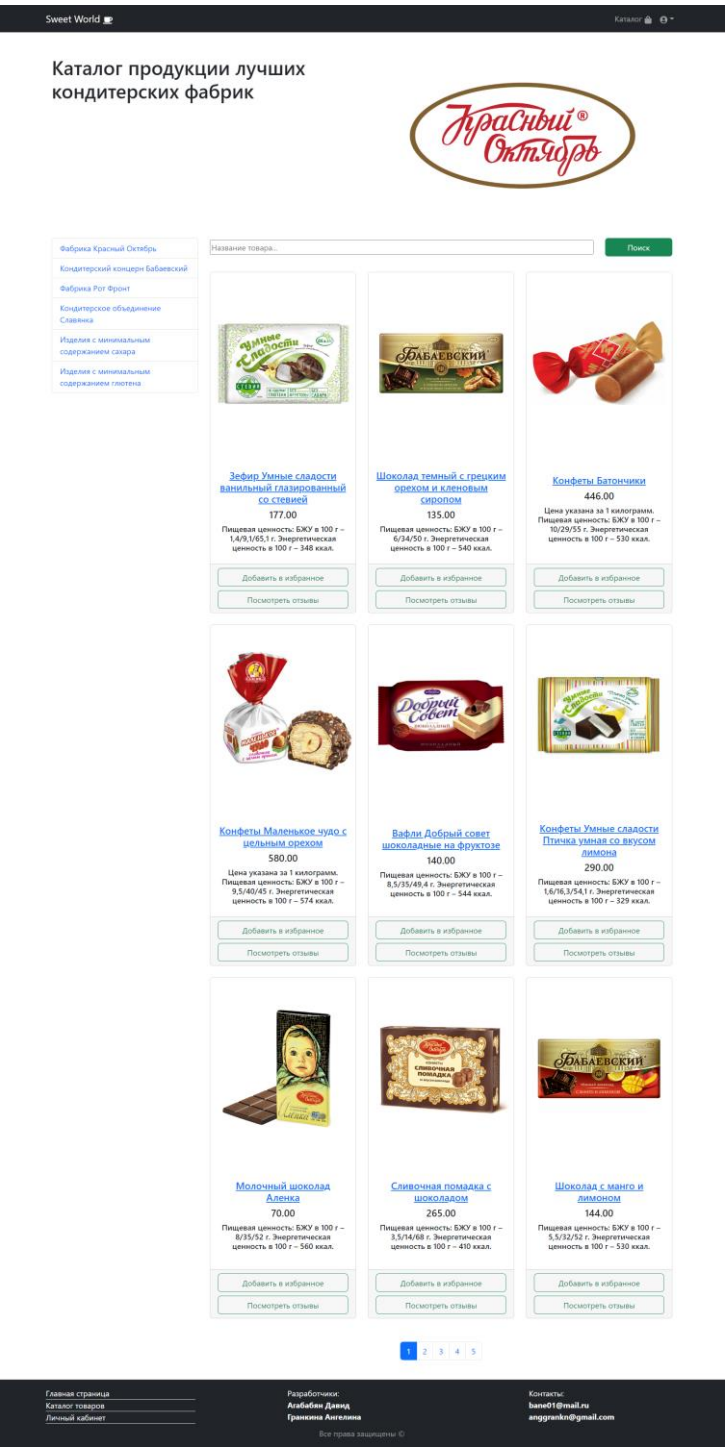



Рисунок 31 - Просмотр каталога и функция поиска

Помимо этого, можно переходить к конкретным категориям и смотреть определенного рода продукцию (Рисунок 32).

Sweet World

Каталог

Каталог продукции лучших кондитерских фабрик



Фабрика Красный Октябрь


Кондитерский концерн Бабаевский

Фабрика Рот Фронт

Кондитерское объединение Славянка

Изделия с минимальным содержанием сахара

Изделия с минимальным содержанием глютена




[Шоколад темный с грецким орехом и кленовым сиропом](#)

135.00

Пищевая ценность: БЖУ в 100 г – 6/34/50 г. Энергетическая ценность в 100 г – 540 ккал.

Добавить в избранное

Посмотреть отзывы




[Шоколад с манго и лимоном](#)

144.00

Пищевая ценность: БЖУ в 100 г – 5,5/32/52 г. Энергетическая ценность в 100 г – 530 ккал.

Добавить в избранное

Посмотреть отзывы




[Шоколад горький 75% какао элитный](#)

129.00

Пищевая ценность: БЖУ в 100 г – 10,8/38,5/37,1 г. Энергетическая ценность в 100 г – 545 ккал.

Добавить в избранное

Посмотреть отзывы




[Шоколад темный с грейпфрутовыми кусочками](#)

140.00

Пищевая ценность: БЖУ в 100 г – 7/34/49 г. Энергетическая ценность в 100 г – 540 ккал.

Добавить в избранное

Посмотреть отзывы



[Шоколад Люкс](#)

100.00

Пищевая ценность: БЖУ в 100 г – 5,4/36,3/51,8 г. Энергетическая ценность в 100 г – 549 ккал.

Добавить в избранное

Посмотреть отзывы

1

Рисунок 32 - Страница конкретной категории (фабрики) каталога

43

Для неавторизованного пользователя также доступен просмотр отзывов других пользователей (Рисунок 33).

Sweet World

Каталог Войти

Отзывы на товар Зефир Умные сладости ванильный глазированный со стевией

Добавить отзыв

Отзыв:

Оставить комментарий

Список отзывов:

Мишана на продукт Зефир Умные сладости ванильный глазированный со стевией

Классные конфеты

Рисунок 33 - Просмотр отзывов

3.4.2 Для авторизованного пользователя

После авторизации пользователю становится доступной возможность добавлять в избранное конкретный товар. Для этого на странице каталога товаров необходимо нажать на кнопку «Добавить в избранное», при этом повторное добавление обрабатывается ошибкой (Рисунки 34-35).

Фабрика Красный Октябрь

Кондитерский концерн Бабаевский

Фабрика Рот Фронт

Кондитерское объединение Славянка

Изделия с минимальным содержанием сахара

Изделия с минимальным содержанием глютена

Товар успешно добавлен в избранное

Зефир Умные сладости ванильный глазированный со стевией

177.00

Пищевая ценность: БЖУ в 100 г – 1,4/9,1/65,1 г. Энергетическая ценность в 100 г – 348 ккал.

Добавить в избранное

Посмотреть отзывы

Конфеты Умные сладости Птичка умная со вкусом лимона

290.00

Пищевая ценность: БЖУ в 100 г – 1,6/16,3/54,1 г. Энергетическая ценность в 100 г – 329 ккал.

Добавить в избранное

Посмотреть отзывы

Конфеты Умные сладости Птичка умная со вкусом крем-брюле

336.00

Пищевая ценность: БЖУ в 100 г – 1,8/18/55 г. Энергетическая ценность в 100 г – 329 ккал.

Добавить в избранное

Посмотреть отзывы

Рисунок 34 - Успешное добавление в избранное

44

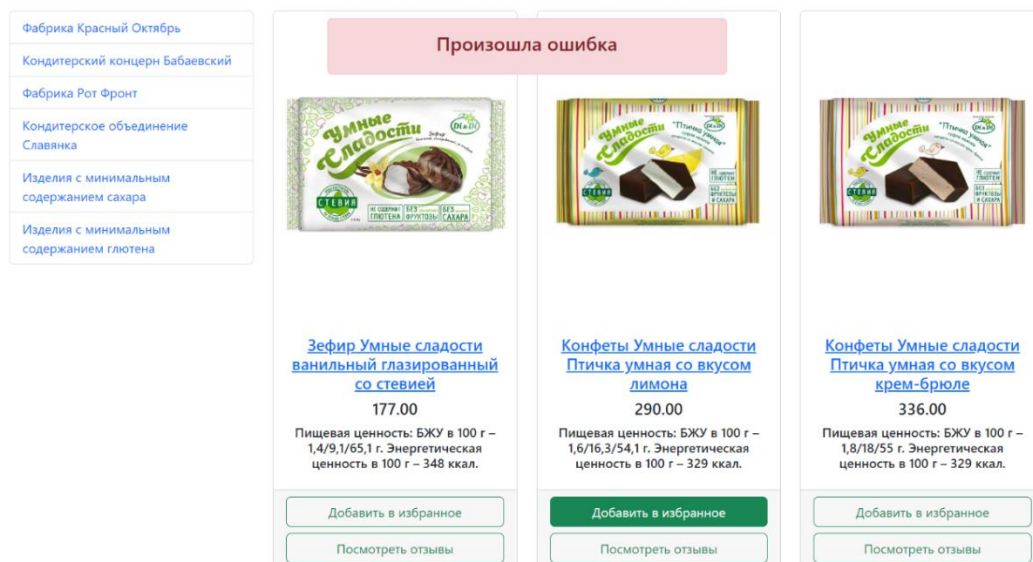


Рисунок 35 - Повторное добавление товара в избранное

Список избранных товаров, как и информацию о себе, введенную при регистрации, пользователь может посмотреть в своем личном кабинете (Рисунок 36).

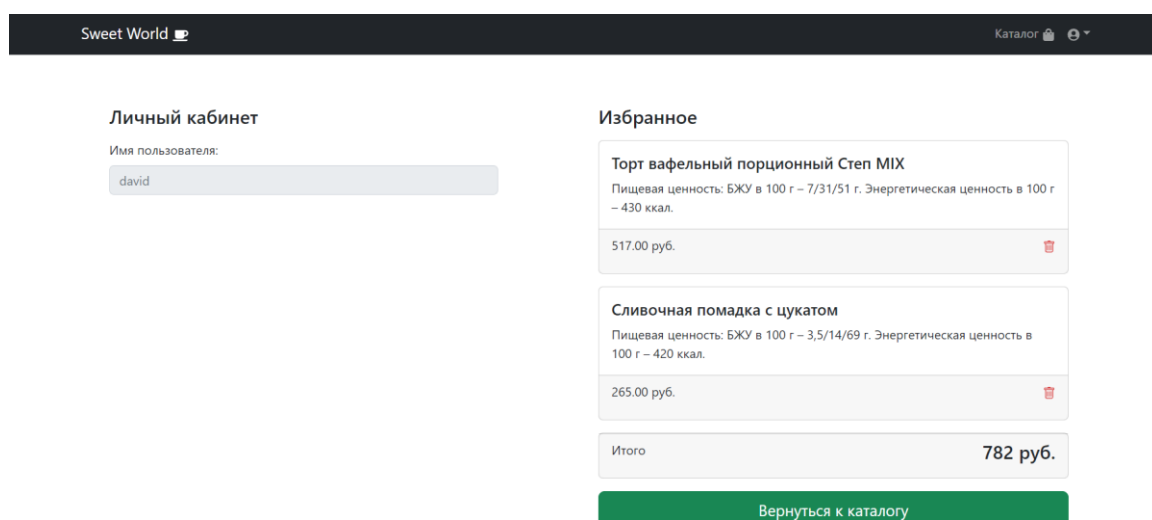


Рисунок 36 - Личный кабинет пользователя со списком избранных товаров

Также авторизованному пользователю доступна функция написания комментария. Для этого на странице каталога товаров необходимо нажать на кнопку «Посмотреть отзывы» (Рисунок 37), после нажатия на которую

происходит переход на экран с отзывами на конкретный товар (Рисунок 38), где необходимо ввести свое сообщение. На этом экране также выводится имя пользователя, пишущего отзыв, и название соответствующего товара.

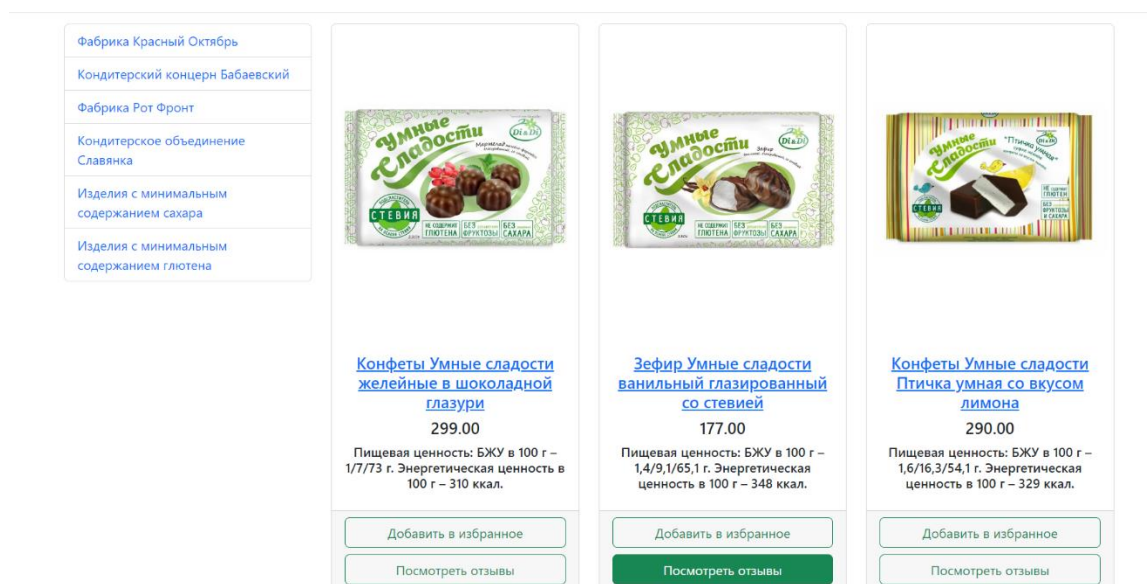


Рисунок 37 - Просмотр отзывов в каталоге

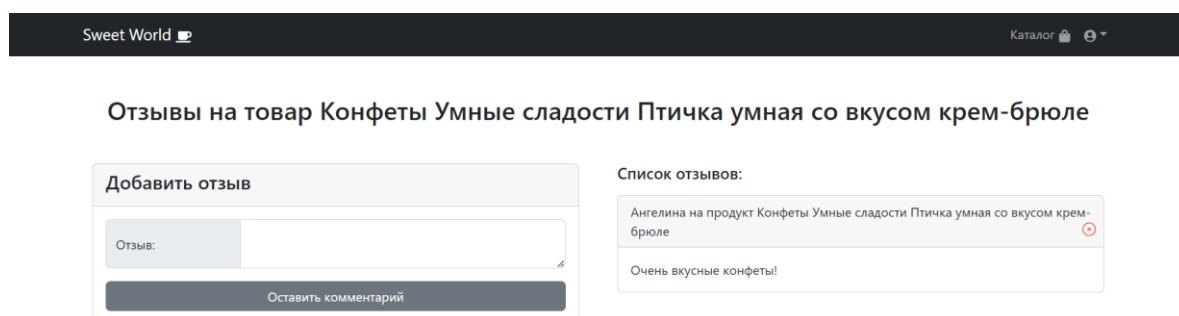


Рисунок 38 - Экран добавления комментария

3.4.3 Для администратора

Для администратора доступен переход от главной страницы (Рисунок 39) к панели администрирования, где при помощи встроенных инструментов

Django возможно управление товарами, их категориями (фабриками), комментариями и пользователями (Рисунки 40-43).

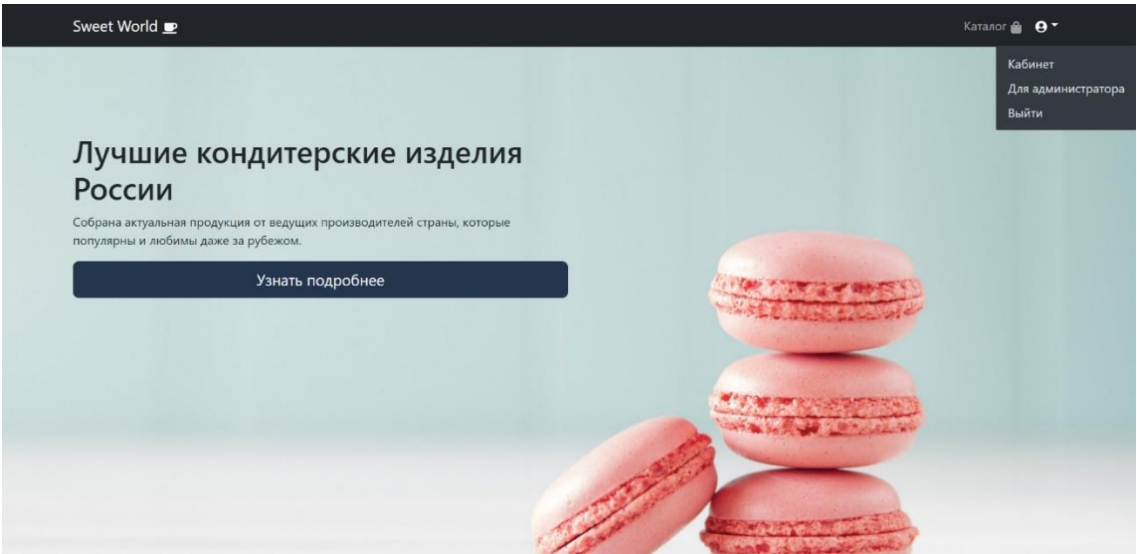


Рисунок 39 - Главная страница с панелью для администратора

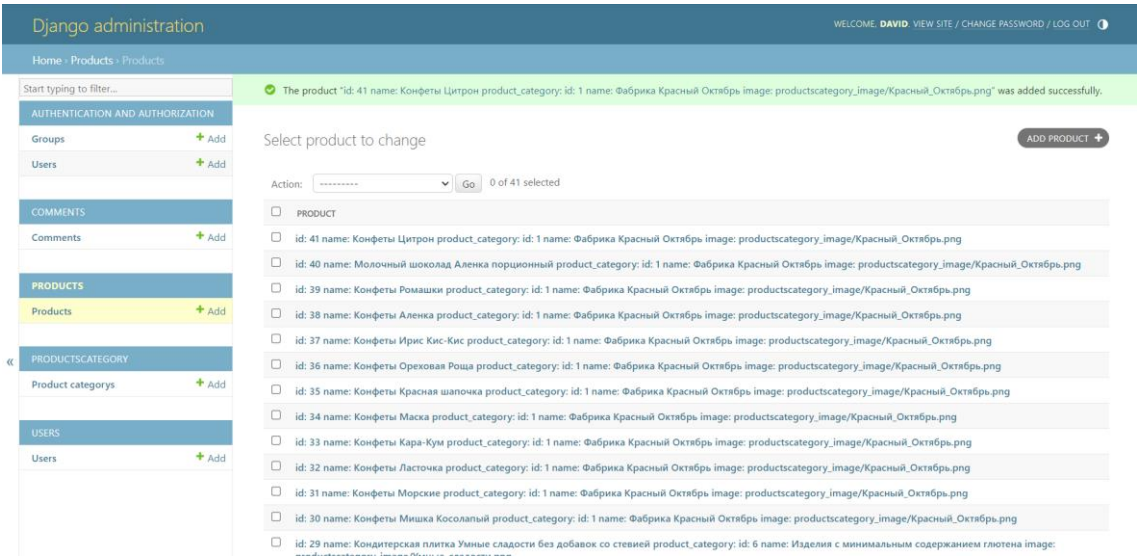


Рисунок 40 - Управление товарами

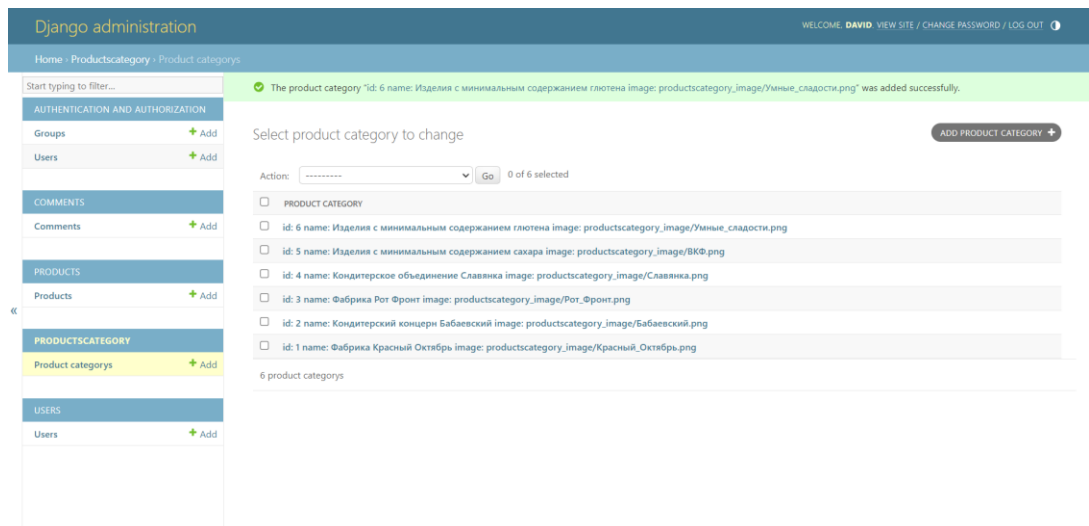


Рисунок 41 - Управление категориями (фабриками) товаров

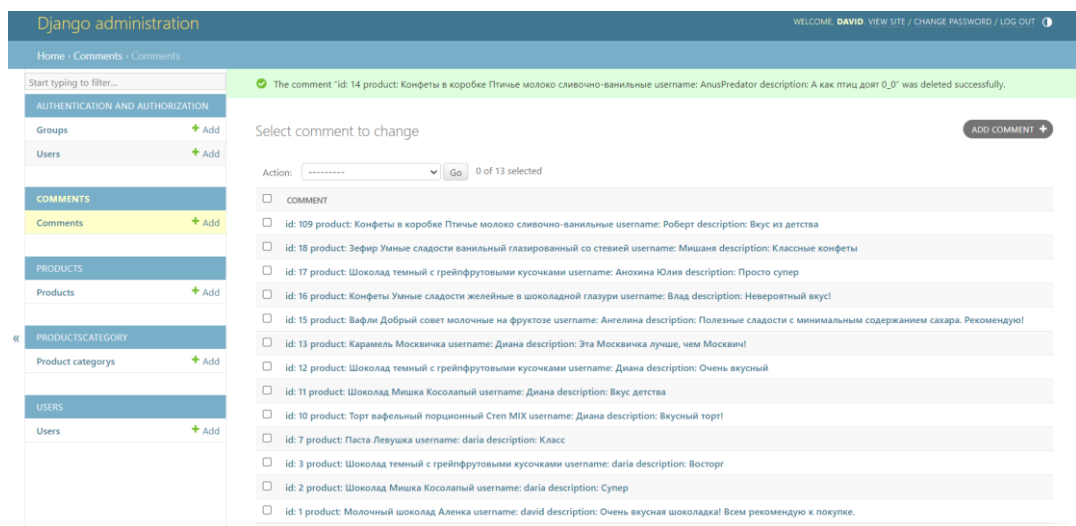


Рисунок 42 - Управление отзывами

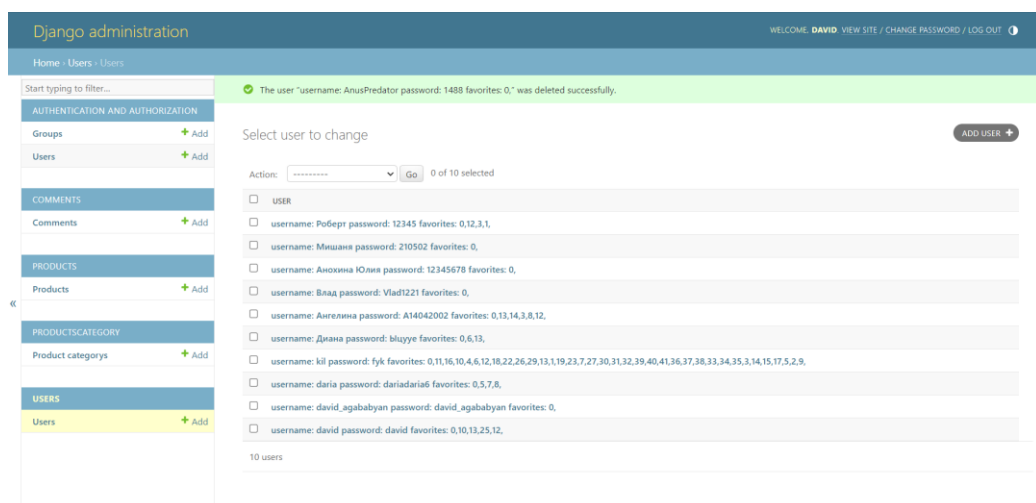


Рисунок 43 - Управление пользователями

4 Тестирование

4.1 Модульное тестирование (unit-тесты)

Unit-тесты — это тесты, которые проверяют корректность работы отдельных компонентов (или модулей) программного продукта. Они помогают убедиться, что каждый из компонентов работает правильно, и позволяют быстро обнаруживать ошибки и дефекты в коде [6].

Целью unit-тестов является проверка, что каждая единица программного кода работает правильно в изоляции от других компонентов системы. Unit-тесты проверяют, что методы возвращают ожидаемые результаты при заданных входных данных, а также что ошибочные ситуации обрабатываются корректно [6].

Тестирование серверной части осуществлялось в папках `user` и `comments`, в файлах `test.py` (Рисунки 44-45). Было создано по четыре теста, результаты которого приведены на Рисунке 46.

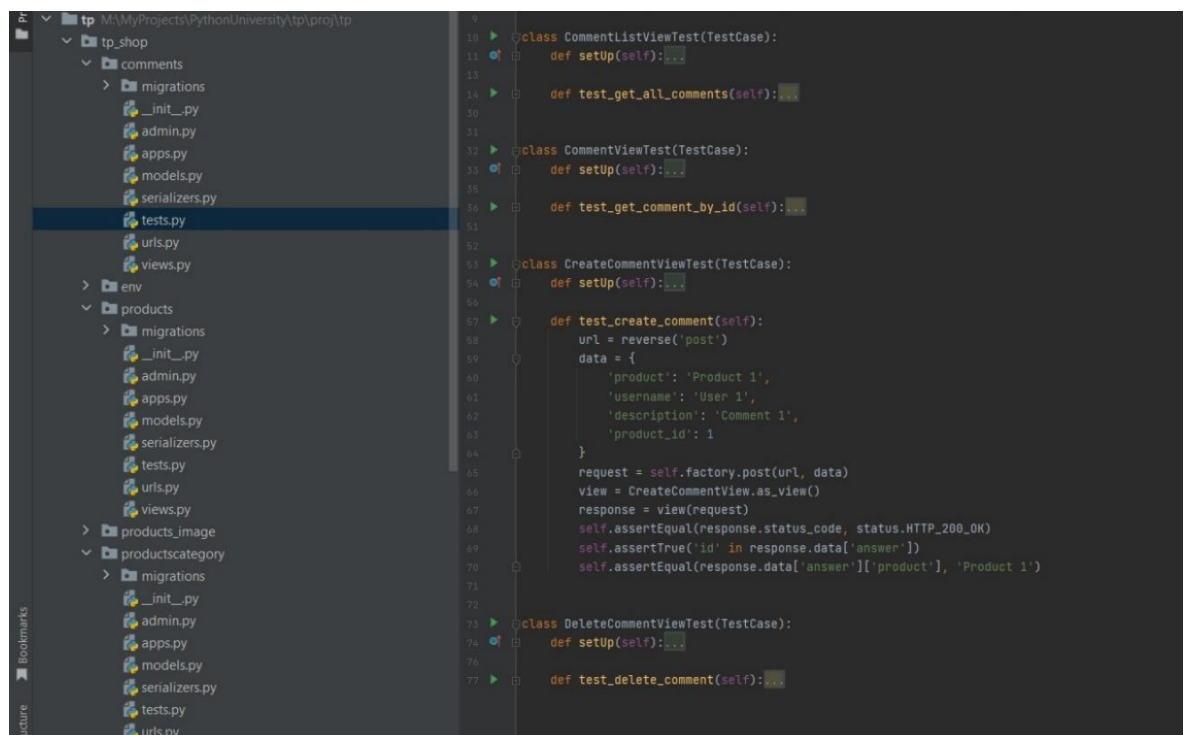


Рисунок 44 - Unit-тестирование для комментариев

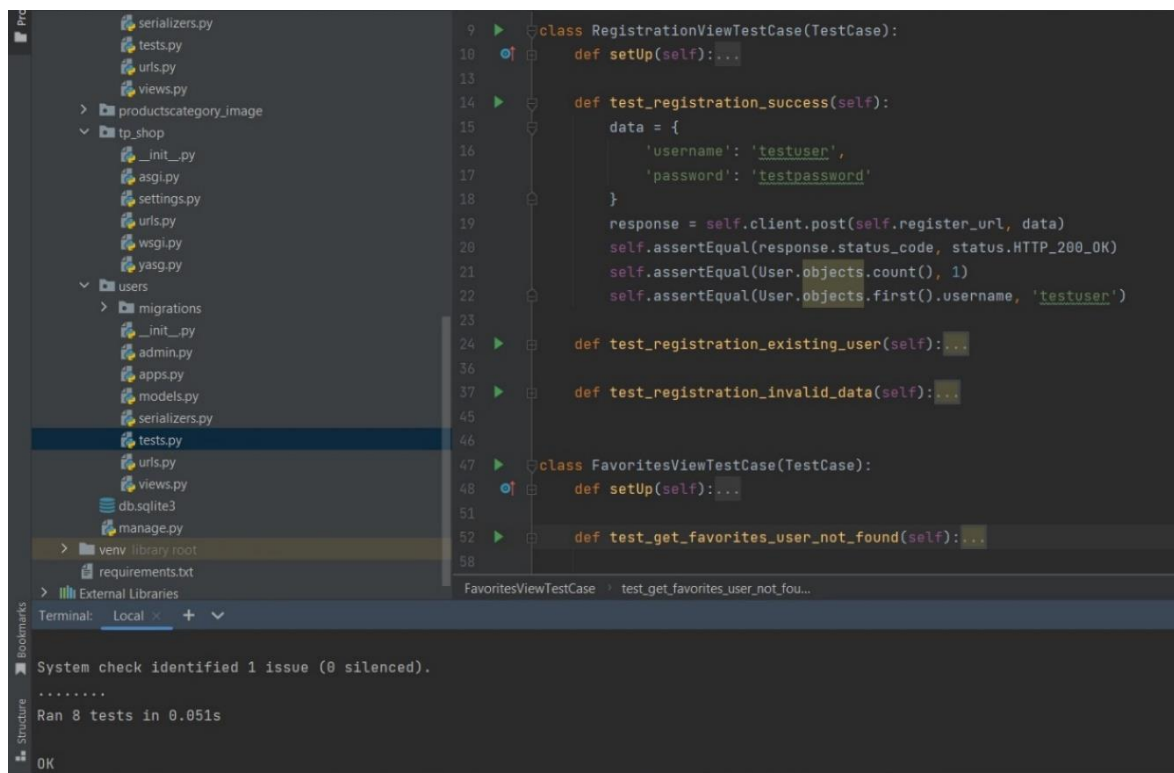


Рисунок 45 - Unit-тестирование для пользователей

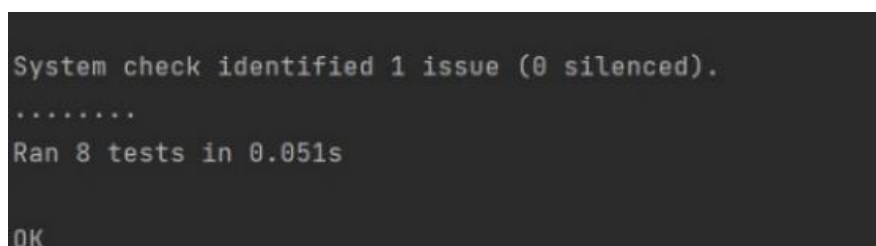


Рисунок 46 - Результат тестирования

4.2 Дымовое тестирование

Дымовое тестирование — это тип тестирования, при котором производится кратковременная проверка основной функциональности системы, чтобы убедиться в отсутствии серьезных проблем, ошибок или сбоев.

Такой тип тестирования полезен, чтобы обнаружить крупные проблемы в функциональности системы до того, как будут проведены более подробные и насыщенные тесты. Если в процессе дымового тестирования найдены ошибка или проблемы, то это может стать причиной для проведения

дополнительных и более детальных тестов, чтобы устранить недочеты и проблемы в работе программы.

В ходе дымового тестирования выполняются базовые операции или сценарии, которые предполагаются как наиболее важные и часто используемые пользователем. Далее представлены результаты дымового тестирования для основных сценариев неавторизованного, авторизованного пользователей и администратора (Таблица 1-3).

Таблица 1 - Результаты дымового тестирования для неавторизованного пользователя

Тестовый сценарий	Результат теста
Регистрация	Пройден
Авторизация	Пройден
Просмотр каталога товаров	Пройден
Просмотр категории каталога товаров	Пройден
Осуществление поиска товара по наименованию	Пройден
Просмотр комментариев	Пройден

Таблица 2 - Результаты дымового тестирования для неавторизованного пользователя

Тестовый сценарий	Результат теста
Регистрация	Пройден
Авторизация	Пройден
Просмотр каталога товаров	Пройден
Просмотр категории каталога товаров	Пройден
Осуществление поиска товара по наименованию	Пройден

Добавление в избранное	Пройден
Очистка списка избранного	Пройден
Получение информации о пользователе в личном кабинете	Пройден
Просмотр комментариев	Пройден
Добавление комментария	Пройден
Удаление комментария	Пройден

Таблица 3 - Результаты дымового тестирования для администратора

Тестовый сценарий	Результат теста
Авторизация	Пройден
Просмотр категорий каталога товаров	Пройден
Добавление категории каталога	Пройден
Изменение категории каталога	Пройден
Удаление категории каталога	Пройден
Просмотр списка товаров	Пройден
Добавление товара	Пройден
Изменение данных о товаре	Пройден
Удаление товара	Пройден
Просмотр комментариев	Пройден
Просмотр списка пользователей и их данных	Пройден
Добавление пользователя	Пройден
Удаление пользователя	Пройден

4.3 Тестирование пользовательского интерфейса

Тестирование пользовательского интерфейса (GUI-тестирование) — это процесс тестирования элементов управления в приложении, который помогает убедиться, что интерфейс соответствует ожидаемой функциональности, включая проверку различных функций и элементов, таких как окна, диалоговые окна, кнопки, переключатели, выпадающие списки, формы, меню и т.д.

Задача проведения GUI-тестов — убедиться, что в функциях пользовательского интерфейса отсутствуют дефекты. В Таблице 4 представлена часть результатов тестирования пользовательского интерфейса. В ней отражены тестовые сценарии для неавторизованного пользователя.

Таблица 4 - Результаты GUI-тестирования

Тестовый сценарий	Ожидаемый результат	Статус теста
Нажатие на кнопку «Узнать подробнее»	Переход на страницу каталога товаров	Пройден
Нажатие на логотип	Переход на главную страницу	Пройден
Нажатие на кнопку «Главная» в меню/футере	Переход на главную страницу	Пройден
Нажатие на кнопку «Каталог» в меню/футере	Переход на страницу каталога товаров	Пройден
Нажатие на кнопку «Кабинет» в меню/футере	Переход на страницу личного кабинета пользователя	Пройден
Нажатие на кнопку «Войти» в меню	Переход на страницу авторизации	Пройден
Нажатие на кнопку	Добавление товара в избранное	Пройден

«Добавить в избранное»	и вывод сообщения	
Нажатие на кнопку «Посмотреть отзывы»	Переход на страницу с комментариями по конкретному товару	Пройден
Нажатие на кнопку «Создать комментарий»	Добавление комментария на сайте	Пройден
Нажатие на кнопку «Вернуться к каталогу»	Переход на страницу каталога товаров	Пройден
Нажатие на кнопку «Войти»	Переход на главную страницу	Пройден
Нажатие на ссылку «Нужен аккаунт? Зарегистрируйся!»	Переход на страницу регистрации	Пройден
Нажатие на кнопку «Создать аккаунт»	Переход на страницу авторизации	Пройден
Нажатие на ссылку «Уже есть аккаунт? Авторизоваться»	Переход на страницу авторизации	Пройден
Нажатие на кнопку «Поиск»	Осуществление поиска по каталогу кондитерских изделий	Пройден

Заключение

В ходе выполнения курсовой работы были выполнены все поставленные задачи. Был разработан каталог кондитерских изделий с возможностью добавления товаров в список избранного и добавления комментариев пользователями.

В начале разработки был проведен анализ предметной области, определены основные требования к разрабатываемой системе, определены основные сценарии веб-приложения. По результатам разработки проводился ряд тестов с целью проверки работоспособности системы.

Благодаря проекту были решены следующие задачи у пользователя:

- Ознакомиться с ассортиментом кондитерских фабрик от ведущих компаний России, в том числе для диабетиков и худеющих людей.
- Получить информацию о пищевой и энергетической ценности товаров.
- Добавлять товары в список «Избранное» и очищать его.
- Просмотреть, добавлять и удалять комментарии.

И следующие для администратора:

- Добавлять, удалять, просматривать и редактировать товары и категории (кондитерские компании).
- Управлять комментариями пользователей.
- Редактировать права пользователей.

Таким образом, итоги разработки, проверенные в ходе тестирования, позволяют достигнуть поставленных заказчиком целей и решают сформулированные в начале разработки задачи.

Список использованной литературы

1. Django - что это за фреймворк на Python: возможности и преимущества [Электронный ресурс]. — Режим доступа: <https://blog.skillfactory.ru/glossary/django/>. — Заглавие с экрана. — (Дата обращения: 14.03.2023).
2. React – JavaScript-библиотека для создания пользовательских интерфейсов [Электронный ресурс]. — Режим доступа: <https://ru.react.js.org/?ref=dtf.ru>. — Заглавие с экрана. — (Дата обращения: 16.03.2023).
3. Полное руководство по 14 типам диаграмм UML [Электронный ресурс]. — Режим доступа: <https://www.cybermedian.com/ru/a-comprehensive-guide-to-14-types-of-uml-diagram/>. — Заглавие с экрана. — (Дата обращения: 18.03.2023).
4. Диаграммы сотрудничества [Электронный ресурс]. — Режим доступа: <https://helpiks.org/9-53448.html>. — Заглавие с экрана. — (Дата обращения: 19.03.2023).
5. Тестирование API с помощью Swagger: особенности и преимущества [Электронный ресурс]. — Режим доступа: <https://blog.ithillel.ua/ru/articles/apitesting-with-swagger>. — Заглавие с экрана. — (Дата обращения: 20.05.2023).
6. Модуль unittest: тестируем свои программы [Электронный ресурс]. — Режим доступа: <https://pythonworld.ru/moduli/modul-unittest.html>. — Заглавие с экрана. — (Дата обращения: 25.05.2023).