

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное автономное образовательное учреждение  
высшего образования

**«Южно-Уральский государственный университет  
(национальный исследовательский университет)»**

**Высшая школа электроники и компьютерных наук**

**Кафедра системного программирования**

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой, д.ф.-м.н.,  
профессор

\_\_\_\_\_ Л.Б. Соколинский

«\_\_\_»\_\_\_\_\_ 2023 г.

## **Разработка веб-сайта для цветочного магазина**

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**ЮУрГУ – 09.03.04.2023.308-277ВКР**

Научный руководитель,  
профессор кафедры СП, д.г.н,  
к.ф.-м.н.

\_\_\_\_\_ С.М. Абдуллаев

Автор работы,  
студент группы КЭ-403

\_\_\_\_\_ А.С. Гордеева

Ученый секретарь  
(нормоконтролер)

\_\_\_\_\_ И.Д. Володченко

«\_\_\_»\_\_\_\_\_ 2023 г.

Челябинск, 2023 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
**«Южно-Уральский государственный университет  
(национальный исследовательский университет)»**  
Высшая школа электроники и компьютерных наук  
Кафедра системного программирования

УТВЕРЖДАЮ

Зав. кафедрой СП

\_\_\_\_\_ Л.Б. Соколинский

06.02.2023 г.

**ЗАДАНИЕ**

**на выполнение выпускной квалификационной работы бакалавра**  
студентке группы КЭ-403  
Гордеевой Александре Сергеевне,  
обучающейся по направлению  
09.03.04 «Программная инженерия»

**1. Тема работы** (утверждена приказом ректора от 25.04.2023 г. № 753-13/12)

Разработка веб-сайта для цветочного магазина.

**2. Срок сдачи студентом законченной работы:** 05.06.2023 г.

**3. Исходные данные к работе**

3.1. React documentation. [Электронный ресурс] URL:

<https://reactjs.org/docs/getting-started.html> (дата обращения: 10.02.2023 г.).

3.2. Redux Toolkit documentation. [Электронный ресурс] URL:

<https://redux-toolkit.js.org> (дата обращения: 10.02.2023 г.).

3.3. Mock Api documentation. [Электронный ресурс] URL:

<https://mockapi.io/docs> (дата обращения: 10.02.2023 г.).

3.4. Основы HTML. [Электронный ресурс] URL:

<https://htmlacademy.ru/courses/297> (дата обращения: 10.02.2023 г.).

3.5. CSS справочник. [Электронный ресурс] URL: <http://htmlbook.ru/css> (дата обращения: 10.02.2023 г.).

#### **4. Перечень подлежащих разработке вопросов**

- 4.1. Провести анализ предметной области.
- 4.2. Определить требования к системе.
- 4.3. Спроектировать веб-сайт.
- 4.4. Разработать макет веб-сайта.
- 4.5. Реализовать фронтенд веб-сайта.
- 4.6. Провести тестирование.

**5. Дата выдачи задания: 06.02.2023 г.**

**Научный руководитель,**  
профессор кафедры СП, д.г.н, к.ф.-м.н.

С.М. Абдуллаев

**Задание принял к исполнению**

А.С. Гордеева

## ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ.....	5
1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ .....	7
1.1. Анализ аналогичных проектов .....	7
1.2. Средства разработки .....	10
2. ПРОЕКТИРОВАНИЕ .....	11
2.1. Требования к системе .....	11
2.2. Варианты использования системы .....	12
2.3. Выбор архитектуры .....	13
2.4. Диаграмма компонентов .....	15
2.5. Диаграмма активности .....	15
2.6. Проектирование дизайна веб-сайта .....	17
3. РЕАЛИЗАЦИЯ .....	18
3.1. Создание проекта .....	18
3.2. Создание каталога товаров .....	19
3.3. Пагинация, сортировка, фильтрация и поиск товаров .....	22
3.4. Разработка корзины .....	24
3.5. Разработка страницы товара .....	26
3.6. Создание страницы оформления заказа .....	27
3.7. Разработка слайдера, прокрутки .....	28
3.8. Адаптивная верстка .....	29
4. ТЕСТИРОВАНИЕ .....	30
4.1. Функциональное тестирование .....	30
4.2. Тестирование верстки.....	32
ЗАКЛЮЧЕНИЕ .....	33
ЛИТЕРАТУРА.....	34
ПРИЛОЖЕНИЯ.....	36
Приложение А. Спецификация вариантов использования.....	36
Приложение Б. Адаптивная верстка .....	41

## **ВВЕДЕНИЕ**

### **Актуальность**

У любой современной компании существует сайт. Это один из элементов престижа, ведь именно в Интернете потенциальные клиенты будут в первую очередь искать информацию о фирме.

По состоянию на январь 2022 года в Интернете насчитывалось более 5,1 миллиарда пользователей. На 18 декабря 2021 года насчитывалось более 1.9 миллиарда веб-сайтов. Согласно прогнозам, в 2022 году мировые розничные продажи в электронной коммерции составят 2022 триллиона долларов [1]. Интернет является основным источником информации в современном мире. У человека уже выработалась привычка находить данные об интересующих его услугах, предприятиях, мероприятиях и т.п. именно в Интернете.

Исходя из приведенных выше данных, можно сделать вывод о том, что наличие веб-сайта практически необходимое условие даже для не крупных организаций. Сайт позволяет компаниям повысить эффективность продаж и сделать информацию о товаре более доступной, тем самым привлекая потенциальных клиентов.

### **Постановка задачи**

Целью выпускной квалификационной работы является разработка фронтенда веб-сайта для цветочного магазина. Для достижения поставленной цели необходимо решить следующие задачи:

- 1) провести анализ предметной области;
- 2) выбрать средства разработки;
- 3) определить требования;
- 4) спроектировать веб-сайт;
- 5) разработать макет веб-сайта;
- 6) реализовать веб-сайт;
- 7) провести тестирование.

## **Структура и содержание работы**

Работа состоит из введения, 4 глав, заключения и списка литературы. Объем работы составляет 44 страницы, объем списка литературы – 15 источников.

В первой главе проводится анализ предметной области, обзор аналогов и средств разработки, на основе анализа определяются основные задачи для реализации.

Во второй главе определяются функциональные и нефункциональные требования к системе, демонстрируются диаграммы вариантов использования, компонентов и активности, а также описывается выбранная архитектура и проектирование дизайна веб-сайта.

Третья глава посвящена описанию реализации системы в соответствии с определенными требованиями и поставленными задачами.

В четвертой главе описывается тестирование приложения, приведены результаты функционального тестирования и тестирования верстки.

В приложении А содержатся таблицы, описывающие основные варианты использования (ВИ) разрабатываемого игрового приложения.

В приложении Б содержатся скриншоты веб-сайта на различных размерах экрана.

# 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

## 1.1. Анализ аналогичных проектов

Для лучшего понимания того, на какой функционал должен быть ориентирован веб-сайт, проведем анализ сайтов компаний, предоставляющих подобные услуги.

«Flowersloft.ru» [2] – сайт цветочного магазина. Из достоинств можно выделить, что сервис предоставляет возможность заказа букета двумя способами: сбор по приложенному фото в специальной форме, добавление отдельных цветов в корзину. Так же имеется полный ассортимент магазина, разделенный по категориям и уже готовые букеты. Присутствует анимация некоторых элементов.

В адаптивности изъянов не было замечено. Сайт работает одинаково хорошо на различных устройствах.

Из недостатков можно отметить небольшой каталог и слишком маленький шрифт на экранах небольшого размера, что может затруднить чтение. Главная страница рассматриваемого сайта изображена на рисунке 1.

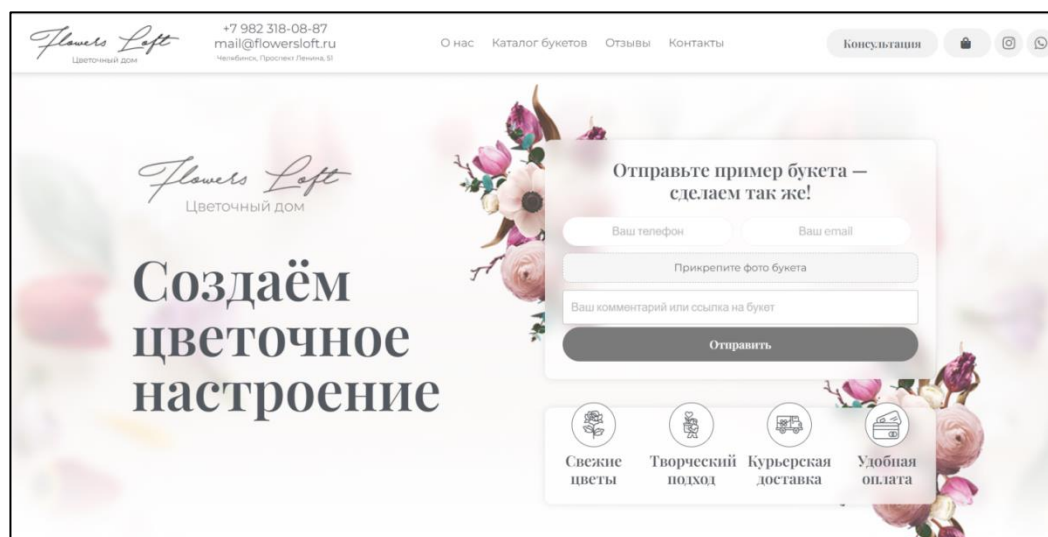


Рисунок 1 – Скриншот главной страницы сайта «Flowersloft»

Рассмотрим ещё одну популярную сеть цветочных магазинов «fantulpan.ru» [3]. Из достоинств можно выделить хорошую демонстрацию

выполненных цветочных композиций, возможность просмотра ассортимента не только по категориям, но и по цвету. Большой ассортимент товара и предоставляемых услуг. На сайте можно оформить заказ путем добавления товаров в корзину, есть возможность связаться с онлайн-консультантом. Имеются отзывы других клиентов.

После тестирования на различных устройствах проблем в адаптивности замечено не было. Сайт хорошо работает на разных размерах экрана.

Из недостатков сайта можно выделить медленную загрузку страниц и долгий отклик. Главная страница сайта предоставлена на рисунке 2.

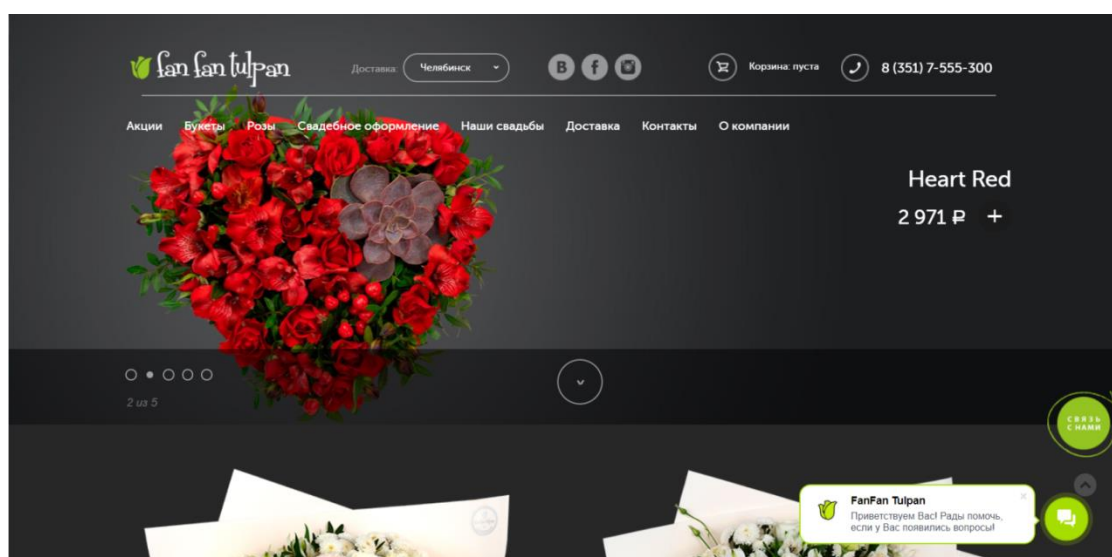


Рисунок 2 – Скриншот главной страницы сайта «fanfantulpan»

«Megaflowers.ru» [4] – популярная сеть цветочных магазинов и доставок по всему миру. Из достоинств можно выделить интуитивно понятный интерфейс, наличие акций на определенные виды товаров, возможность регулировать количество цветов в букете, представленном на сайте. К каждому букету есть рейтинг и отзывы. На сайте присутствует сортировка цветам, цене и популярности. Есть возможность заказа букета через корзину и связь с онлайн-консультантом.

Из недостатков сайта можно выделить небольшую проблему с адаптивностью. На некоторых размерах экрана смещается название и цена то-



вара, текст становится нечитаемым. Главная страница сайта предоставлена на рисунке 3.

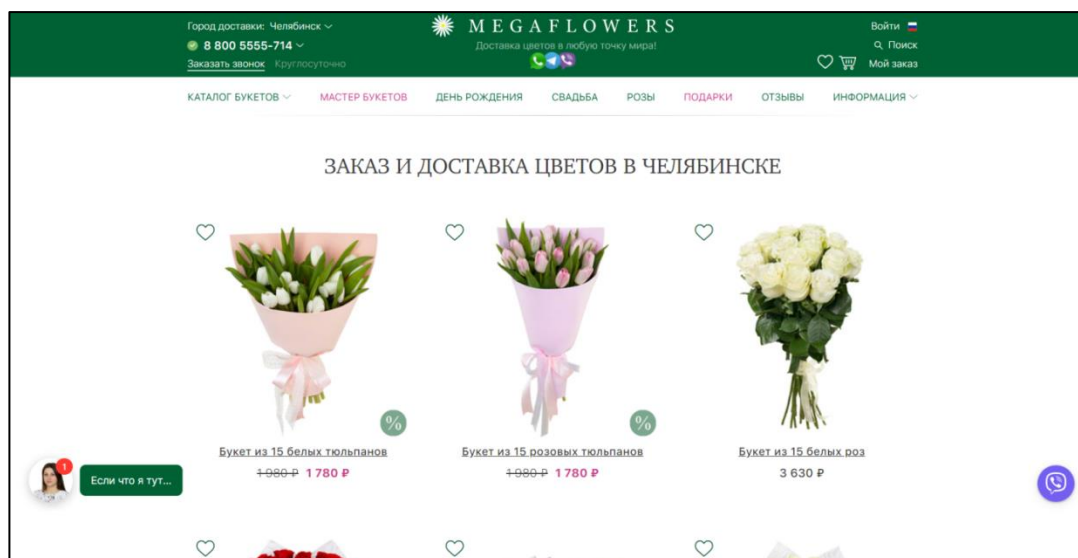


Рисунок 3 – Скриншот главной страницы сайты «Megaflowers»

В результате анализа аналогов были выделены основные задачи для реализации веб-сайта. Необходимо реализовать сортировку по различным параметрам и поиск по сайту для удобства нахождения пользователем нужного товара. Важным разделом сайта является корзина. Выявлено, что на подобных сайтах нет необходимости в создании профиля пользователя, так как вся необходимая информация указывается при заказе. Важно создать качественную адаптацию, т.к. в современном мире многие используют различные устройства для просмотра информации. Так же очень важной частью является интуитивный интерфейс и хорошо подобранные цвета сайта.

Для того чтобы избежать в разрабатываемом проекте выявленные недостатки аналогов необходимо обеспечить эффективность, надежность и масштабируемость приложения. Требуется разработать гибкую структуру и качественную оптимизацию для веб-сайта.

## **1.2. Средства разработки**

Для разработки дизайна веб-страниц был выбран сервис Figma. Для создания веб-сайта выбраны следующие программные средства: HTML, CSS, TypeScript, React, Redux Toolkit, Mock Api. В качестве редактора кода использован VSCode.

Figma [5] – графический редактор для создания прототипов сайтов и приложений.

HTML [6] («язык гипертекстовой разметки») – это код, который используется для структурирования и отображения веб-страницы и ее контента.

CSS [7] («каскадные таблицы стилей») – формальный язык описания внешнего вида страниц, написанного с использованием языка разметки.

TypeScript [8] – типизированный динамический язык программирования высокого уровня, используется для создания интерактивных веб-страниц и приложений.

React [9] – библиотека JavaScript с открытым кодом для создания внешних пользовательских интерфейсов.

Redux [10] – инструмент для управления состоянием данных и пользовательским интерфейсом в приложениях JavaScript с большим количеством сущностей.

Mock Api [11] – сервис для создания тестового RESTful API.

### **Выводы по первой главе**

В первой главе был произведен обзор и анализ схожих по концепции сайтов. Было выявлено, какая структура лучше подойдет для создаваемого сайта. Также определены инструменты для реализации проекта.

## **2. ПРОЕКТИРОВАНИЕ**

### **2.1. Требования к системе**

Перед разработкой программного продукта необходимо сформулировать требования к нему. Эти требования можно разделить на функциональные и нефункциональные. К функциональным требованиям относятся действия, которые система должна выполнять, т.е. это описание того, что нужно реализовать в процессе работы. Нефункциональные требования описывают как именно будет выполнена реализация и особенности эксплуатации.

#### **Функциональные требования к проектируемой системе**

1. Пользователь должен иметь возможность сортировать товары по популярности, цене и алфавиту.
2. Пользователь должен иметь возможность фильтровать товары по конкретному цветку.
3. Пользователь должен иметь возможность добавлять товары в корзину и удалять их.
4. Пользователь должен иметь возможность искать товар по названию.
5. Пользователь должен иметь возможность оформить заказ.
6. Веб-сайт должен быть адаптивным под разные размеры экрана.

#### **Нефункциональные требования к проектируемой системе**

1. Веб-сайт должен быть разработан с использованием HTML, CSS, TypeScript.
2. Веб-сайт должен быть разработан с помощью библиотек React, Redux Toolkit, React Redux.
3. Веб-сайт должен использовать Mock Api для хранения пользовательских данных.

## 2.2. Варианты использования системы

Разработанная диаграмма вариантов использования представлена на рисунке 4.

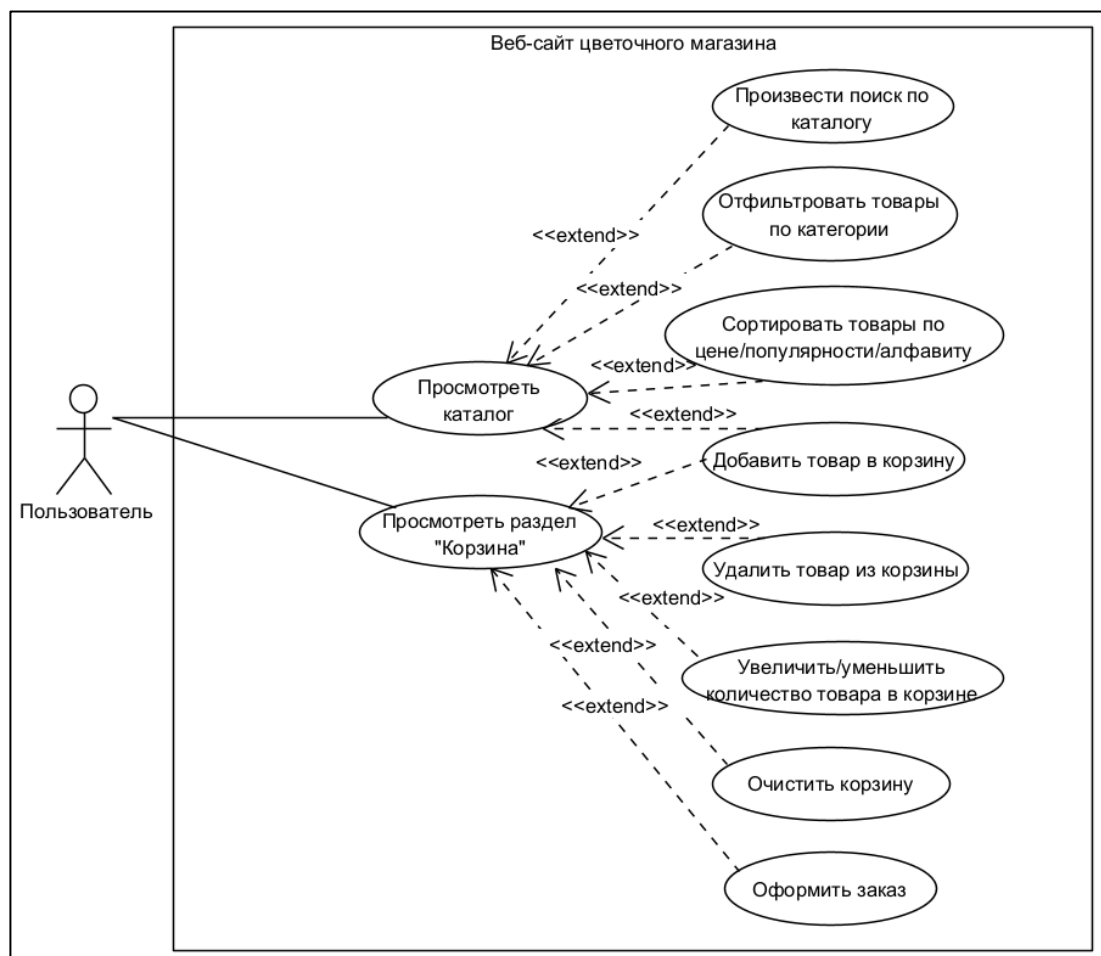


Рисунок 4 – Диаграмма вариантов использования

Основные актеры, взаимодействующие с системой.

На диаграмме представлен 1 актер – пользователь. Пользователем является клиент цветочного магазина, который посещает сайт либо с целью ознакомления с каталогом, либо с целью наполнения корзины.

Краткое описание вариантов использования.

1. Просмотреть каталог. Пользователь может посмотреть каталог всех товаров, которые предоставляет магазин.

2. Произвести поиск по каталогу. Пользователь может найти товар по названию через специальное поле поиска.

3. Отфильтровать товары по категории. Пользователь может отфильтровать товары по конкретному цветку.

4. Сортировать товары по цене/популярности/алфавиту. Пользователь может включить показ товаров по возрастанию или убыванию цены или популярности, а также показ в алфавитном или обратном алфавитном порядке.

5. Добавить товар в корзину. Пользователь может добавить необходимые товары в корзину.

6. Просмотреть раздел «Корзина». Пользователь может посмотреть свою корзину.

7. Удалить товар из корзины. Пользователь может удалить полностью конкретную позицию товара из корзины.

8. Увеличить/уменьшить количество товара в корзине. Пользователь может уменьшить или увеличить уже имеющийся в корзине товар на 1.

9. Очистить корзину. Пользователь может удалить все позиции товаров из корзины.

10. Оформить заказ. Пользователь может заполнить форму и отправить заказ.

Спецификация вариантов использования системы приведена в таблицах 1–10 приложения А.

## **2.3. Выбор архитектуры**

### **Flux архитектура [12]**

Для создания пользовательского интерфейса сайта была выбрана библиотека React. При работе с данным инструментом создатели библиотеки используют Flux архитектуру для управления потоком данных в приложении.

Flux делится на 4 основные части.

1. Действие (action) – объект, который описывает какое-либо действие в приложении.
2. Хранилище (store) – содержит состояния и логику приложения.
3. Диспетчер (dispatcher) – является посредником, получает действия и передает их всем подписанным на него хранилищам.
4. Представления (view) – стандартные компоненты React, визуальная часть приложения.

Любое изменение в UI должно быть обработано. Когда в приложении происходит какое-либо взаимодействие с представлением диспетчер отправляет действие в хранилище, оно меняет свое состояние и дает сигнал о том, что представление может перерисоваться. Визуальное представление архитектуры Flux представлено на рисунке 5.

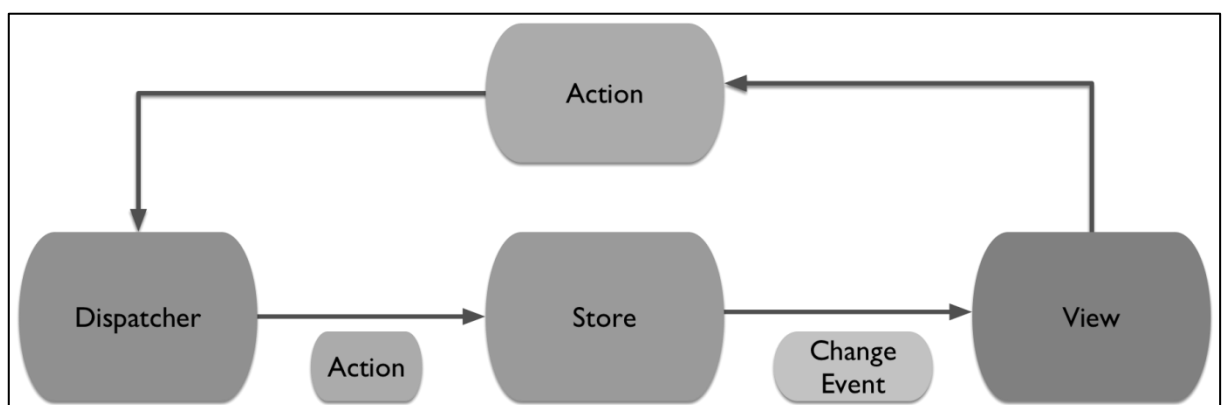


Рисунок 5 – Flux архитектура

### Single Page Application (SPA) [13]

Обычный сайт состоит из множества html-страниц и при переходе пользователя в какой-либо раздел на сайте браузер загружает новую страницу. Это достаточно медленный процесс, а пользователи стремятся получить информацию как можно быстрее.

Single Page Application (одностраничное приложение) – это всего одна html-страница в которую приходит много JS-кода. Скрипт работает на клиенте и динамически выводит необходимые данные (разметку). Даже

если идет переключение на другой раздел сайта новый html не подгружается. Скрипт ловит изменения в url адресе отправляет ajax-запрос на сервер, возвращаются данные и по ним js отрисовывает нужную разметку.

Преимущество рассмотренной архитектуры состоит в том, что при каком-либо изменении на сайте не обновляется вся страница, а обновляется только измененный элемент. За счет этого информация подгружается быстрее. Также из преимуществ SPA отмечают гибкость разработки и меньшую нагрузку на сервер.

## 2.4. Диаграмма компонентов

Диаграмма компонентов, представленная на рисунке 6, демонстрирует связи между функциональными элементами системы.

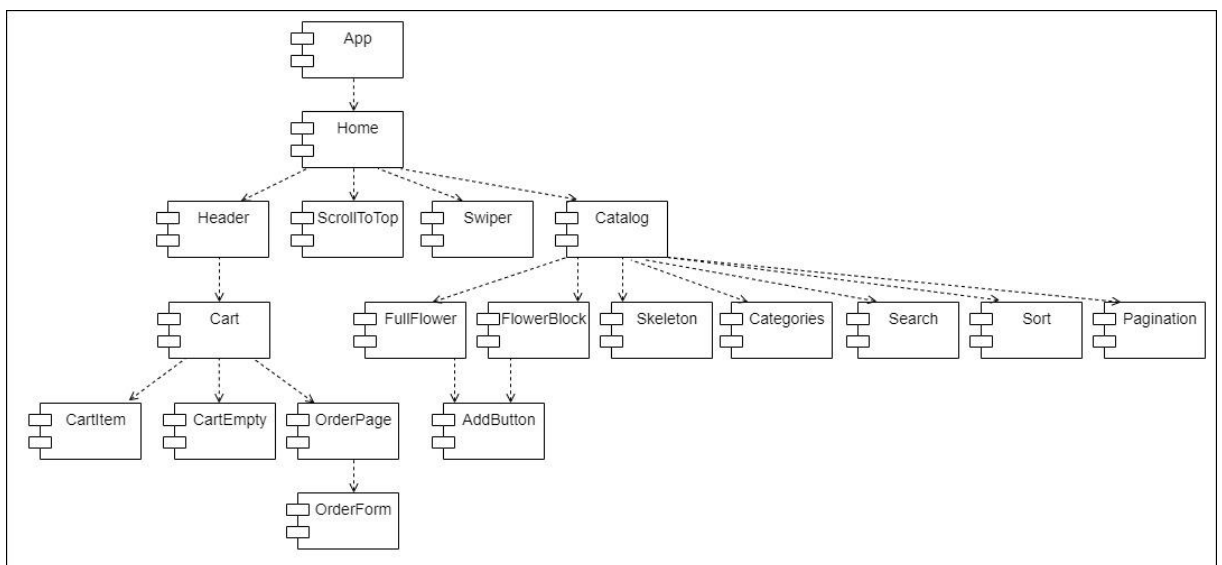


Рисунок 6 – Диаграмма компонентов

## 2.5. Диаграмма активности

Одной из основных задач сайта является возможность добавления товара в корзину. Диаграмма активности [14] описывает данный алгоритм (рисунок 7).

В начале, пользователь нажимает на кнопку добавления товара в корзину. Данное действие отправляется в хранилище. Система проверяет, есть ли объект с соответствующим идентификатором в массиве, который хранит все добавленные в корзину товары. Если такой объект есть, поле «количество» увеличивается на 1. В противном случае в массив добавляется новый объект и его количество устанавливается равным единице. Далее подсчитывается общая сумма товаров в корзине. После изменения состояния корзины соответствующие компоненты перерисовываются, добавленные товары отображаются в корзине.

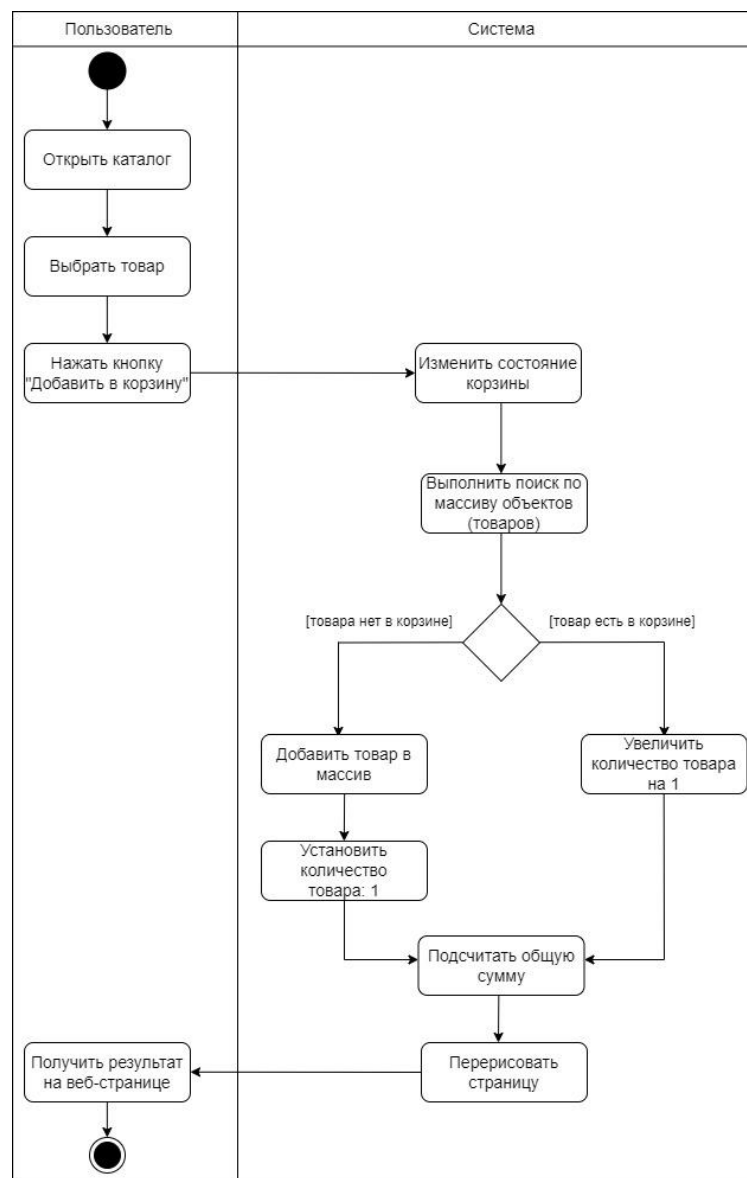


Рисунок 7 – Диаграмма активности



## 2.6. Проектирование дизайна веб-сайта

Сайта должен иметь качественный UX/UI дизайн. Цвета должны быть приятными и не вызывать раздражения у пользователя. Шрифт, его цвет и цвет фона выбираются по стандартам для того, чтобы прочтение текста не вызывало затруднений. Компоненты сайта должны быть расположены так, чтобы пользователь мог найти их интуитивно.

Основные цвета сайта были выбраны с помощью онлайн-приложения ColorScheme для правильного сочетания оттенков. С помощью сервиса Figma, на основе требований, был разработан макет сайта (рисунок 8).

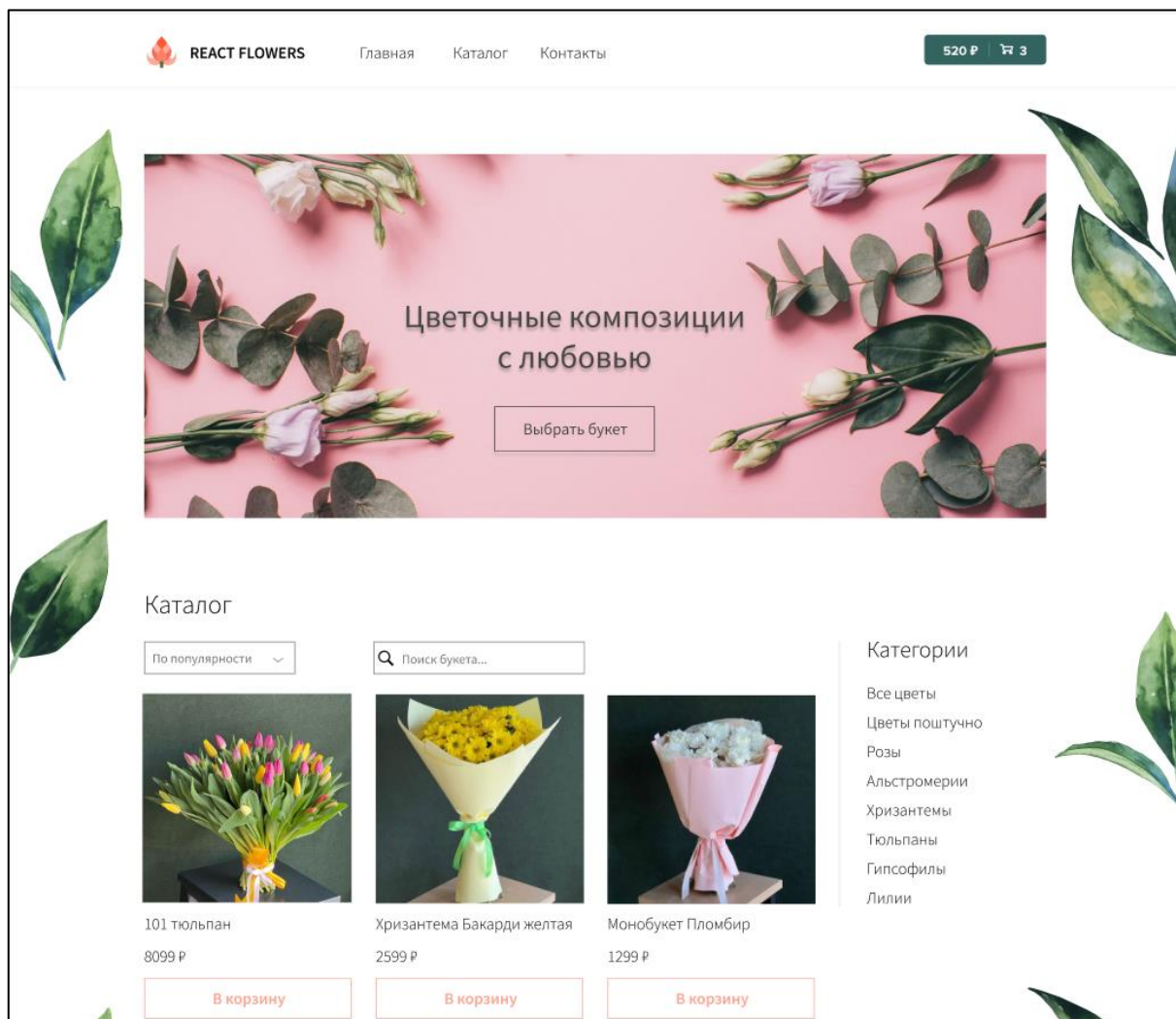


Рисунок 8 – Макет главной страницы сайта

### 3. РЕАЛИЗАЦИЯ

#### 3.1. Создание проекта

Для создания React-приложения [15] нужно прописать в консоли команду «create-react-app». Это действие создаст проект со всеми базовыми модулями. Перед началом разработки веб-сайта необходимо установить все необходимые пакеты, которые будут отвечать за состояния данных в приложении, а именно Redux Toolkit и React Redux. Для этого прописываем в консоли необходимые команды, установка происходит через пакетный менеджер, командой «npm install».

##### Создание хранилища

В папке проекта создадим глобальное хранилище – файл store.js. Он будет хранить состояния всего приложения. С помощью функции `configureStore()` скомбинируем все редьюсеры, каждый из них отвечает за конкретную часть приложения, то есть за состояния какого-либо раздела (листинг 1). Редьюсеры – это чистые функции, которые принимают состояния и отправленное хранилищу действие, а возвращают обновленное состояние в соответствии с действием. Для того, чтобы хранилище было доступно всем компонентам необходимо обернуть приложение в тег `<Provider>` и передать ему в качестве значения созданное хранилище.

##### Листинг 1 – Создание хранилища

```
import { configureStore } from '@reduxjs/toolkit';
import cartSlice from './slices/cartSlice';
import filterSlice from './slices/filterSlice';

export const store = configureStore({
  reducer: {
    filter: filterSlice,
    cart: cartSlice,
    flowers: flowersSlice,
  },
});
```

### 3.2. Создание каталога товаров

Для разработки каталога был создан компонент FlowerBlock.tsx, отрисовывающий карточку товара в соответствии с разработанным ранее макетом.

Для хранения состояний был создан слайс (часть хранилища) flowerSlice. Изначальные состояния данного элемента – пустой массив, предназначенный для хранения товаров и переменная для определения текущего статуса загрузки товаров со значением «loading», которая может принимать только одно из трех значений: loading, success, error.

Подготовка данных. Все элементы каталога хранятся на API сервисе. Для наполнения сайта товарами, данные были собраны и подготовлены в формате json, а после загружены на API. Каждый элемент содержит следующую информацию: id, название, цена, картинка, категория (содержит информацию о цветках в букете), рейтинг и описание. На рисунке 9 показана часть подготовленных данных.

```
[
  {
    "id": "1",
    "name": "Белые розы",
    "price": 4999,
    "image": "https://kupibuket74.ru/netcat_files/5/22/msg5246846366_68106.jpg",
    "category": [
      2
    ],
    "rating": 2,
    "description": "Букет из белых Эквадорских роз",
    "info": "-роза Эквадор - 50 см - 25 шт., \r\n-атласная лента - 2 м."
  },
  {
    "id": "2",
    "name": "101 тюльпан",
    "price": 8099,
    "image": "https://kupibuket74.ru/netcat_files/5/22/msg5246846366_64352_0.jpg",
    "category": [
      5
    ],
    "rating": 4,
    "description": "букет из 101 тюльпана в гармоничном миксовом сочетании",
    "info": "-тюльпан - 101 шт., \r\n-лента атласная - 1м."
  },
  {
    "id": "3",
    "name": "Хризантема Бакарди",
    "price": 2599,
    "image": "https://kupibuket74.ru/netcat_files/5/22/msg5246846366_68749.jpg",
    "category": [
      3
    ],
    "rating": 3,
    "description": "Букет из жёлтых хризантем",
    "info": "-хризантема сорта Бакарди- 9 шт., \r\n-фоамиран - 1,5 м., \r\n-лента атласная - 1 м."
  },
]
```

Рисунок 9 – Данные товаров

Загрузка товаров с API. Для реализации загрузки данных используется библиотека `axios`, она позволяет отправлять запросы к заданной точке. Чтобы получить данные с сервера необходимо сделать `get`-запрос на `url`-адрес. Для корректной работы запросов необходимо применение хука `useEffect()`, так как запрос является побочным эффектом (`side effect`), его нельзя выполнить в основной части функциональной компоненты. В качестве параметров в хук передается функция, то есть сам запрос и список зависимостей, то есть те переменные, при изменении которых запрос будет делаться вновь (листинг 2).

### Листинг 2 – Код запроса

```
useEffect(() => {
  setIsLoading(true);

  const order = sortType.includes('-') ? 'asc' : 'desc';
  const sortBy = sortType.replace('-', '');
  const category = categoryId > 0 ? `category=${categoryId}` : '';
  const search = searchValue ? `&search=${searchValue}` : '';

  axios
    .get(
      `https://63c2c97de3abfa59bdb327f5.mockapi.io/items?page=${currentPage}&limit=4&${category}&sortBy=${sortBy}&order=${order}${search}`,
    )
    .then((response) => {
      setItems(response.data);
      setIsLoading(false);
    });
}, [categoryId, sortType, searchValue, currentPage]);
```

После того, как данные были получены, они записываются в заранее созданный массив, который хранится в состояниях приложения. Для того, чтобы трансформировать массив полученных значений в необходимый вид используется метод `map`, с его помощью каждый объект массива преобразуется в `tsx`-компонент, созданный ранее. Далее требуется вставить информацию, хранящуюся в объекте, в необходимое место в разметке. На листинге 3 представлен компонент карточки товара.

### Листинг 3 – Компонент карточки товара

```
const FlowerBlock: React.FC<FlowerBlockProps> = ({ id, name, price, image
}) => {
  return (
    <div className={style.item}>
      <Link to={`/flower/${id}`}>
        <img src={image} />
        <p className={style.title}>{name}</p>
      </Link>
      <div className={style.flower_bottom}>
        <div className={style.price}>
          <p>{price} ₺</p>
        </div>
        <div>
          <AddButton id={id} name={name} price={price} image={image} />
        </div>
      </div>
    </div>
  );
};

export default FlowerBlock;
```

Данные с сервера подгружаются не моментально, соответственно, когда пользователь только открывает сайт есть вероятность того, что он увидит пустой каталог. Для удобства пользования сайтом был добавлен «скелетон». Скелетные экраны – это пустые страницы, которые постепенно заполняются контентом, например, текстом и изображениями. Это один из способов показать пользователю, что сайт загружается и скоро будет показан контент. Скелетон представляет собой очень упрощенную версию сайта и дает примерное представление об интерфейсе.

Для реализации скелетона была создана переменная `setIsLoading` с изначальным значением `true`. После того как приходит ответ от сервера, в запросе, значение переменной меняется на `false` (листинг 2). Через тернарный оператор проверяем идет ли сейчас загрузка и, в зависимости значения переменной, показываем либо компонент скелетона, либо массив товаров. Вид сайта при загрузке каталога представлен на рисунке 10. Скелетон анимирован, пока идет загрузка карточки переливаются.

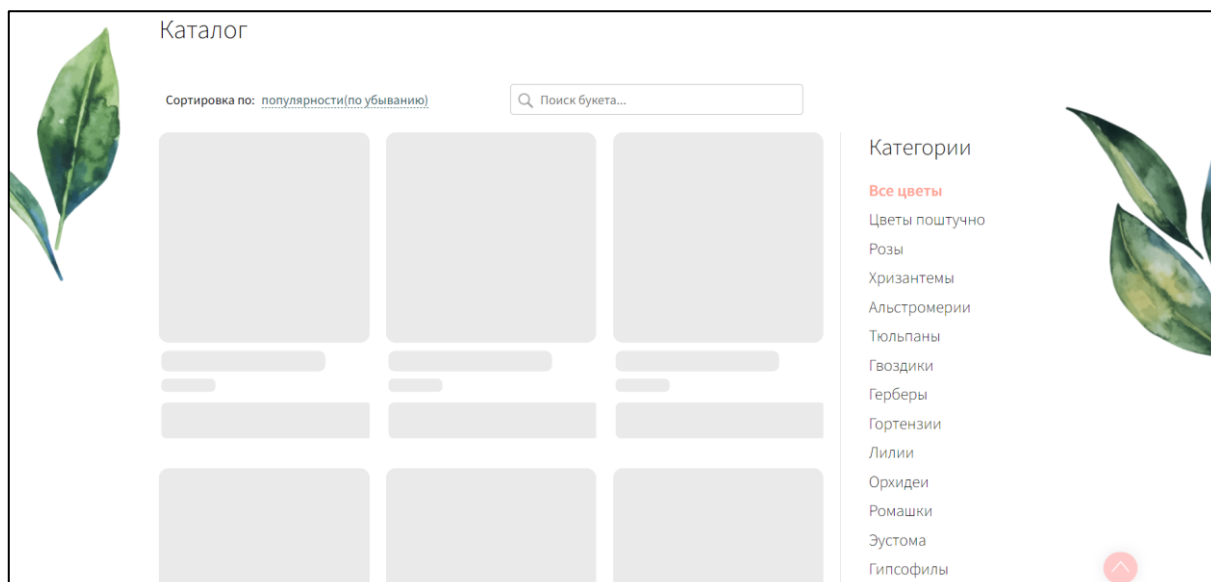


Рисунок 10 – Загрузка каталога

### 3.3. Пагинация, сортировка, фильтрация и поиск товаров

Для создания пагинации (постраничного вывода товаров), сортировки, фильтрации и поиска был создан слайс (часть хранилища) `filterSlice` (листинг 4). В переменной `initialState` содержатся изначальные состояния объектов, которые будут отображаться на сайте, пока пользователь не совершит какое-либо действие ведущее к изменению состояний. Также слайс содержит все необходимые редьюсеры и экспортирует их.

#### Листинг 4 – Слайс `filterSlice`

```
import { createSlice } from '@reduxjs/toolkit';

const initialState = {
  categoryId: 0,
  sort: { name: 'популярности(по возрастанию)', sortProperty: 'rating' },
  currentPage: 1,
};

export const filterSlice = createSlice({
  name: 'filter',
  initialState,
  reducers: {
    setCategoryId: (state, action) => {
      state.categoryId = action.payload;
    },
    setSort: (state, action) => {
      state.sort = action.payload;
    },
    setCurrentPage: (state, action) => {
```

```

        state.currentPage = action.payload;
      },
    },
  });

export const { setCategoryId, setSort, setCurrentPage } = filterSlice.actions;

export default filterSlice.reducer;

```

## Пагинация

Пагинация реализована с помощью библиотеки `react-pagination`. Необходимо установить какое количество элементов будет выводиться на странице и сколько всего будет страниц. По событию на клик по кнопке переключения страниц действие отправляется в хранилище, состояние меняется (`currentPage`) и дает сигнал о том, что компонент может перерисоваться, по `flux` архитектуре. Данная переменная содержится в списке зависимостей хука `useEffect()`, поэтому при ее изменении запрос делается снова, подгружая новую группу товаров.

## Сортировка

На сайте есть возможность сортировки товаров по возрастанию и убыванию цены, популярности и алфавита. Переключение типа сортировки по клику реализовано аналогично переключению страниц в пагинации. Все возможные типы сортировки хранятся в массиве объектов. Сам объект содержит в себе название типа и свойство сортировки, определяющее возрастание или убывание. После выбора типа сортировки пользователем происходит запрос и сервер возвращает данные в необходимом порядке.

## Фильтрация

Фильтрация по виду цветка реализована за счет того, что в объекте товара содержится его категория. Переключение между цветами реализовано аналогично предыдущим пунктам. При выборе определенного цветка происходит запрос на показ товаров, содержащих только выбранную категорию. Сервер возвращает необходимые данные.

## Поиск

Для поиска по товарам создано специальное поле ввода. После каждого ввода символа делается запрос на сервер и проверяется на соответствие названию или цветку в составе букета. Но для того, чтобы сервер не перегружался от постоянных запросов сделана небольшая задержка в пол секунды, это значит, что запрос будет происходить только тогда, когда пользователь не набирает символы хотя бы пол секунды. Результат поиска показан на рисунке 11.

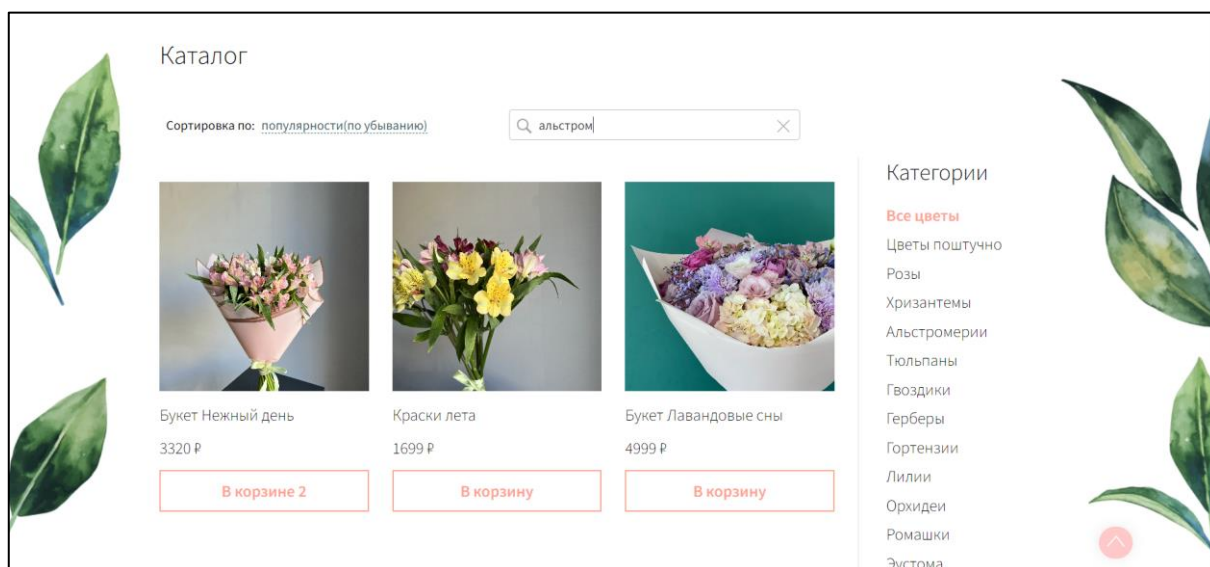


Рисунок 11 – Результат поиска

### 3.4. Разработка корзины

Для разработки корзины на сайте как и в предыдущих случаях был создан отдельный слайс (часть хранилища) для хранения состояний корзины. Изначальные состояния данного элемента – массив для хранения товаров, добавленных в корзину и их общая стоимость. Подсчет общей стоимости вынесен в отдельную функцию для удобства дальнейшего использования. Для корректной работы корзины в хранилище добавлены следующие действия, ведущие к изменению состояний: добавить 1 товар, удалить 1 товар, удалить товар полностью, очистить корзину полностью.



Также для возможности взаимодействия пользователя с корзиной был создан функциональный компонент `Cart.tsx`, который отображает товары в корзине, их количество и общую стоимость, а также кнопку для перехода на страницу оформления заказа (рисунок 12). Он использует хук `useSelector` для получения товаров и общей стоимости из состояния корзины, а затем отображает эти данные на странице, а также хук `useDispatch()` для отправки действий в хранилище. В данном компоненте есть проверка пуста ли корзина, если это так, то на странице будет отображаться другой компонент – `CartEmpty.tsx`. Часть реализации корзины представлена в листинге 5.

### Листинг 5 – Реализация корзины

```
export default function Cart() {
  const { items, totalPrice } = useSelector(selectCart);
  const dispatch = useDispatch();

  const onClickClear = () => {
    dispatch(clearItems());
  };

  if (!totalPrice) {
    return <CartEmpty />;
  }

  return (
    <div className="container">
      <p className="cart_title">Корзина</p>
      <div className={style.clear_btn}>
        <button onClick={onClickClear} className={style.clear}>
          Очистить корзину
        </button>
      </div>
      <div>
        {items.map((item) => (
          <CartItem key={item.id} {...item} />
        ))}
        <div className={style.root}>
          <p className={style.total}>Итого: {totalPrice} Р</p>
          <Link to="/order">
            <button className={style.checkout}>Оформить заказ</button>
          </Link>
        </div>
      </div>
    </div>
  );
}
```

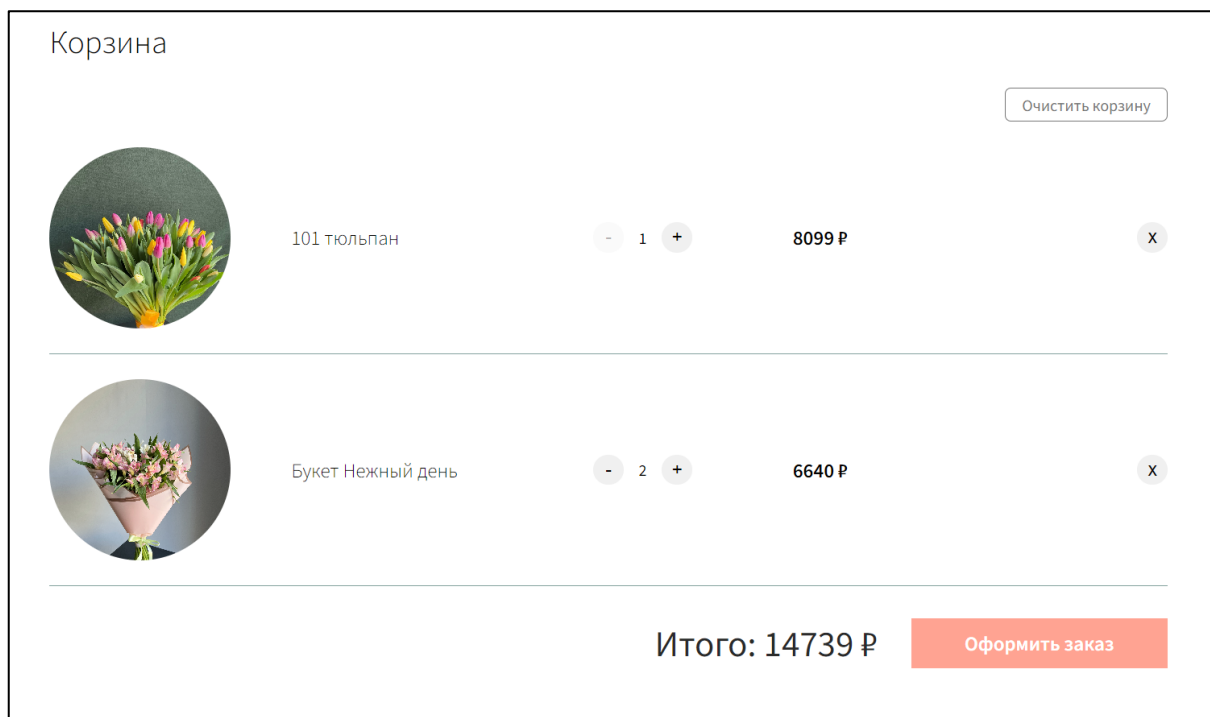


Рисунок 12 – Корзина товаров

Для того, чтобы содержимое корзины не исчезало после обновления страницы было создано локальное хранилище (`localStorage`). Локальное хранилище позволяет хранить пары ключ-значение в браузере и сохраняет их после перезапуска страницы или браузера. Теперь при каждом изменении состава корзины данные сохраняются в локальное хранилище (листинг 6), а изначальные состояния корзины достаются из него по ключу.

#### Листинг 6 – Сохранение данных в локальное хранилище

```
React.useEffect(() => {
  if (isMounted.current) {
    const json = JSON.stringify(items);
    localStorage.setItem('cart', json);
  }
  isMounted.current = true;
}, [items]);
```

### 3.5. Разработка страницы товара

Подробную информацию о букете можно получить кликнув на один из элементов каталога. В таком случае пользователь перейдет на страницу

конкретного товара. Для реализации был создан функциональный компонент FullFlower.tsx. Чтобы получить необходимую информацию нужно сделать запрос на сервер по конкретному id. Реализация выполнена аналогично разработке каталога товаров. Страница товара представлена на рисунке 13.

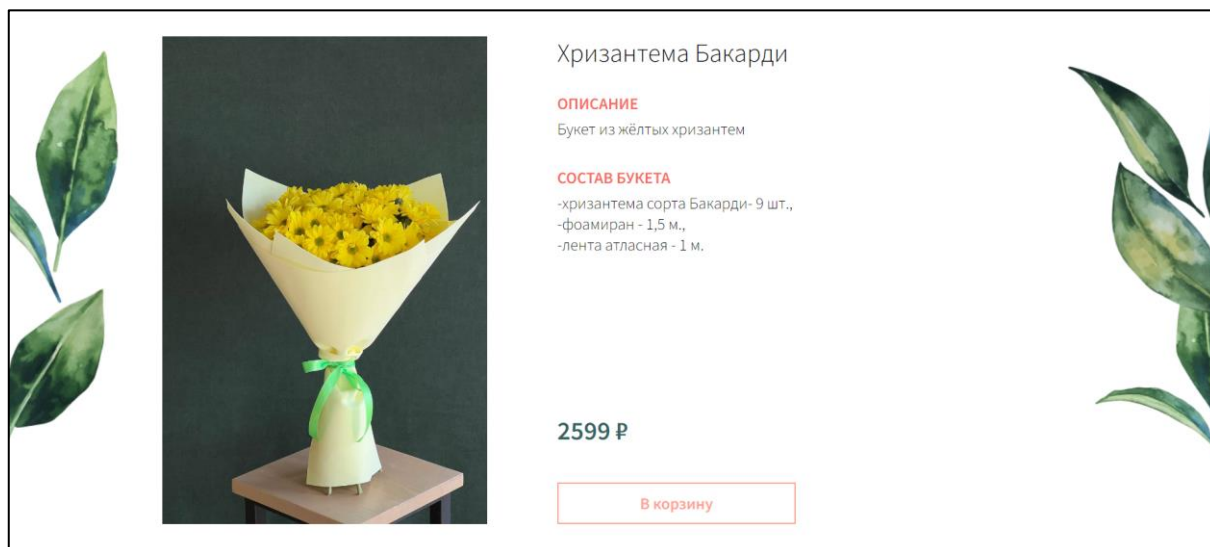


Рисунок 13 – Страница товара

### 3.6. Создание страницы оформления заказа

Из корзины (не пустой) есть возможность перейти на страницу оформления заказа. Оформление заказа подразумевает заполнение формы личной информацией (ФИО, номер телефона, email), а также информацией о доставке (адрес доставки, дата, время). Для создания формы была использована библиотека formik. Она позволяет управлять состоянием формы, валидировать ее и отправлять. Также для валидации была использована библиотека уир. Все поля проходят проверку на корректность введенных данных, присутствуют обязательные для заполнения поля. Если данные введены некорректно или отсутствуют появляется соответствующее сообщение об ошибке, а кнопка для отправки формы заблокирована (рисунок 14).

В данный момент сайт не работает с реальной компанией, поэтому сейчас информация о заказе просто выводится в консоль для понимания того, что форма работает корректно. При необходимости можно подключить отправку формы на сторонний ресурс.

## Оформление заказа

### Контактные данные

Александра

Телефон

Обязательное поле

аа

Некорректный E-mail

### Состав заказа

101 тюльпан x1 8099 Р

Букет Нежный день x2 6640 Р

### Подробности доставки

Адрес доставки (улица, дом, квартира)

дд.мм.гггг

Время доставки

☐ Согласен на обработку персональных данных

Оформить заказ

Итого: 14739 Р

Рисунок 14 – Страница оформления заказа

### 3.7. Разработка слайдера, прокрутки

#### Якорная прокрутка

Якорная прокрутка на сайте реализована для удобства пользования. Если пользователь пролистает страницу более чем на 100 пикселей вниз в углу появится специальная кнопка, по нажатию на которую можно вернуться в начало страницы.

## Слайдер

На главной странице сайта расположена карусель изображений. Слайдер реализован с помощью библиотеки Swiper, которая позволяет управлять изображениями в карусели и способом их пролистывания. Слайдер добавлен с целью призыва перейти в каталог товаров и для более приятного внешнего вида страницы.

### 3.8. Адаптивная верстка

Адаптивность на сайте реализована при помощи медиа-запросов и CSS-свойств. При использовании медиа-запросов контент на сайте перестраивается, когда ширина экрана достигает определенной точки. Так же некоторым элементам сайта заданы свойства flex-компонентов. Это способствует адаптивному распределению контента. Таким образом, контент отображается качественно и остается читабельным. Один из медиа-запросов представлен в листинге 7. Версии сайта для различных типов устройств представлены в приложении Б.

#### Листинг 7 – Медиа-запрос для экранов не более 991 пикселя

```
@media screen and (max-width: 991px) {  
  .container {  
    width: 720px;  
    margin: 0 auto;  
  }  
  .content_items {  
    grid-template-columns: repeat(2, 1fr);  
    grid-column-gap: 29px;  
    grid-row-gap: 29px;  
  }  
  .content_top {  
    display: block;  
  }  
  .catalog_title,  
  .cart_title,  
  .order_title {  
    margin: 30px 0 20px 0;  
  }  
}
```

## 4. ТЕСТИРОВАНИЕ

### 4.1. Функциональное тестирование

Функциональное тестирование заключается в проверке системы на соответствие заданных в требованиях функциональных задач. Результаты тестирования приведены в таблице 1.

Таблица 1 – Результаты функционального тестирования

№	Название теста	Действия	Ожидаемый результат	Тест пройден?
1	Проверка открытия каталога	Открыть веб-сайт и перейти в раздел «Каталог»	Отображаются все доступные для заказа товары	Да
2	Проверка поиска товаров	Ввести в поисковую строку название букета или название цветка в составе букета	Отображаются все товары, в названии или в составе которых найдено введенное слово	Да
3	Проверка фильтрации товаров	Выбрать и нажать на одну из доступных категорий товаров	Отображаются все товары, в составе которых есть цветок соответствующей категории	Да
4	Проверка сортировки товаров	Нажать на текущий тип сортировки и в выпадающем списке выбрать один из типов сортировки товаров	Товары отображаются в соответствии с выбранным типом сортировки	Да
5	Проверка пагинации	В каталоге товаров нажать на номер страницы, на которую нужно переключиться. Либо нажать на кнопку вперед/назад для переключения на следующую/предыдущую страницу.	Отображается группа товаров, в соответствии с номером выбранной страницы	Да

Продолжение таблицы 1

№	Название теста	Действия	Ожидаемый результат	Тест пройден?
6	Проверка добавления товара в корзину	На карточке товара нажать на кнопку «В корзину»	Кнопка «В корзину» заменяется. Выбранный товар отображается в корзине. Сумма товара добавлена к общей сумме корзины. Количество и общая сумма товаров отображается на пиктограмме корзины корректно. Данные сохраняются в локальное хранилище	Да
7	Проверка перехода на страницу корзины	Нажать на пиктограмму корзины в шапке сайта	Отображается интерфейс корзины товаров. В корзине присутствуют все добавленные ранее товары и их общая стоимость	Да
8	Проверка уменьшения/увеличения товара в корзине	Добавить хотя бы 1 товар в корзину. Находясь в корзине нажать на кнопку «минус»/«плюс» возле одного из добавленных товаров	Счетчик товара изменяется в меньшую/большую сторону на 1. Если количество товара единица, кнопка уменьшения недоступна. Данные сохраняются в локальное хранилище	Да
9	Проверка удаления товара из корзины	Добавить хотя бы 1 товар в корзину. Находясь в корзине нажать на кнопку «удалить» возле одного из добавленных товаров	Товар полностью удаляется из корзины. Информация сохраняется в локальное хранилище	Да
10	Проверка очистки корзины	Добавить хотя бы 1 товар в корзину. Находясь в корзине, нажать на кнопку «Очистить корзину»	Товары удалены из корзины. Отображается страница пустой корзины. Информация сохраняется в локальное хранилище	Да

№	Название теста	Действия	Ожидаемый результат	Тест пройден?
11	Проверка перехода на страницу оформления заказа	Добавить хотя бы 1 товар в корзину. Находясь в корзине, нажать на кнопку «Оформить заказ»	Отображается форма для заполнения личными данными и все добавленные в корзину товары	Да
12	Проверка валидации формы	Заполнить поля личными данными и информацией о доставке	Если все поля заполнены корректно, кнопка для отправки заказа доступна. Если поля заполнены некорректно, появляется соответствующее сообщение, кнопка отправки заказа заблокирована	Да
13	Проверка оформления заказа	Заполнить все поля корректными данными и нажать на кнопку «Отправить заказ»	Отображается сообщение об оформлении заказа. Информация о заказе выведена в консоль	Да

## 4.2. Тестирование верстки

Тестирование интерфейса было проведено в различных браузерах и на разных размерах экрана.

Тест проводился в следующих браузерах:

- Opera;
- Google Chrome;
- Microsoft Edge;
- Safari.

Интерфейс пользователя во всех перечисленных браузерах остается идентичным, весь контент остается на своих местах. Также было проведено тестирование верстки на адаптивность в разных браузерах. При увеличении и уменьшении размера экрана контент адаптируется под устройство. Тесты успешно пройдены.



## **ЗАКЛЮЧЕНИЕ**

В рамках данной работы был спроектирован и реализован фронтенд для веб-сайта цветочного магазина на React. Были решены следующие задачи.

1. Проведен обзор аналогичных проектов, что позволило изучить существующие подходы в области разработки веб-сайтов для цветочных магазинов. Выявлены основные требования, которые должны быть реализованы в данном проекте.

2. Были выбраны подходящие средства разработки, учитывая требования проекта и его масштаб.

3. Определены функциональные и нефункциональные требования, которые обеспечивали плавную работу веб-сайта и удовлетворение потребностей пользователей.

4. Была выбрана подходящая архитектура приложения.

5. Разработаны макеты сайта, учитывая цветовую гамму, шрифты и т.п., которые создавали привлекательный и согласованный внешний вид сайта. Были учтены принципы юзабилити, чтобы обеспечить удобство использования для пользователей.

6. Разработан веб-сайт в соответствии с поставленными ранее требованиями.

7. Проведено функциональное тестирование и тестирование верстки, что позволило убедиться в соответствии реализованного проекта поставленным требованиям.

## ЛИТЕРАТУРА

1. Websiterating. [Электронный ресурс] URL: <https://www.websiterating.com/ru/research/internet-statistics-facts/> (дата обращения: 25.01.2023 г.).
2. Веб-сайт компании «FlowersLoft». [Электронный ресурс] URL: <https://flowersloft.ru> (дата обращения: 25.01.2023 г.).
3. Веб-сайт компании «fanfantulpan». [Электронный ресурс] URL: <https://fanfantulpan.ru> (дата обращения: 25.01.2023 г.).
4. Веб-сайт компании «Megaflowers». [Электронный ресурс] URL: <https://cheljabinsk.megaflowers.ru> (дата обращения: 25.01.2023 г.).
5. Figma. [Электронный ресурс] URL: <https://www.figma.com> (дата обращения: 02.02.2023 г.).
6. Основы HTML. [Электронный ресурс] URL: <https://htmlacademy.ru/courses/297> (дата обращения: 02.02.2023 г.).
7. CSS справочник. [Электронный ресурс] URL: <http://htmlbook.ru/css> (дата обращения: 02.02.2023 г.).
8. JavaScript documentation. [Электронный ресурс] URL: <https://developer.mozilla.org/en-US/docs/Web/JavaScript> (дата обращения: 09.02.2023 г.).
9. React documentation. [Электронный ресурс] URL: <https://reactjs.org/docs/getting-started.html> (дата обращения: 10.02.2023 г.).
10. Redux documentation. [Электронный ресурс] URL: <https://redux-toolkit.js.org> (дата обращения: 10.02.2023 г.).
11. Mock Api documentation. [Электронный ресурс] URL: <https://mockapi.io/docs> (дата обращения: 10.02.2023 г.).
12. Flux documentation. [Электронный ресурс] URL: <https://facebook.github.io/flux/> (дата обращения: 19.02.2023 г.).

13. SPA. [Электронный ресурс] URL: <https://learn.microsoft.com/en-us/aspnet/core/client-side/spa/intro?view=aspnetcore-8.0> (дата обращения: 20.03.2023 г.).

14. Петрова, О. Б. Разработка и анализ требований проектирования программного обеспечения: практикум: учебное пособие // Санкт-Петербург: СПбГУТ им. М.А. Бонч-Бруевича, 2022. – 37 с.

15. Мардан А. React быстро. Веб-приложения на React, JSX, Redux и GraphQL. – СПб: Питер, 2019. – 560 с.

## ПРИЛОЖЕНИЯ

### Приложение А. Спецификация вариантов использования

Спецификация вариантов использования (ВИ) системы приведена в таблицах 1–10.

Таблица 1 – Спецификация ВИ «Просмотреть каталог»

Прецедент: Просмотреть каталог
ID: 1
Краткое описание: Просмотр представленного каталога цветов.
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: отсутствуют
Основной поток: 1. Прецедент начинается, когда пользователь переходит в раздел «Каталог». 2. Пользователь просматривает каталог.
Постусловия: 1. Пользователь просмотрел каталог.
Альтернативные потоки: Нет

Таблица 2 – Спецификация ВИ «Произвести поиск по каталогу»

Прецедент: Произвести поиск по каталогу
ID: 2
Краткое описание: Поиск товаров в каталоге по названию.
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: отсутствуют
Основной поток: 1. Прецедент начинается, когда пользователь ставит фокус на поле ввода для поиска. 2. Пользователь печатает необходимые символы. 3. Система производит поиск товаров.
Постусловия: 1. Пользователь получил результат поиска.
Альтернативные потоки: Нет

Таблица 3 – Спецификация ВИ «Отфильтровать товары по категории»

Прецедент: Отфильтровать товары по категории
ID: 3
Краткое описание: Показ позиций товаров, содержащих определенный цветок.
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: отсутствуют
Основной поток: 1. Прецедент начинается, когда пользователь переходит в раздел «Каталог». 2. Пользователь выбирает и кликает по необходимой категории. 3. Система производит фильтрацию товаров.
Постусловия: 1. Пользователь получил результат фильтрации.
Альтернативные потоки: Нет

Таблица 4 – Спецификация ВИ «Сортировать товары по цене/алфавиту/популярности»

Прецедент: Сортировать товары по цене/популярности/алфавиту
ID: 4
Краткое описание: Показ товаров по возрастанию или убыванию цены/популярности/алфавита.
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: отсутствуют
Основной поток: 1. Прецедент начинается, когда пользователь переходит в раздел «Каталог». 2. Пользователь кликает на элемент «Сортировать по». 3. Пользователь выбирает и кликает на необходимый вид сортировки. 4. Система производит сортировку товаров.
Постусловия: 1. Пользователь получил отсортированный каталог.
Альтернативные потоки: Нет

Таблица 5 – Спецификация ВИ «Добавить товар в корзину»

Прецедент: Добавить товар в корзину
ID: 5
Краткое описание: Перенос товара в корзину.
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: отсутствуют
Основной поток: 1. Прецедент начинается, когда пользователь переходит в раздел «Каталог». 2. Пользователь выбирает товар и нажимает на кнопку «Добавить». 3. Система добавляет товар в корзину.
Постусловия: 1. Товар отображается в корзине.
Альтернативные потоки: Нет

Таблица 6 – Спецификация ВИ «Посмотреть раздел «Корзина»»

Прецедент: Посмотреть раздел «Корзина»
ID: 6
Краткое описание: Просмотр корзины Пользователя.
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: отсутствуют
Основной поток: 1. Прецедент начинается, когда пользователь переходит в раздел «Корзина». 2. Пользователь просматривает свою корзину.
Постусловия: 1. Пользователь просмотрел корзину.
Альтернативные потоки: Нет

Таблица 7 – Спецификация ВИ «Удалить товар из корзины»

Прецедент: Удалить товар из корзины
ID: 7
Краткое описание: Удаление позиции товара из корзины.
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: 1. В корзине имеется хотя бы 1 позиция товара.
Основной поток: 1. Прецедент начинается, когда пользователь переходит в раздел «Корзина». 2. Пользователь выбирает товар для удаления и нажимает на соответствующую кнопку 3. Система удаляет позицию товара.
Постусловия: 1. Товар больше не содержится в корзине.
Альтернативные потоки: Нет

Таблица 8 – Спецификация ВИ «Увеличить/уменьшить количество товара в корзине»

Прецедент: Увеличить/уменьшить количество товара в корзине
ID: 8
Краткое описание: Увеличить или уменьшить количество определенного товара в корзине.
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: 1. В корзине имеется хотя бы 1 позиция товара.
Основной поток: 1. Прецедент начинается, когда пользователь переходит в раздел «Корзина». 2. Пользователь выбирает товар для редактирования и нажимает на соответствующую кнопку 3. Система уменьшает/увеличивает количество определенного товара на 1.
Постусловия: 1. Количество товара в корзине изменено.
Альтернативные потоки: Нет

Таблица 9 – Спецификация ВИ «Очистить корзину»

Прецедент: Очистить корзину
ID: 9
Краткое описание: Удалить все товары из корзины.
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: 1. В корзине имеется хотя бы 1 позиция товара.
Основной поток: 1. Прецедент начинается, когда пользователь переходит в раздел «Корзина». 2. Пользователь нажимает на кнопку «Очистить корзину». 3. Система удаляет все позиции из корзины.
Постусловия: 1. Корзина пуста.
Альтернативные потоки: Нет

Таблица 10 – Спецификация ВИ «Оформить заказ»

Прецедент: Оформить заказ
ID: 10
Краткое описание: Отправить заказ на доставку
Главные актеры: Пользователь
Второстепенные актеры: Нет
Предусловия: 1. В корзине имеется хотя бы 1 позиция товара.
Основной поток: 1. Прецедент начинается, когда пользователь переходит в раздел «Корзина». 2. Пользователь нажимает на кнопку «Оформить заказ». 3. Пользователь заполняет все необходимые поля. 4. Пользователь нажимает на кнопку «Отправить заказ». 5. Система считывает данные с полей и сохраняет их.
Постусловия: 1. Данные по заказу сохранены.
Альтернативные потоки: Нет



## Приложение Б. Адаптивная верстка

На рисунках 1–6 приведены скриншоты веб-сайта на различных размерах экрана.

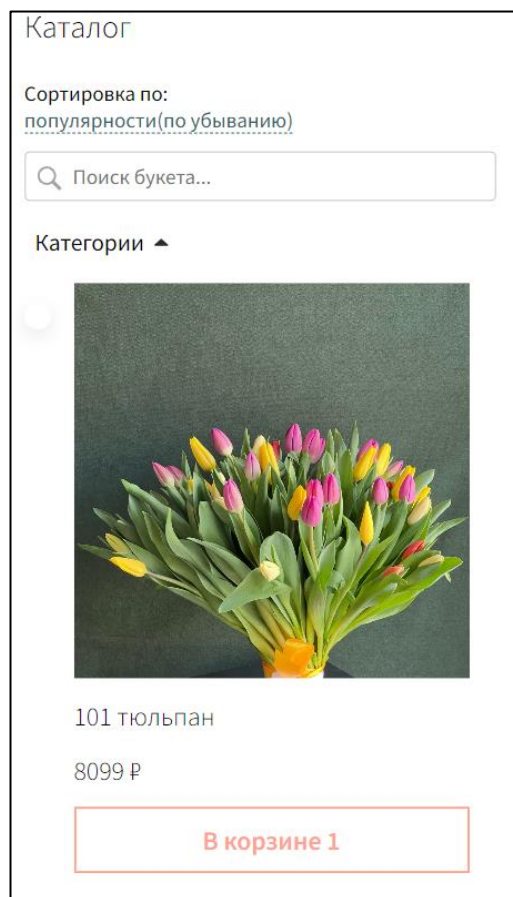


Рисунок 1 – Каталог товаров на мобильной версии сайта

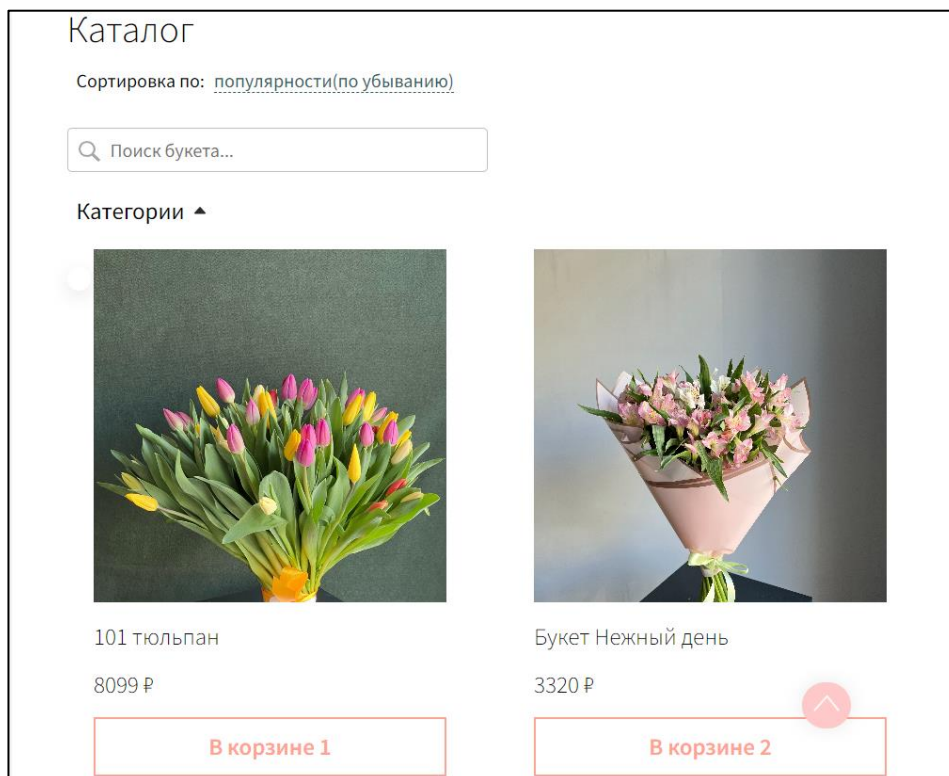


Рисунок 2 – Каталог товаров на планшетной версии сайта

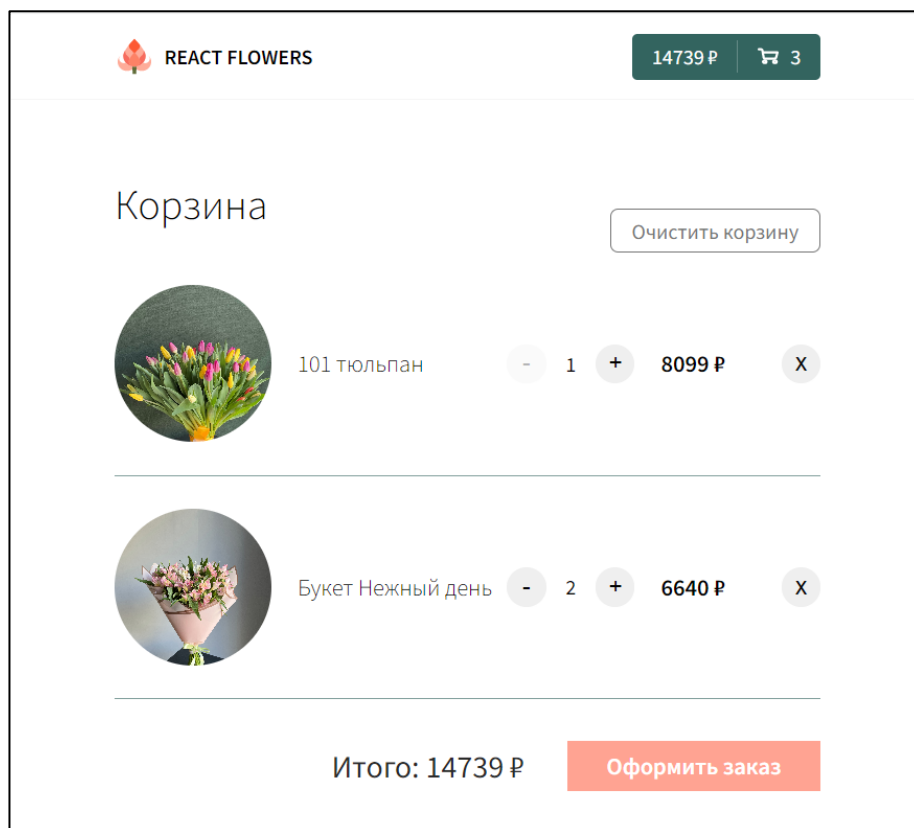


Рисунок 3 – Корзина на планшетной версии сайта

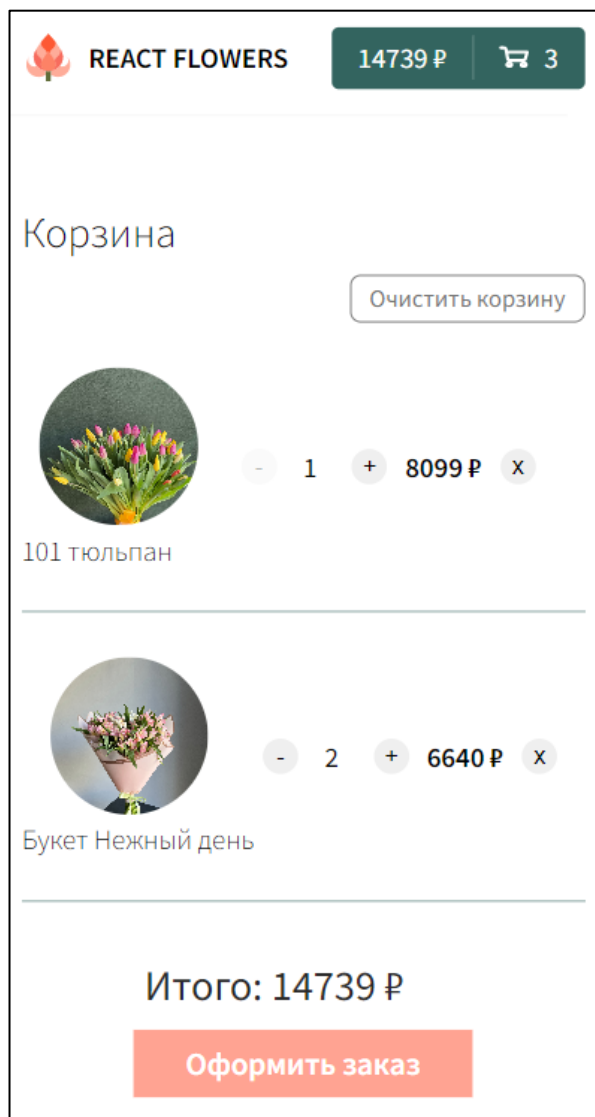


Рисунок 4 – Корзина на мобильной версии сайта

### Оформление заказа

Состав заказа

101 тюльпан x1 8099 Р

Букет Нежный день x2 6640 Р

Итого: 14739 Р

Контактные данные

ФИО

Телефон

E-mail

Подробности доставки

Адрес доставки (улица, дом, квартира)


ДД.ММ.ГГГГ

Время доставки

☐ Согласен на обработку персональных данных

Оформить заказ

Рисунок 5 – Страница оформления заказа на мобильной версии сайта



### Букет Нежный день

**ОПИСАНИЕ**

Букет из нежных альстромерий в розовых оттенках

**СОСТАВ БУКЕТА**

- альстромерия- 15 шт.,
- атласная лента -1 м.,
- пленка матовая - 1,5 м.

**3320 Р**

В корзине 2

Рисунок 6 – Страница товара на планшетной версии сайта