

РЕФЕРАТ

Отчет X с., X рис., 11 источн.

ИНТЕРНЕТ, РАЗРАБОТКА СЕРВЕНОЙ ЧАСТИ ПРИЛОЖЕНИЯ, ВЕБ-ПРИЛОЖЕНИЕ, SPRING, SPRING BOOT, DOCKER, POSTGRESQL, GRADLE, СОВРЕМЕННЫЕ ТЕХНОЛОГИИ, ЦВЕТЫ

Объект исследования – веб-приложение «Цветочный магазин».

Предмет исследования – разработка серверной части веб-приложения «Цветочный магазин».

В ходе работы был проведен анализ предметной области и обзор веб-приложений с аналогичной тематикой.

Рассмотрен процесс создания веб-приложения, используемый программный инструментарий и среда разработки.

Результатом работы является веб-приложение «Цветочный магазин», который предлагает пользователям пополнять корзину, оформлять заказы и просматривать историю покупок в личном кабинете.

Учитывая постоянную потребность в качественных цветочных композициях и подарках, предполагается, что в перспективе число пользователей веб-приложения для цветочного магазина будет постоянно расти. Разнообразие и свежесть предложений, а также удобство использования платформы, будут способствовать привлечению новых клиентов.

СОДЕРЖАНИЕ

[ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ 5](#_Toc165936115)

[ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ 6](#_Toc165936116)

[ВВЕДЕНИЕ 7](#_Toc165936117)

[1 ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ 8](#_Toc165936118)

[1.1 Анализ предметной области 8](#_Toc165936119)

[1.2 Структура базы данных 9](#_Toc165936120)

[2 ОБОСНОВАНИЕ ВЫБОРА ТЕХНОЛОГИЙ 11](#_Toc165936121)

[2.1 Используемое прикладное программное обеспечение 11](#_Toc165936122)

[2.2 Используемые технологии 11](#_Toc165936123)

[3 РАЗРАБОТКА ПРИЛОЖЕНИЯ 13](#_Toc165936124)

[3.1 Структура приложения 13](#_Toc165936125)

[3.2 Подключение к базе данных 13](#_Toc165936126)

[3.3 Разработка серверной части приложения 14](#_Toc165936127)

[3.4 Тестирование приложения 14](#_Toc165936128)

[3.5 Контейнеризация 22](#_Toc165936129)

[ЗАКЛЮЧЕНИЕ 24](#_Toc165936130)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 25](#_Toc165936131)

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

В настоящем отчете применяют следующие термины с соответствующими определениями:

|  |  |
| --- | --- |
| Хостинг | * услуга по предоставлению ресурсов для размещения информации на сервере, постоянно имеющем доступ к сети |
| Фреймворк | * программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта |
| Render | * бесплатный хостинг для размещения приложений |
| Dockerfile | * файл для предварительной работы, набор инструкций, который нужен для записи образа |
|  |  |
|  |  |
|  |  |

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

В настоящем отчете применяются следующие сокращения и обозначения:

|  |  |
| --- | --- |
| HTML | * HyperText Markup Language, язык гипертекстовой разметки |
| JPA | * Java Persistence API |
| JS | * JavaScript, язык программирования |
| UML | * Unified Modeling Language |

ВВЕДЕНИЕ

Целью курсовой работы является процесс разработки серверной части веб-приложения "Цветочный магазин", а также анализ предметной области с целью определения основных требований к приложению. Основная задача заключается в разработке собственной архитектуры и функционала приложения, используя язык программирования Java [1], фреймворк Spring [2] и базу данных PostgreSQL [3]. Для тестирования функционала приложения будет использоваться инструмент Postman [4], а для контейнеризации Docker [5]. Для достижения поставленной цели были сформулированы следующие задачи:

* провести анализ предметной области, связанной с разрабатываемым приложением,
* выбрать инструменты для ведения разработки,
* разработать приложение с использованием выбранных средств разработки,
* протестировать разработанное приложение.

Для выполнения этих задач применяются методы анализа и сравнительного исследования. Информационной базой для этой работы являются знания, полученные в ходе практических занятий курса «Шаблоны программных платформ на языке Java», а также данные из интернет-ресурсов.

В разделе описания логической структуры приведены методы анализа, которые были применены в ходе работы, и структура базы данных приложения.

В разделе, посвященном обоснованию выбора технологий, представлена информация о использованных программных инструментах.

Раздел, посвященный разработке приложения, содержит информацию о созданном приложении, его тестировании и общей структуре.

В разделе заключения подводятся итоги по выполненной разработке и приводятся ссылки на исходный код и хостинг веб-приложения.

1 ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ

1.1 Анализ предметной области

Анализ предметной области проводился среди веб-приложений на тематику «Цветочный магазин». Были выбраны три веб-приложения: Flowwow [6], простоцветы [7] и Мега Цветы [8], представленные на рисунках 1-3.

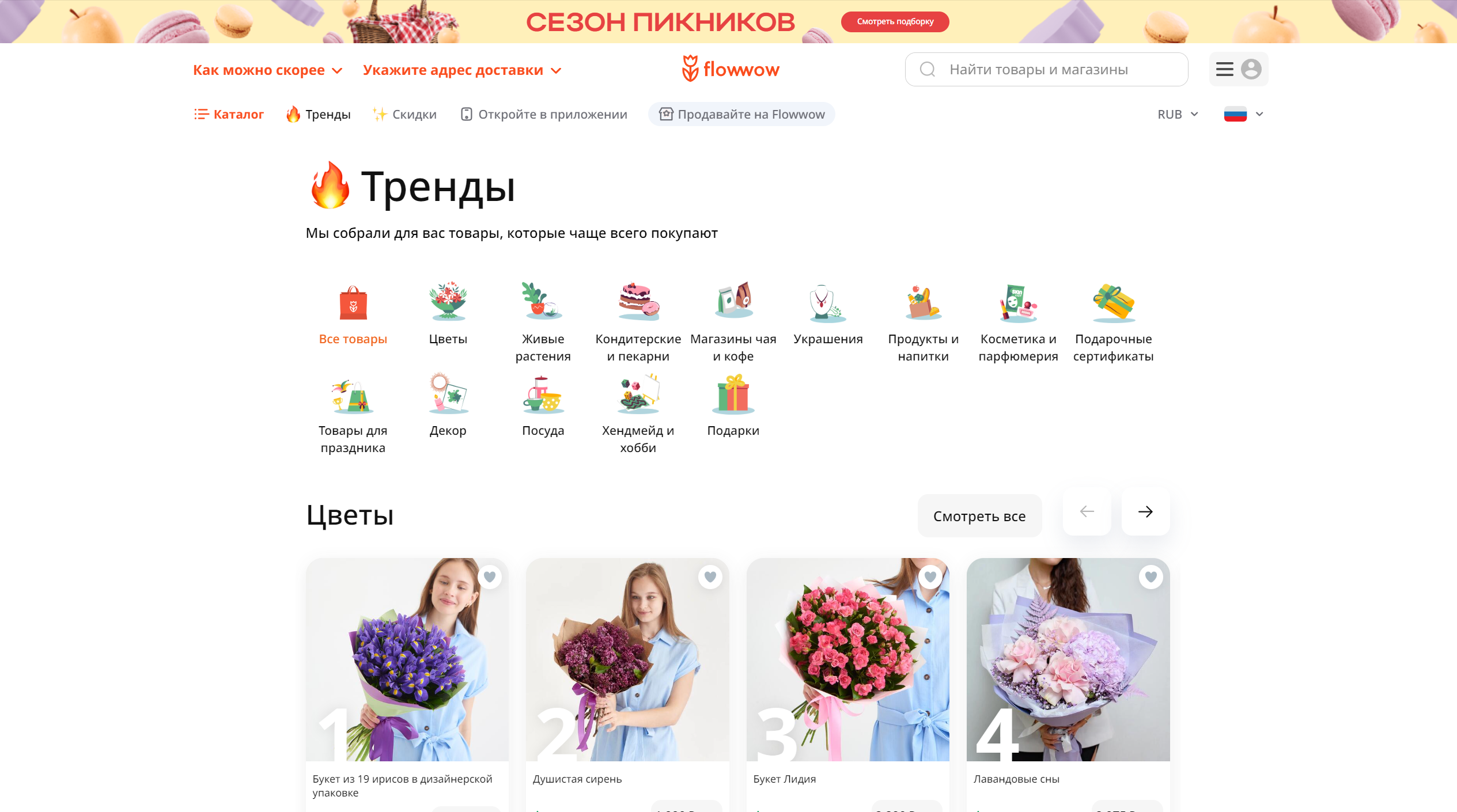


Рисунок 1 – Клиентская часть сайта Четыре Лапы

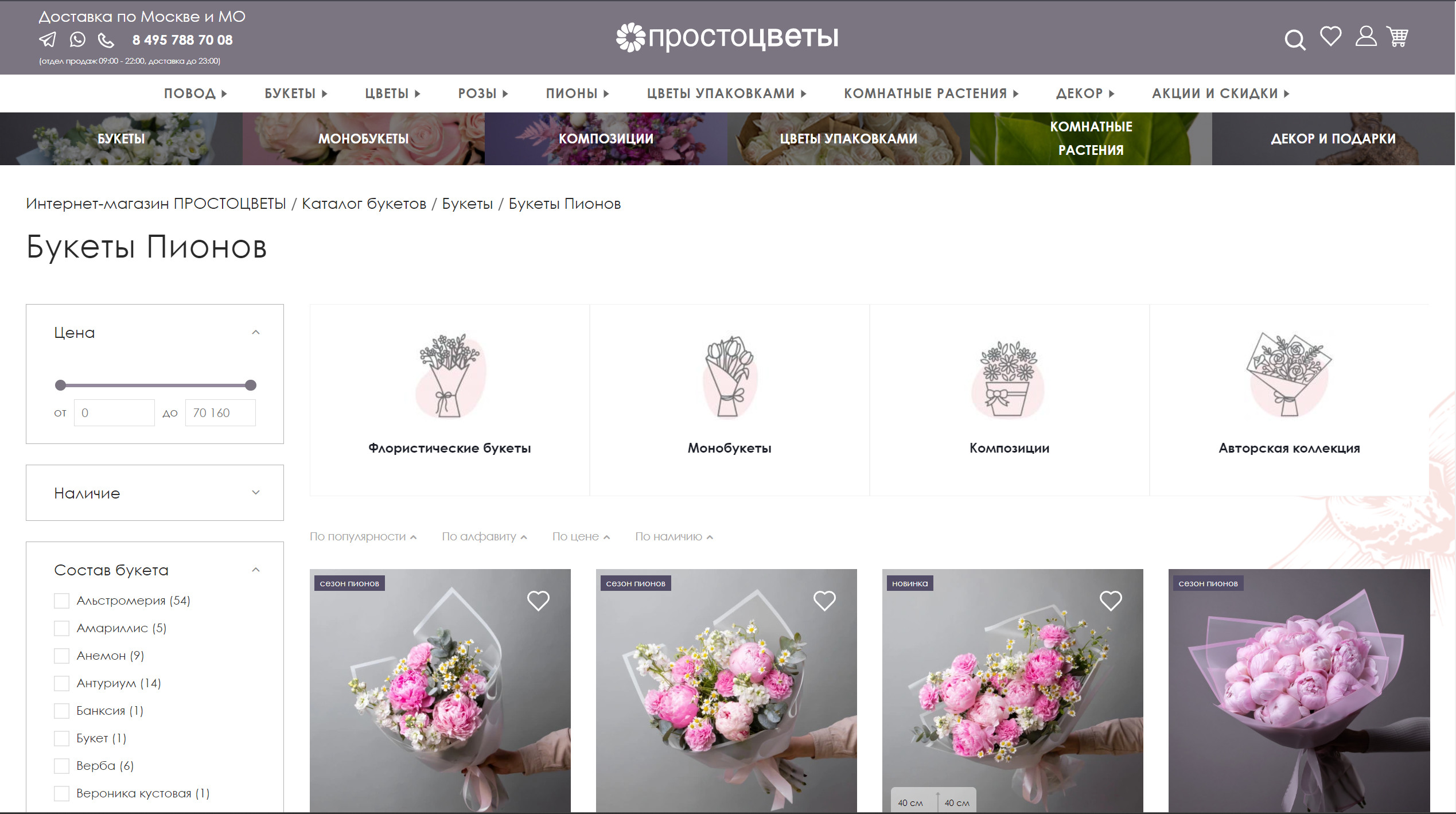


Рисунок 2 – Клиентская часть сайта ZooMag

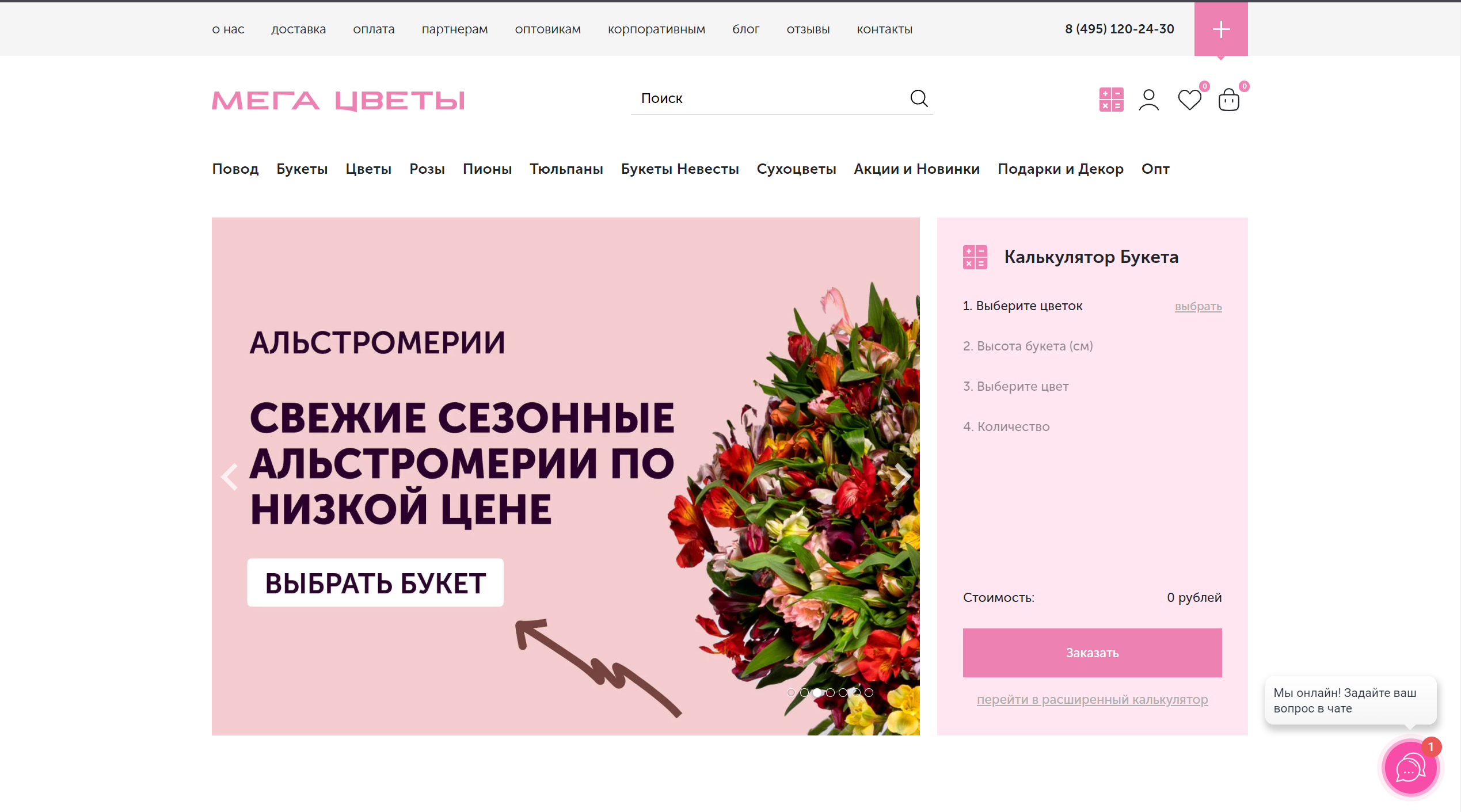


Рисунок 3 – Клиентская часть сайта Petshopru

Все упомянутые веб-приложения включают функционал регистрации и входа в аккаунт, возможность просмотра товаров, добавление их в корзину, оформление заказа на основе содержимого корзины, а также удобный просмотр истории заказов пользователя.

Из анализа данных веб-приложений следует выделить ключевые возможности, которыми должен обладать разрабатываемый веб-ресурс данной тематики:

– регистрация нового аккаунта,

– вход в существующий аккаунт,

– выход из аккаунта,

– просмотр товаров,

– добавление товаров в корзину,

– оформление заказа,

– просмотр заказов внутри личного кабинета.

1.2 Структура базы данных

Для хранения пользователей, с хешированным паролем, товаров и заказов веб-приложения, используется база данных в определенной структурой, описанной с помощью UML-диаграмм на рисунке 4.

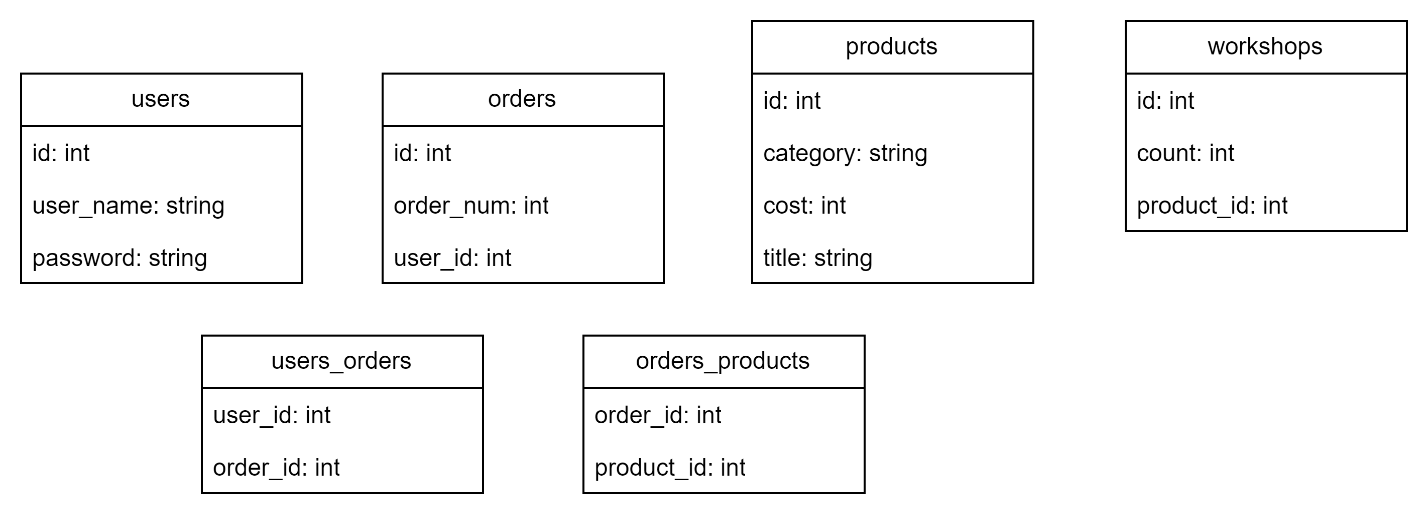


Рисунок 4 – Структура базы данных приложения

2 ОБОСНОВАНИЕ ВЫБОРА ТЕХНОЛОГИЙ

2.1 Используемое прикладное программное обеспечение

Для разработки приложения использовалась интегрированная среда разработки программного обеспечения IntelliJ IDEA, разработанная компанией JetBrains.

2.2 Используемые технологии

При разработке приложения был использован язык программирования Java, система автоматической сборки Gradle [8], фреймворк Spring с его дополнительными инструментами, такими как Spring Data JPA для работы с базой данных (который является надстройкой над Hibernate) и Spring Security для обеспечения функционала регистрации и авторизации. Для создания динамических HTML страниц был использован шаблонизатор Thymeleaf [9]. В качестве базы данных применялась PostgreSQL. Для тестирования использовался Postman, а для развертывания – Docker. Приложение размещено на хостинге Render [10], включая базу данных.

Java - язык программирования, который обеспечивает возможность разработки приложений для различных устройств и платформ, включая персональные компьютеры, мобильные устройства и серверы, благодаря своей многоцелевой природе.

Gradle – это мощный инструмент автоматизации сборки с открытым исходным кодом, который обеспечивает быструю и надежную работу. Его гибкий декларативный язык сборки позволяет автоматизировать разнообразные сценарии создания программного обеспечения, что делает его привлекательным выбором для разработчиков.

Spring представляет собой полноценную модель разработки и настройки для современных корпоративных приложений, созданных на базе языка программирования Java, что позволяет развертывать их на различных платформах. Основной фокус Spring - это инфраструктурная поддержка на уровне приложений, что дает возможность разработчикам избежать привязки к конкретным средам развертывания.

Spring Data JPA, в составе Spring Data, упрощает создание репозиториев на основе Java Persistence API (JPA), что существенно упрощает разработку приложений на Spring, использующих доступ к данным.

Spring Security предоставляет эффективные и гибко настраиваемые инструменты для аутентификации и управления доступом в Java приложениях. Его ключевым преимуществом является легкость расширения, что позволяет удовлетворять индивидуальные требования проекта.

Thymeleaf – это инновационный серверный механизм шаблонов на языке программирования Java, который разработан для удобного создания качественных макетов веб-страниц и автономных приложений. Его особенности включают широкий набор функций и гибкую настройку, делая его привлекательным выбором для разработчиков.

PostgreSQL представляет собой мощную и бесплатную систему управления базами данных с открытым исходным кодом. Эта система предоставляет разработчикам и администраторам возможность создавать, хранить и эффективно обрабатывать данные, используя стандартный язык запросов – SQL (Structured Query Language). Благодаря своей гибкости и расширяемости, PostgreSQL широко применяется в различных проектах, включая веб-приложения, корпоративные системы и аналитические платформы.

Postman - инструмент для тестирования API, который предоставляет широкий функционал, включая отправку запросов, создание документации, запуск автотестов и управление версиями.

Docker – это платформа для упаковки приложений в контейнеры со всеми необходимыми зависимостями для их запуска в различных средах.

Render – это комплексная облачная платформа, предназначенная для разработки, развертывания и управления приложениями и веб-сайтами. Она обеспечивает автоматическое развертывание приложений из репозитория Git.

3 РАЗРАБОТКА ПРИЛОЖЕНИЯ

3.1 Структура приложения

Рабочая директория приложения содержит в себе несколько уровней. Серверная часть находиться в директории java/com/flowershop, а клиентская часть, код для страниц, стили, скрипты и изображения, в директории resourses. Подробнее рабочая директория изображена на рисунке 5.

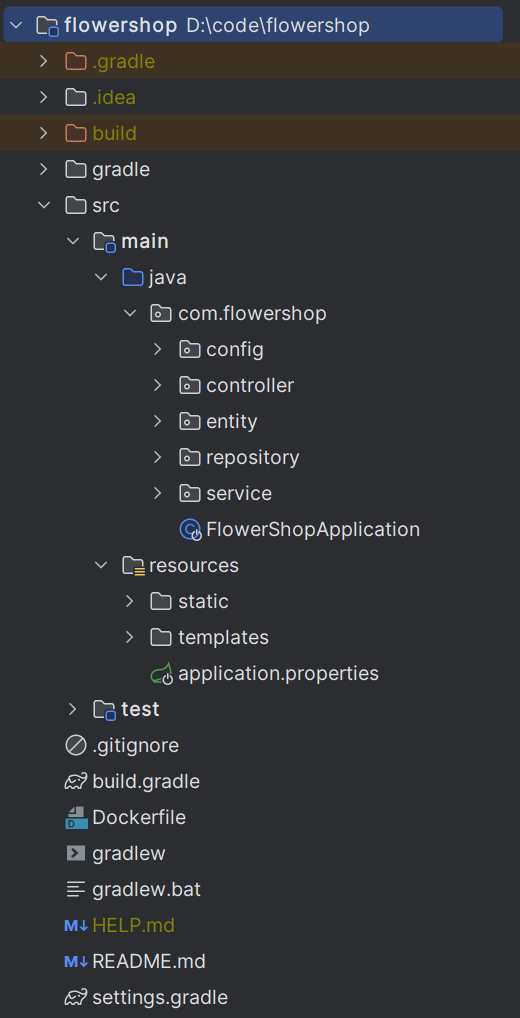


Рисунок 5 – Рабочая директория приложения

3.2 Подключение к базе данных

Для осуществления подключения к базе данных приложения требуется внести соответствующие настройки в файл application.properties, как показано на рисунке 6. Эти настройки включают строку подключения, а также учетные данные пользователя и пароль для доступа к конкретной базе данных. В контексте данного приложения используется база данных, размещенная на хостинге, поэтому все необходимые данные для подключения предоставляются самим хостингом.

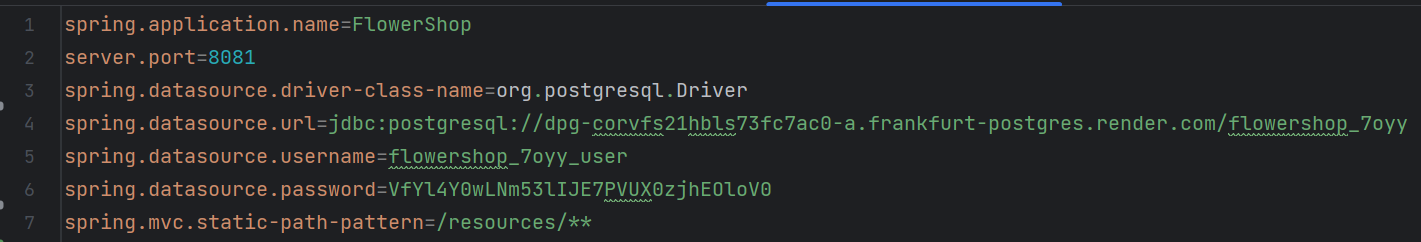


Рисунок 6 – Файл application.properties

3.3 Разработка серверной части приложения

Процессы авторизации и регистрации в приложении реализованы с использованием фреймворка Spring Security. Для обеспечения функциональности этих процессов необходимо настроить соответствующий файл конфигурации, представленный на рисунке 7.



Рисунок 7 – Метод настройки авторизации и регистрации

Для обработки конечных точек авторизации/регистрации написан контроллер EntryController, показанный на рисунке 8.

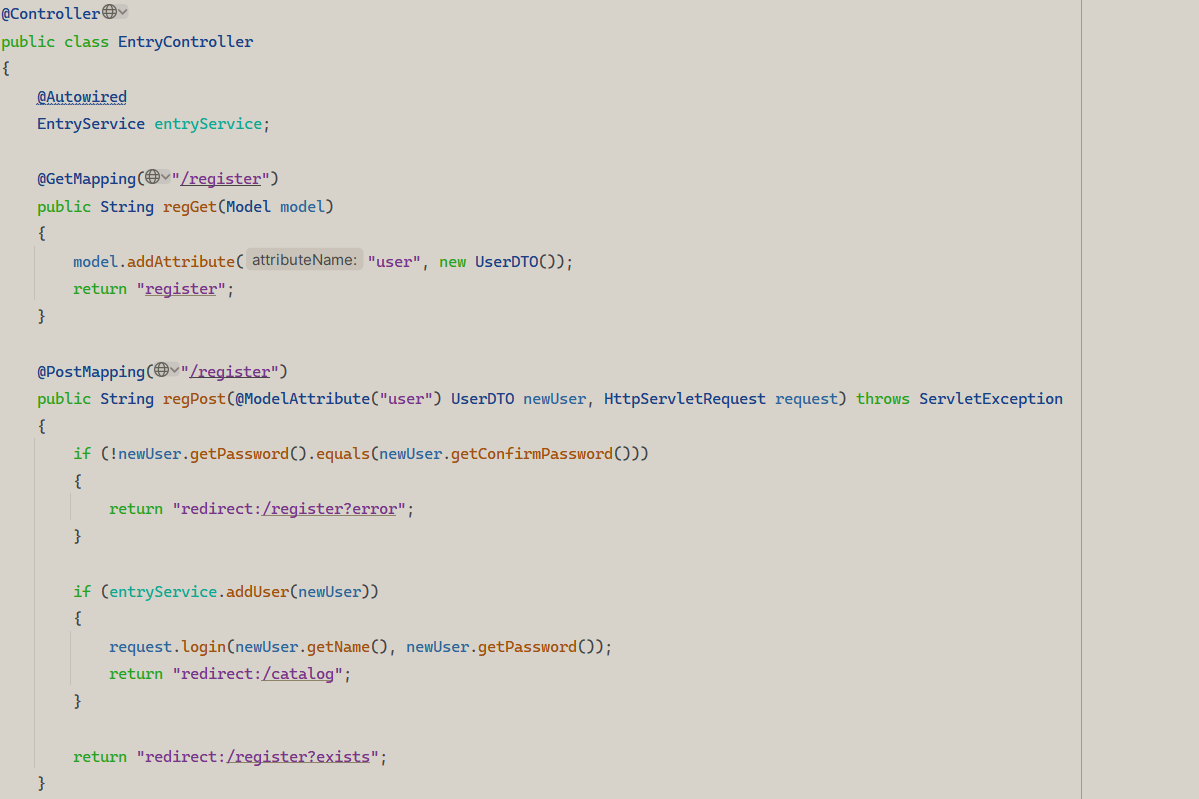


Рисунок 8 – Контроллер для входа и регистрации пользователя

После входа/регистрации, пользователь перенаправляется на страницу с товарами, находящуюся по адресу /catalog. На рисунке 9 показана обработка данной конечной точки.



Рисунок 9 – Обработка вывода товаров

При выборе товаров они попадают в корзину, рисунок 10.

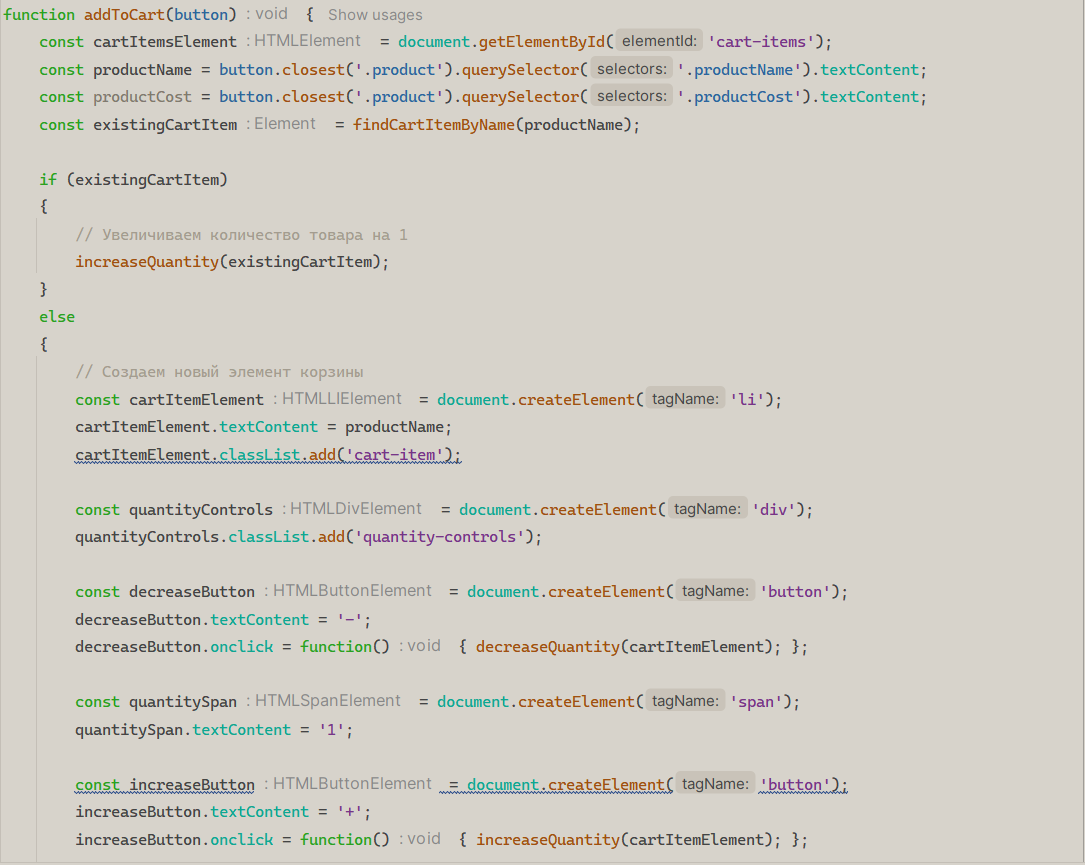


Рисунок 10 – Добавление в корзину товаров

После заполнения корзины товарами, пользователей может оформить заказ, доставая список из хранилища и передавая его на сервер. Контроллер и сервис для обработки оформления заказа изображены на рисунках 11-12.



Рисунок 11 – Контроллер для обработки заказа



Рисунок 12 – Сервис для обработки заказа

После оформления заказа он добавляется к пользователю. Код данного процесса изображен на рисунках 13-14.



Рисунок 13 – Контроллер для создания заказа



Рисунок 14 – Сервис для создания заказа

Для просмотра всех заказов пользователь может зайти в свою учетную запись, где будут выведены личные данные пользователя и список его заказов. Код для учетной записи изображен на рисунках 15-16.



Рисунок 15 – Контроллер для учетной записи



Рисунок 16 – Сервис для учетной записи

Для доступа к данным из базы данных используются репозитории, как например на рисунке 17.



Рисунок 17 – Сервис для учетной записи

3.4 Тестирование приложения

Используя программу Postman и браузер, были протестированы несколько конечных точек приложения. Для начала протестирован вывод каталога товаров, показанный на рисунке 18.

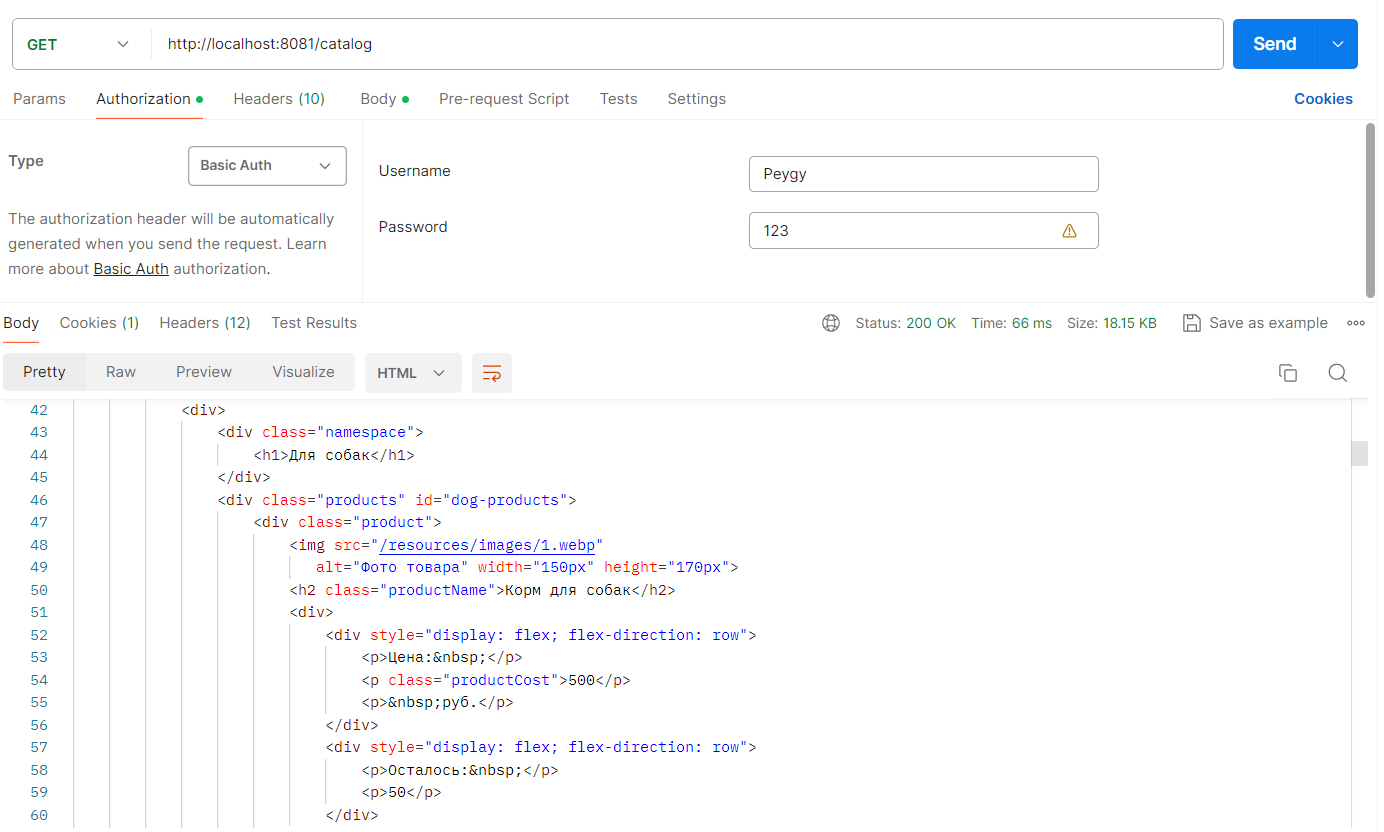


Рисунок 18 – GET запрос тестирования страницы товаров

Тестирование входа в аккаунт показан на рисунке 19.

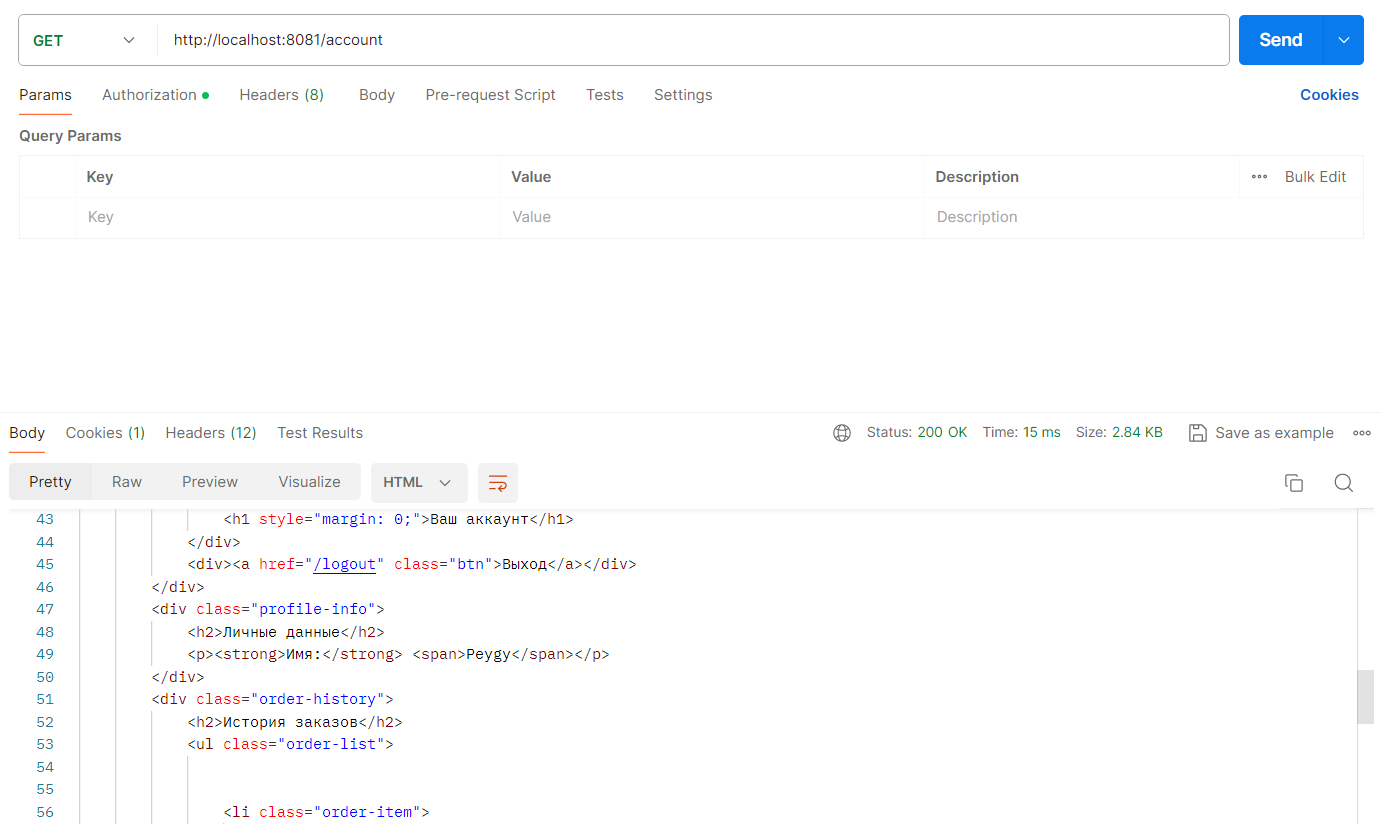


Рисунок 19 – GET запрос тестирования страницы аккаунта

Далее, с помощью браузера, протестировано добавление товаров из каталога в корзину. Тестирование показано на рисунке 20.

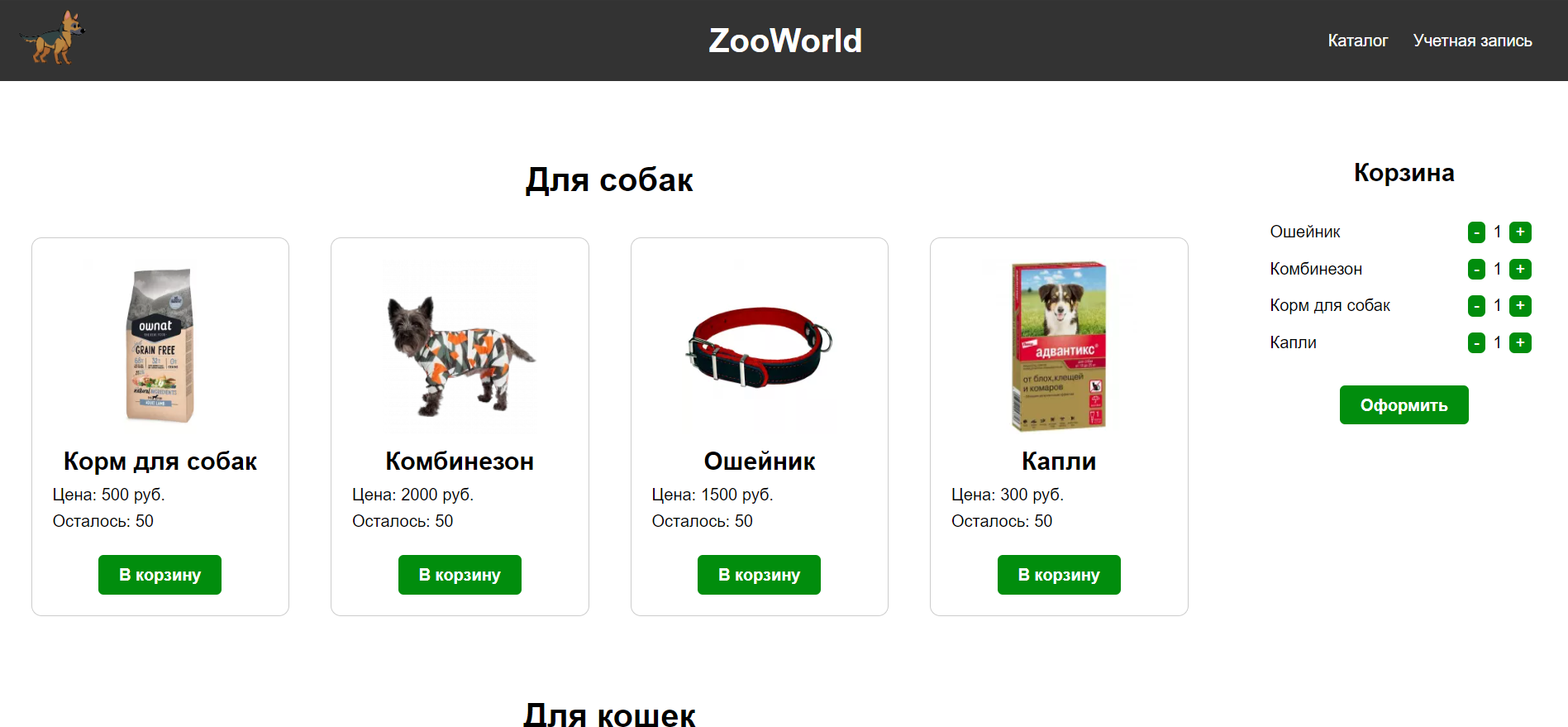


Рисунок 20 – Тестирование добавления товаров в корзину

Тестирование формирования заказа показано на рисунке 21.

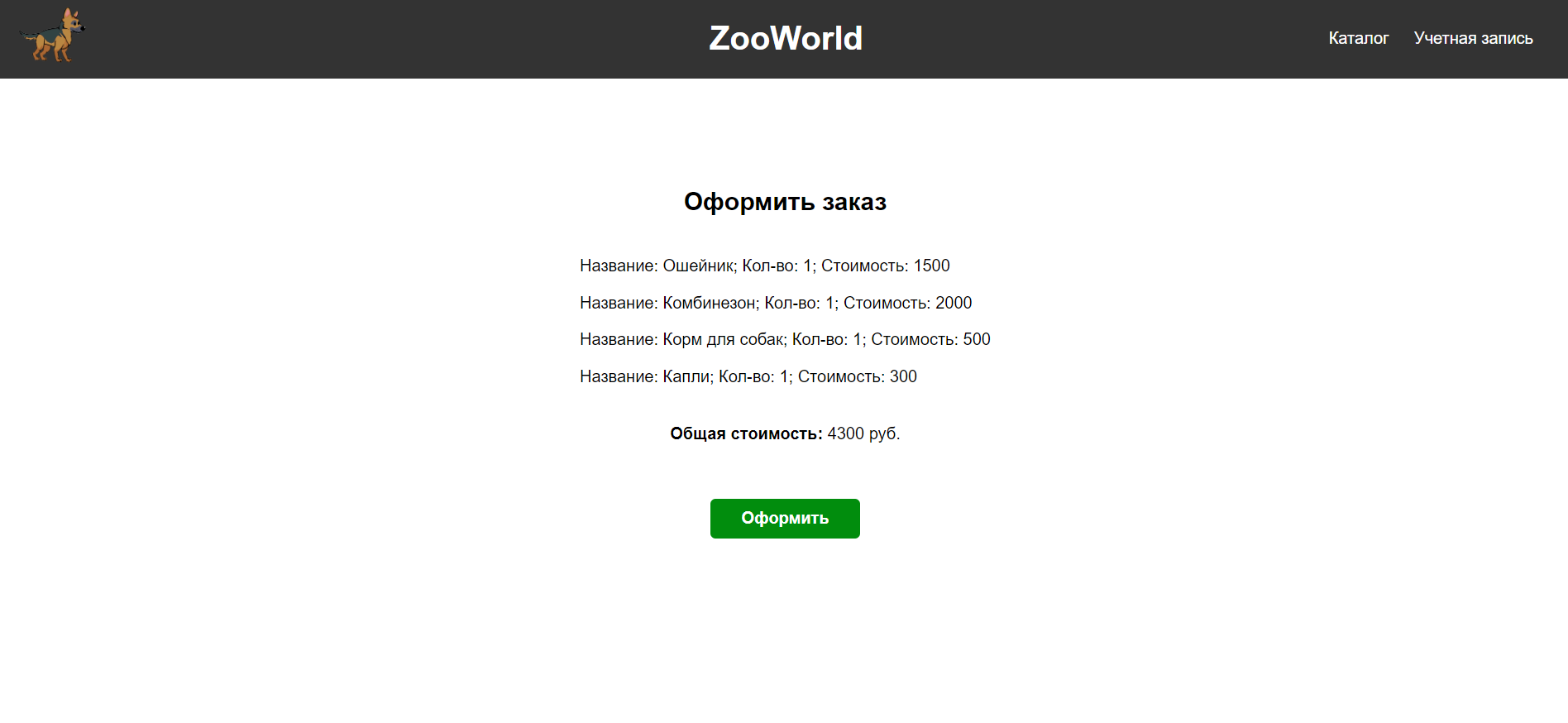


Рисунок 21 – Тестирование формирования заказа

При входе в аккаунт будет выведен список заказов, изображенный на рисунке 22.

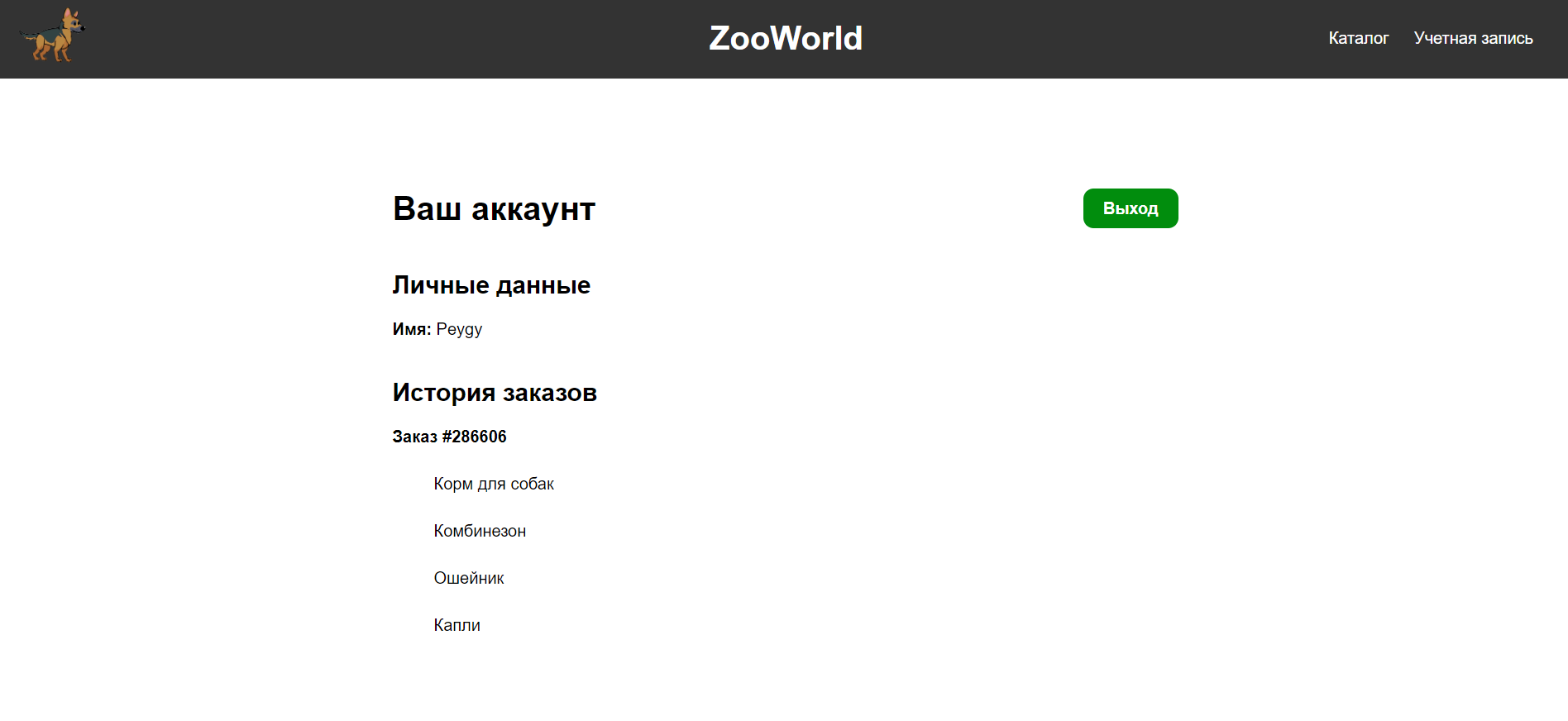


Рисунок 22 – Тестирование вывода заказов пользователя

3.5 Контейнеризация

Для развертывания на хостинге приложения, необходимо прописать Dockerfile, для создания контейнера, в котором будет работать разработанное приложения. В Dockerfile прописан основной образ, команда для копирования всех файлов в образ и запуск jar-файла. Содержимое Dockerfile показано на рисунке 23.



Рисунок 23 – Файл Dockerfile

ЗАКЛЮЧЕНИЕ

В результате процесса разработки приложения были успешно достигнуты поставленные цели и выполнены задачи. Была проведена аналитика предметной области, определены оптимальные инструменты разработки, создано приложение с применением выбранных средств и протестировано его функционирование. Результатом этого процесса является стабильное приложение, которое обладает интуитивно понятным пользовательским интерфейсом и полным набором необходимых функций.

Результаты тестирования выявили высокую оценку по критериям корректности функционирования и оптимизации использования ресурсов. Благодаря своим качественным характеристикам, данное приложение обладает потенциалом для успешной конкуренции на рынке интернет-приложений с аналогичными по тематике сервисами.

Разработанное веб-приложение было развернуто на открытый хостинг от Render: <https://javaflowershop.onrender.com>, исходный код расположен на сайте GitHub: <https://github.com/Shumila71/JavaFlowerShop>.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. OpenJDK 21 Documentation [Электронный ресурс]. – URL: https://devdocs.io/openjdk~21/ (дата обращения 05.04.2024).
2. Spring Framework [Электронный ресурс]. – URL: <https://spring-projects.ru/projects/spring-framework/> (дата обращения 12.04.2024).
3. PostgreSQL Documentation [Электронный ресурс]. – URL: <https://www.postgresql.org/docs/> (дата обращения 12.04.2024).
4. Get started in Postman [Электронный ресурс]. – URL: <https://learning.postman.com/docs/getting-started/overview/> (дата обращения 18.04.2024).
5. Руководство по Docker [Электронный ресурс]. – URL: <https://my-js.org/docs/guide/docker/> (дата обращения 25.04.2024).
6. Flowwow [Электронный ресурс]. – URL: <https://flowwow.com/moscow>/ (дата обращения 12.04.2024).
7. Простоцветы [Электронный ресурс]. – URL: <https://www.prostocvet.ru/> (дата обращения 12.04.2024).
8. Мега цветы [Электронный ресурс]. – URL: <https://megacvet24.ru/> (дата обращения 13.04.2024).
9. Gradle User Manual [Электронный ресурс]. – URL: <https://docs.gradle.org/current/userguide/userguide.html> (дата обращения 11.04.2024).
10. Thymeleaf Documentation [Электронный ресурс]. – URL: <https://www.thymeleaf.org/documentation.html> (дата обращения 16.04.2024).
11. Render Quickstarts [Электронный ресурс]. – URL: <https://docs.render.com/> (дата обращения 02.05.2024).