Содержание

[Введение 3](#_Toc122997838)

[1 Предпроектное исследование 4](#_Toc122997839)

[1.1 Обоснование и цели проекта 4](#_Toc122997840)

[1.2 Конкурентная аналитика 6](#_Toc122997841)

[1.3 Текстовый прототип 7](#_Toc122997842)

[1.4 Визуальный прототип 8](#_Toc122997843)

[1.5 Архитектура проекта 12](#_Toc122997844)

[2 Проектирование дизайна web-страниц 15](#_Toc122997845)

[3 Разработка web-приложения 19](#_Toc122997846)

[3.1 Выбор технологий и инструментов 19](#_Toc122997847)

[3.2 Разработка клиентской части 19](#_Toc122997848)

[3.3 Разработка серверной части 21](#_Toc122997849)

[4 Руководство системного программиста 25](#_Toc122997850)

[5 Описание тестового примера 26](#_Toc122997851)

[Заключение 31](#_Toc122997852)

[Список используемых источников 32](#_Toc122997853)

**Введение**

Объектом разработки является web-приложение, для учета заказов в ресторане “Белый лис”. В качестве предмета исследования выступает web-приложение для учета заказов в ресторане “Белый лис”..

Целью данного проекта является создание web-приложения, которое позволит пользователю заказывать товары, а администратору контролировать и усправлять приложением по своему усмотрению.

Задачами курсового проекта являются:

– формирование базы данных основных объектов;

– разработка web-приложения (клиентской и серверной части).

В качестве ожидаемых результатов можно выделить:

– создание «дружественного для пользователя» интерфейса;

– вёрстка, адаптированная для типов устройств: десктоп;

– корректно функционирующие добавления типов, добавление блюд, выбор по категориям, добавление в корзину.

Разработка web-приложения была проведена в несколько этапов с использованием подходящего программного обеспечения и web-сайтов.

Первый этап – аналитика и проектирование, что осуществляется посредством системы аналитики Google trends и данных различных открытых источников.

Второй этап – проектирование дизайна web-страниц. На данном этапе для создания визуального прототипа, дизайн-концепции и дизайн-макетов используется Figma – графический онлайн-редактор.

Третий этап – разработка web-приложения. На данном этапе реализуются две основных составляющих проекта: фронтенд (React, Redux, Axios, TypeScript) и бэкенд (NodeJS, Express, PostgreSQL).

**1 Предпроектное исследование**

**1.1 Обоснование и цели проекта**

Разрабатываемая онлайн-платформа специализируется на предоставлении продукции ресторана «Белый лис», а также на учете заказов клиентов. В эпоху пандемий и повсеместных закрытий учреждений массового прибывания людей, стало необходимым создание ресторанами онлайн-платформ, для поддержания бизнеса и конкурентоспособности. Эта онлайн-платформа является примером решения этих проблем.

Клиенты при выборе места, где можно поесть, уже не заходят во все рестораны, а ищут то, что им по душе на онлайн-ресурсах.

Проанализировав все возможные варианты, их плюсы и минусы, было решено разработать web-приложение, которое позволяет учитывать заказы в ресторане «Белый лис».

Ниже сформулированы краткосрочные, среднесрочные и долгосрочные бизнес-цели проекта, некоторые из которых будут реализовываться и после сдачи: следовательно, у приложения намечен вектор развития.

Бизнес-цели проекта состоят в следующем:

– повышение конкурентоспособности в среднесрочной перспективе с помощью введения уникальных функций, что не встречались в сервисах-аналогах (на основе конкурентного анализа);

– создать базу данных, включающую в себя основные объекты услуг, товары, пользователей и реализации ролей администратора и пользователя, до этапа разработки web-приложения (бекенд-часть);

– расширить базу для пользователей из других стран, сделать локализацию, в долгосрочной перспективе;

С помощью двух инструментов Google trends было произведено исследование популярности запросов о поиске ресторанов в поисковиках(см. рисунок 1). В целом поиск ресторанов является достаточно популярным поисковым запросом. На основе этого можно сделать вывод, что приложение создавать целесообразно.

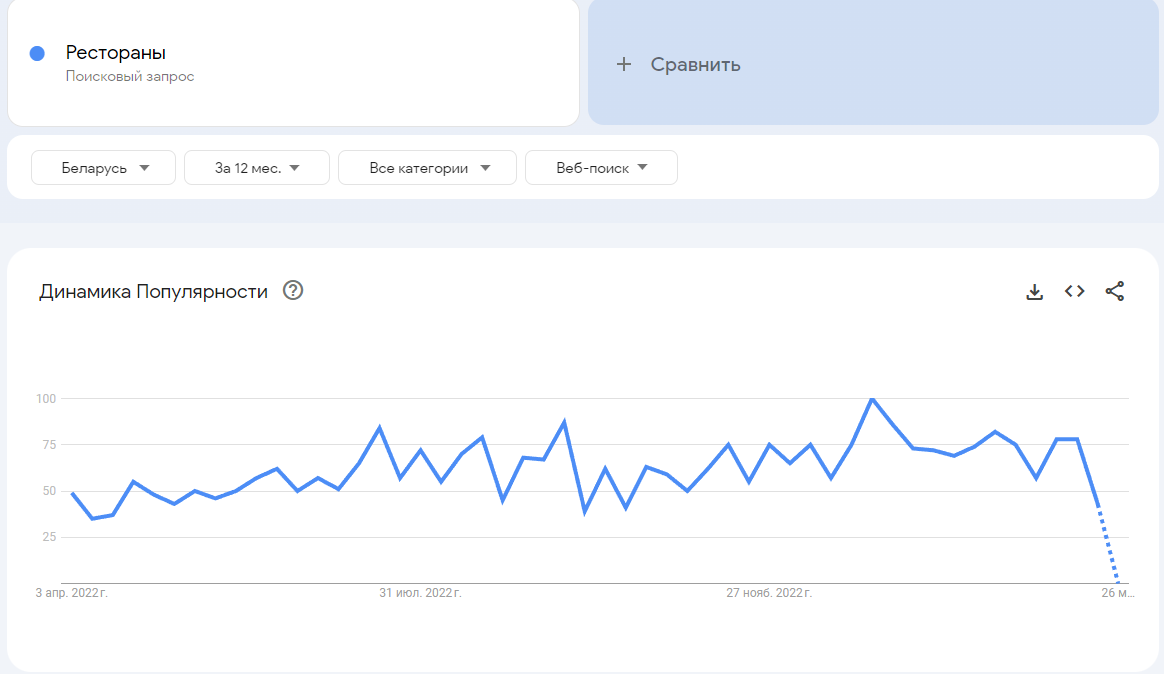


Рисунок 1 – Статистика запросов по поиску ресторанов

Проблемы, которые призвано решить приложение:

– проблему реализации товаров в период пандемии;

– поддержание конкурентоспособности;

– повышение продаж;

– скорости обработки заказов клиентов.

**1.2 Конкурентная аналитика**

Web-приложений, созданных как онлайн-платформ ресторанов слишком много. Среди них можно выделить vilkilozhki.by, burgerhouse.by, carte.by.

Итак, для анализа было выбрано четыре web-приложения:

* Накормим как дома – https://vilkilozhki.by;
* burgerhouse.by – http://burgerhouse.by;
* carte.by– https://carte.by/mogilev/dostavka.

В рассматриваемых web-приложений будут проанализированы следующие характеристики: критерии поиска и качество поиска, удобство создания заказа, простота регистрации, интеграции с различными сервисами, удобство интерфейса, наличие мобильного приложения.

По функционалу все 3 приложения схожи. Но у «carte.by» услуг больше и географический охват распространяется на всю Беларусь, в отличии от «burgerhouse.by» и «Накормим как дома».

Первый конкурент – это приложение «Накормим как дома». Главная страница представлена на рисунке 2.

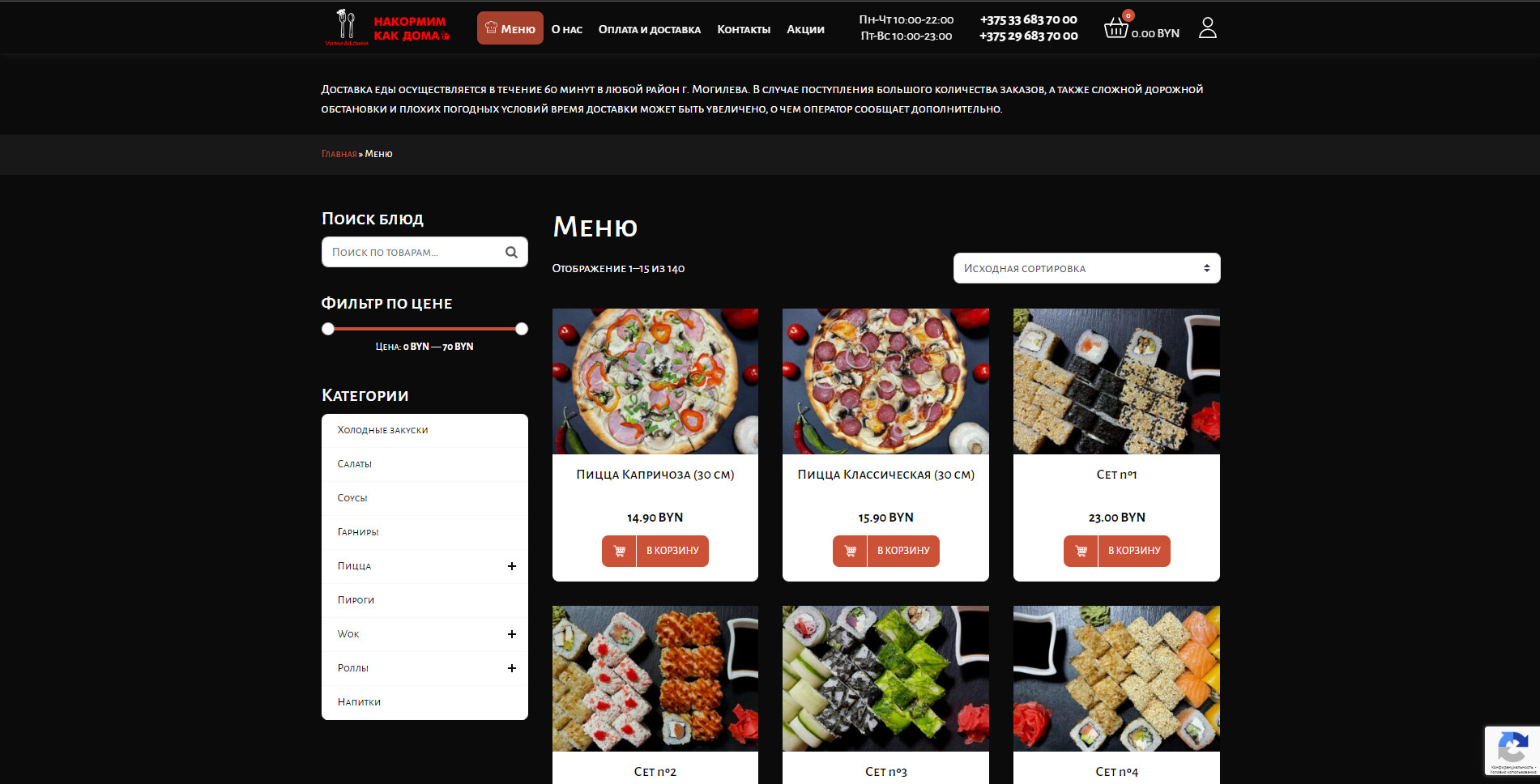


Рисунок 2 – Главная страница Накормим как дома

Преимущества современный дизайн. Достаточно простая аутентификация и регистрация по номеру email.

Недостатки перегруженность интерфейса.

Второй конкурент – это приложение burgerhouse.by. Главная страница представлена на рисунке 3.

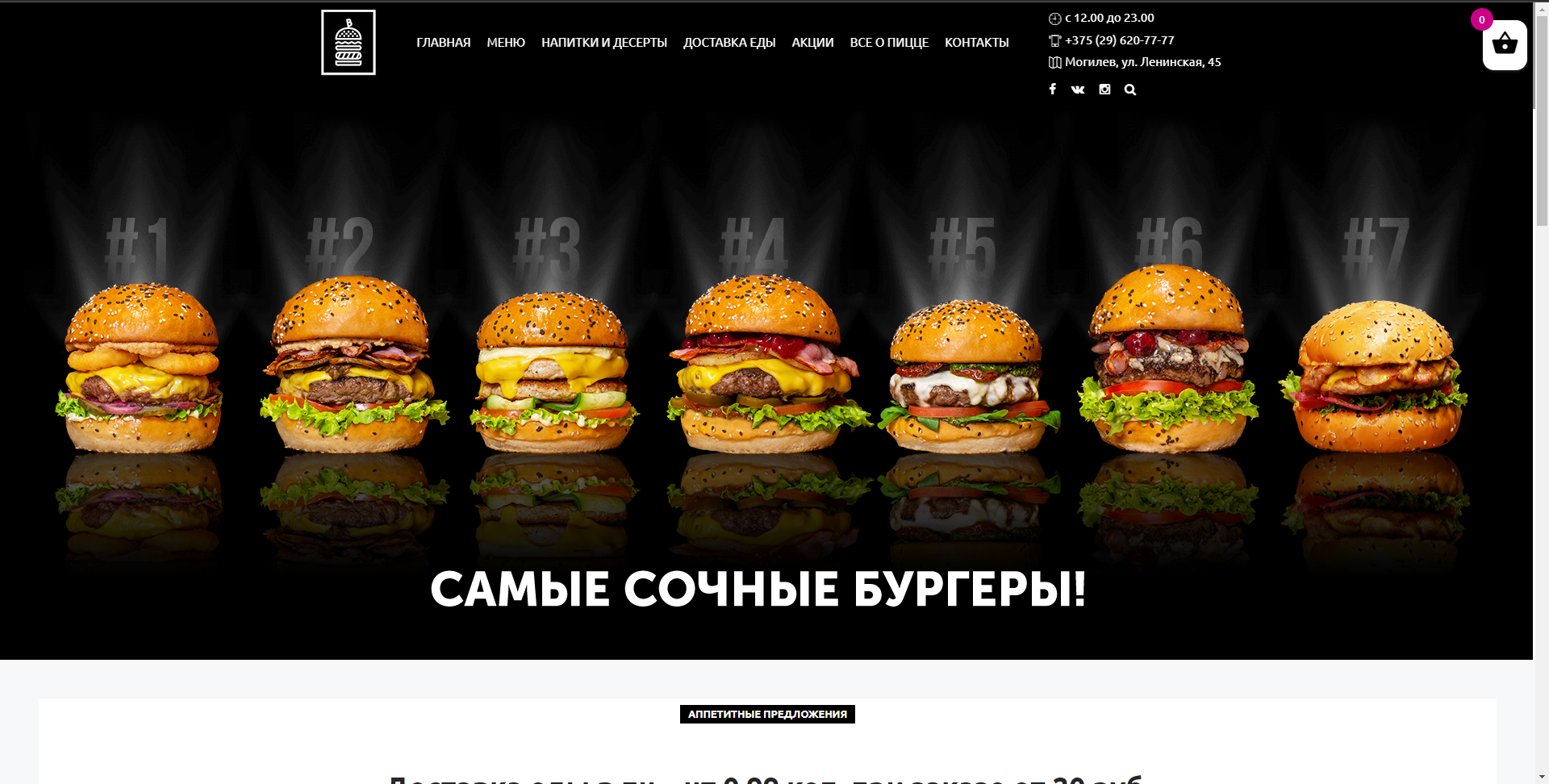


Рисунок 3 – Главная страница burgerhouse.by

Преимуществами данного приложения являются удобное дополнение поиска, нтерфейс создания заявки, удобное разделение по категориям

Из недостатков отсутствие регистрации.

Третий конкурент – это carte.by. Главная страница представлена на рисунке 4.

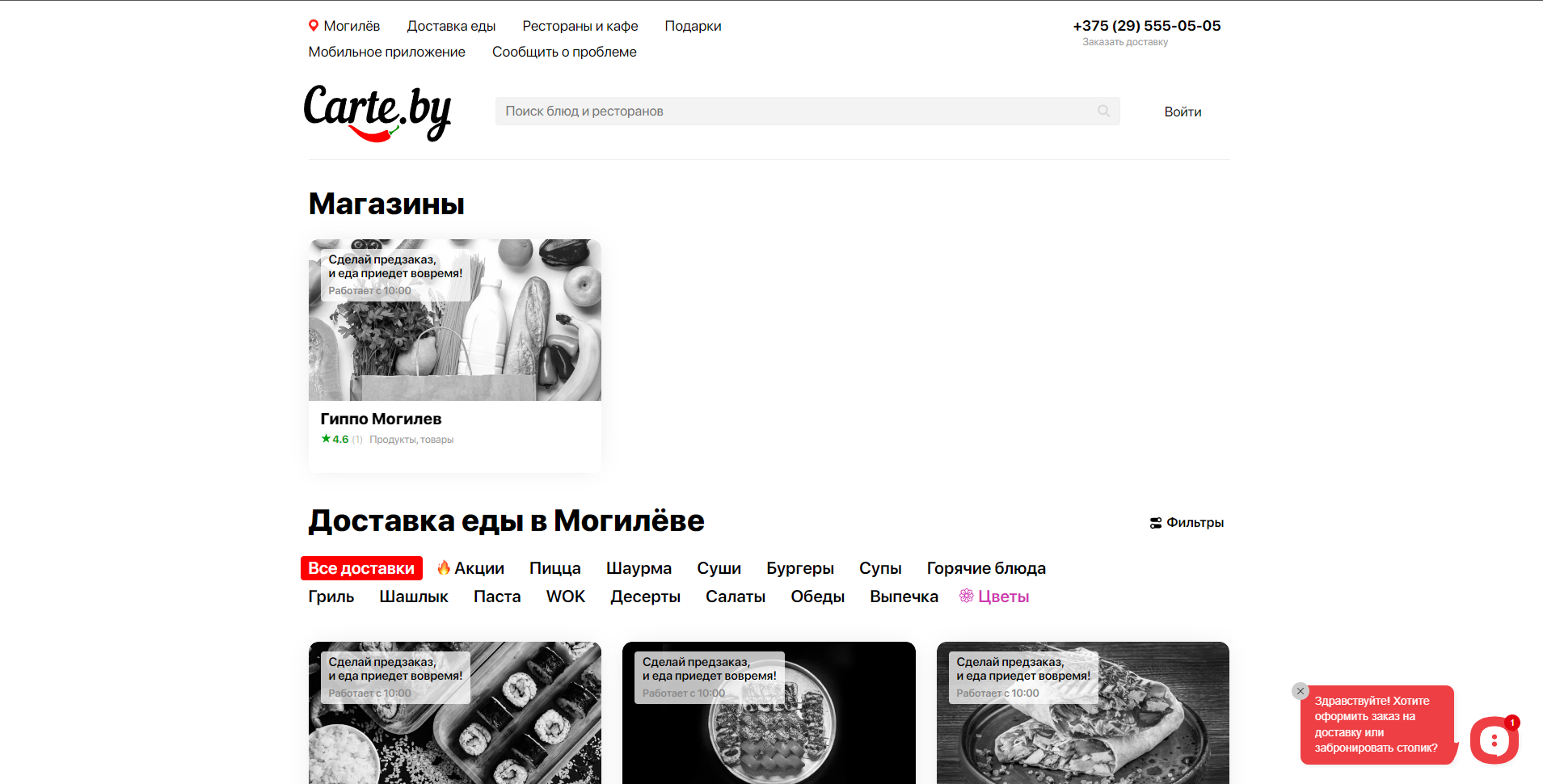


Рисунок 4 – Главная страница carte.by

Преимущества удобный поиск, разделение на категории, интуитивной понятный интерфейс, приятный дизайн.

Из недостатков можно выделить отсутсвие как таковой регистрации.

Для оценки характеристик будет использоваться – 3-ёхбальная шкала, где 1 – функция отсутствует, а 3 – функция есть.

Сравнение характеристик рассматриваемых web-приложений представлено в таблице 1.

Таблица 1 – Сравнение характеристик рассматриваемых web-приложений

|  |  |  |  |
| --- | --- | --- | --- |
| Характеристика | Web-приложение | | |
| Накормим как дома | burgerhouse.by | carte.by |
| Критерии поиска и его качество | 3 (большое количество критериев поиска и высокое качество поиска) | 2 (меньшее количество критериев поиска, но высокое качество поиска) | 3 (большое количество критериев поиска и высокое качество поиска) |

Продолжение таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| Удобство создания заявки | 2 (заказ создается сразу) | 3 (заявка создается пошагово с множеством критериев) | 3 (заявка создается сразу с множеством критериев: имя, геолокация, фото и т.д.) |
| Простота регистрации | 2 (необходимо иметь аккаунт email) | 3 (для регистрации можно использовать множество соцсетей на выбор) | 3 (необходимо иметь номер телефона) |
| Удобство интерфейса | 3 (современный, удобный дизайн) | 2 (интеграции с яндекс картами) | 2 (удобный и лаконичный, но немного устаревший) |
| География услуг | 1 (Могилев) | 1 (Могилев) | 2 (Беларусь) |
| Итого | 11 | 11 | 13 |

По анализу приложений можно сделать следующие выводы:

– наиболее подходящим является приложение «carte.by», https://carte.by/mogilev/dostavka – оно обладает приятным, современным дизайном, удобно в использовании. Недостатком является отсутствие как таковой регистрации.

– оптимального варианта приложения, удовлетворяющего всем требованиям, в настоящее время нет.

**1.3 Текстовый прототип**

Страницы web-приложения: Главная, Подробная информация, Страница авторизации, Корзина, Страница роли администратора.

**Главная страница.**

Несколько секций, как у лендинга. Хэдер –, логотип, название, поиск, корзина и авторизация.

**Корзина**

Хэдер –, логотип, название, поиск, корзина и авторизация. Тело из услуги и кнопки покупки.

**Авторизация.**

Секция с меню авторизации (по телефону).

**Выбор услуги для покупки или добавления в корзину.**

Модальное окно с информацией об выбраной услуге и заблюреный остальной экран.

**Home admin page.**

Sidebar с названиями категорий, окно поиска, карточки статистики сайта.

**Services admin page.**

Sidebar с названиями категорий, окно поиска, таблица зарегистрированных услуг(из БД).

**Services admin page add services.**

Sidebar с названиями категорий, форма для ввода данных.

**1.4 Визуальный прототип**

На данном этапе был разработан визуальный прототип web-приложения с учетом требований, описанных в текстовом прототипе. На рисунке 5-8 представлены визуальные прототипы страниц web-приложения. Разработано при помощи Figma[5].

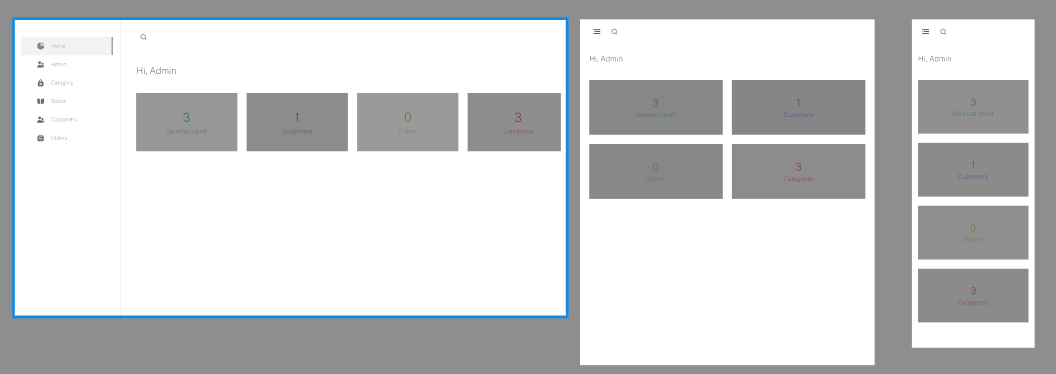


Рисунок 5 – Визуальный прототип Home page admin

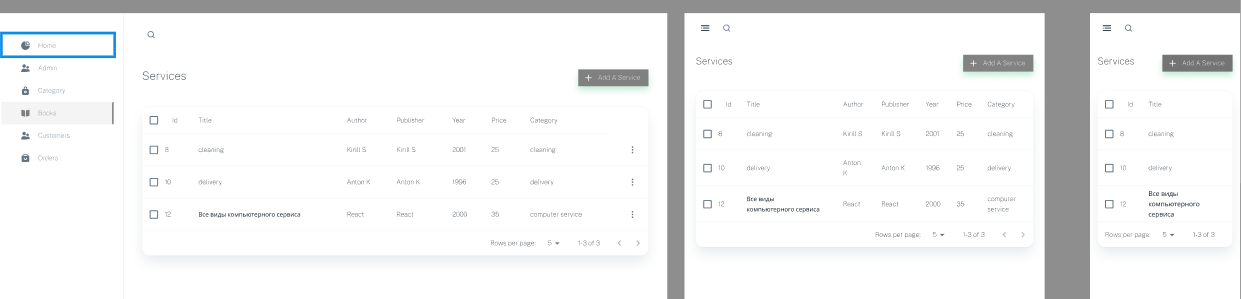


Рисунок 6 – Визуальный прототип Services page admin

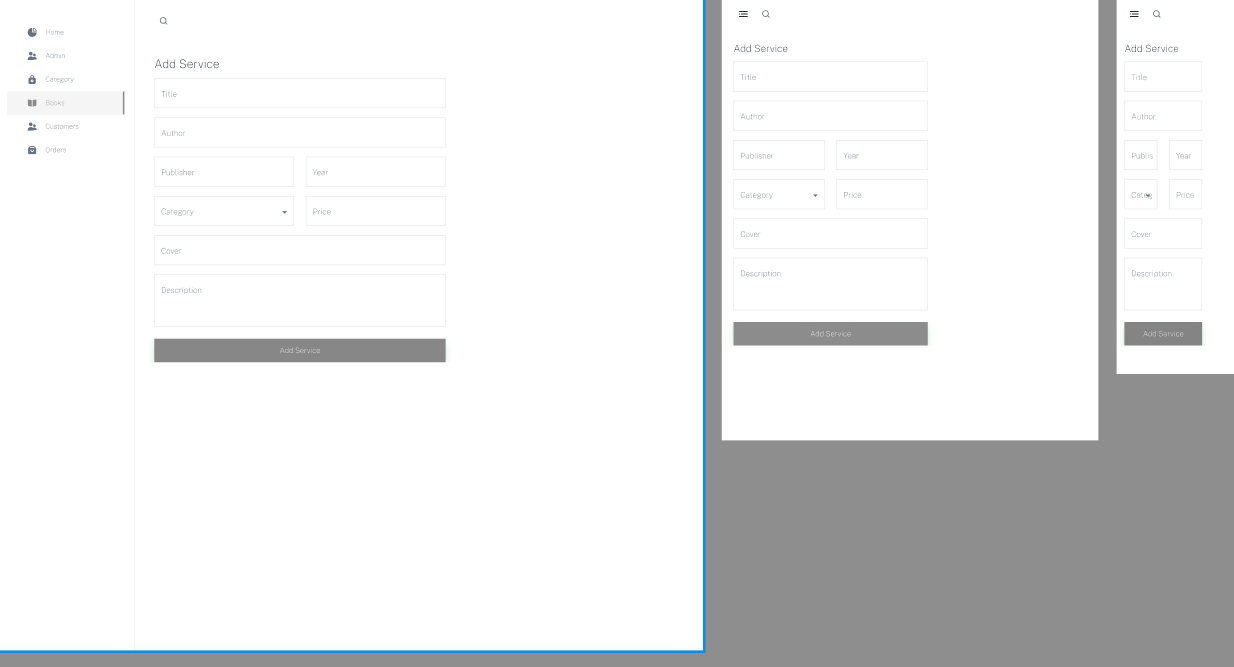


Рисунок 7 – Визуальный прототип Services add page admin

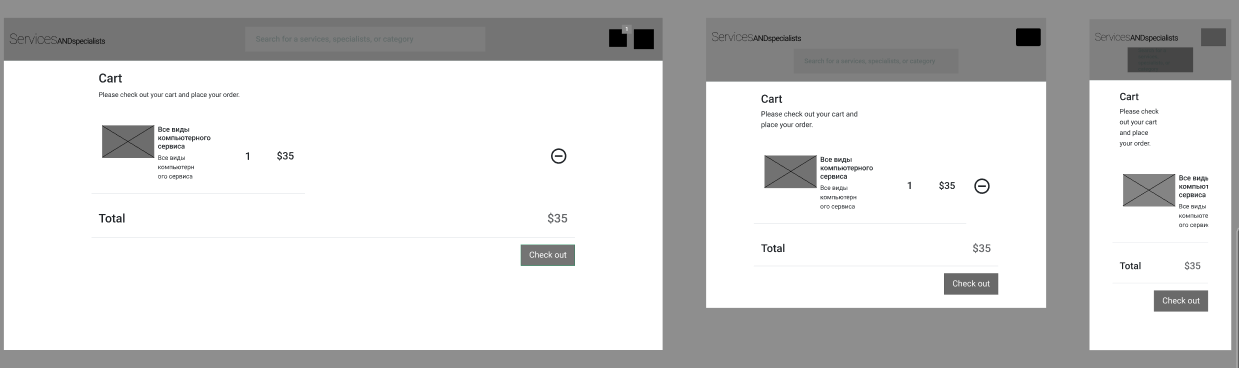


Рисунок 8 – Визуальный прототип страницы корзина

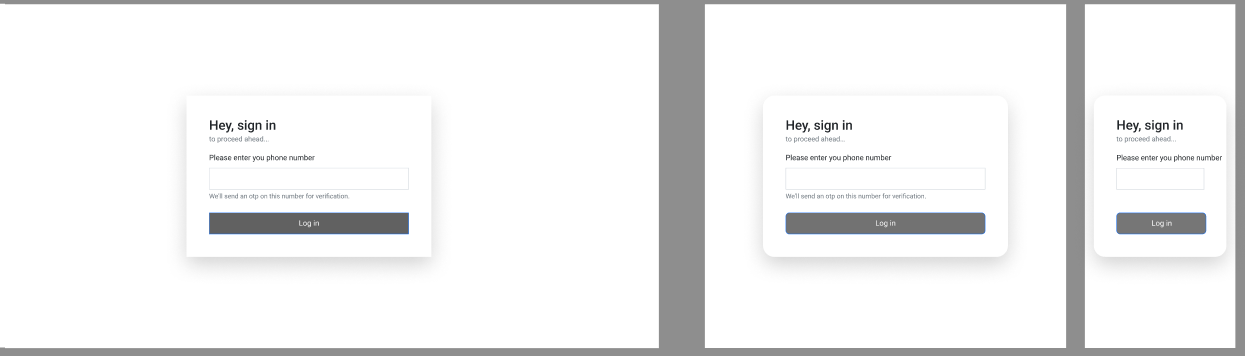


Рисунок 9 – Визуальный прототип страницы авторизации

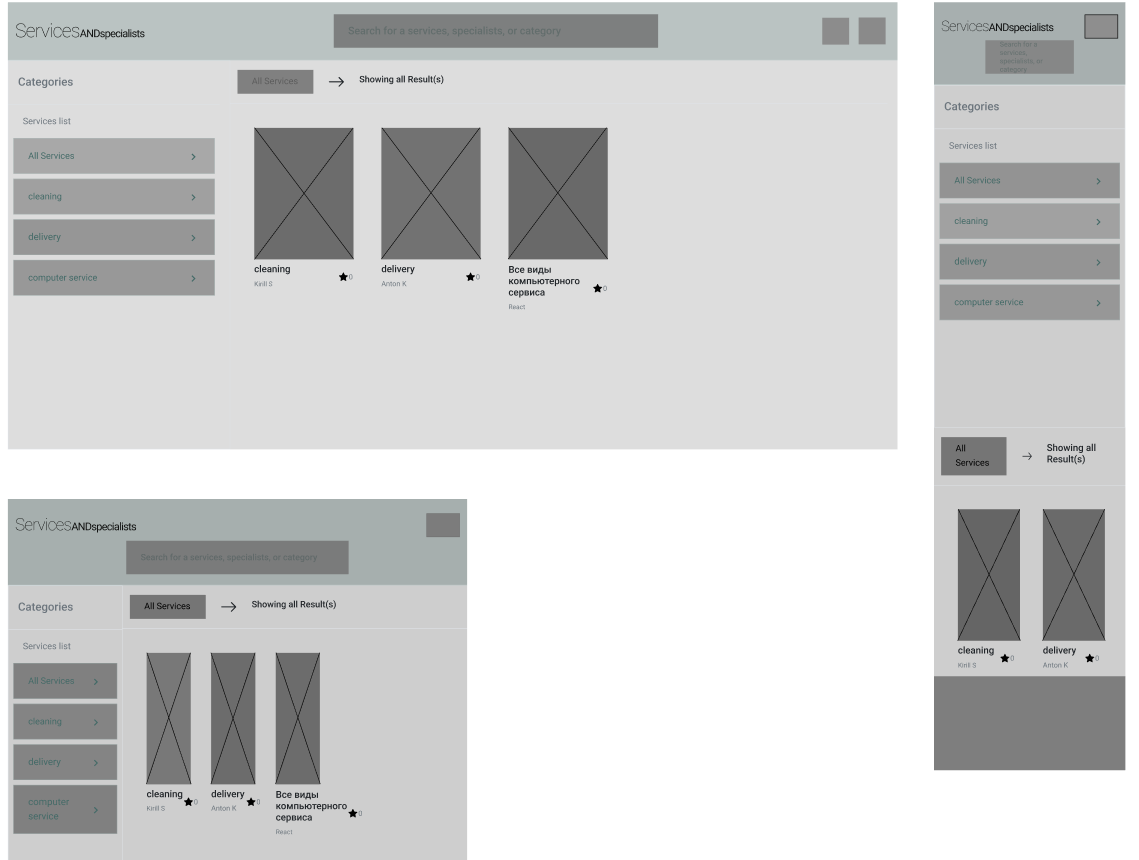


Рисунок 10 – Визуальный прототип Home page

**1.5 Архитектура проекта**

Данное web-приложение имеет древовидную структуру (рисунок 11): из всех видов структур эта является наиболее универсальным вариантом. В таком случае каждой категории отводится отдельная ветка: раздел или подраздел. Как правило, пользователям привычнее всего общаться именно с такими ресурсами.

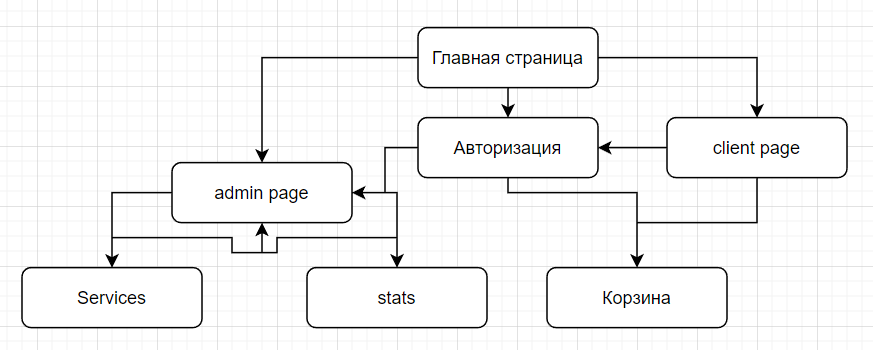


Рисунок 11 – Структура web-приложения

Описание структуры сайта: на главной странице сайта представлены товары и общая информация. Для просмотра этой страницы не обязательно авторизовываться. Это сделано для того, чтобы пользователь мог изучить сайт и посерфить по нему. Однако при выборе товара и добавлении его в корзину или сразу для покупки необходимо авторизоваться. Также авторизация доступна по кнопке в navbar. После авторизации происходит авторизация и валидация пользователя и его перекинет либо на страницу админа, либо на страницу пользователя. Авторизация происходит по токену, токен хранится в localstorage. Поэтому пользователь не будет выкинут при переходе по страницам. На странице админа доступен navbar со страницами модерации. Там можно добавить или модерировать позиции в БД, либо посмотреть статистику на сайте. Из корзины можно сделать заказ, либо убрать его из корзины.

Логика размещения контента будет основана на том, что на главной странице имеются блоки, приглашающие на страницу того или иного раздела. Ознакомившись с ними, пользователь смогут легко выбрать интересующий его раздел и перейти, нажав на соответствующую кнопку. Однако, стоит отметить (что отражено на рисунке 7), разделы также связаны напрямую, потому что если рассматривать нового пользователя, то с высокой вероятностью он захочет подробнее ознакомиться с приложением перед тем, как пройти авторизацию и приступить к работе.

**2 Проектирование дизайна web-страниц**

На первом этапе проектирования дизайна web-страниц были разработаны макеты страниц приложения в соответствии с визуальными прототипами. На рисунках 12-18 представлены макеты страниц web-приложения.

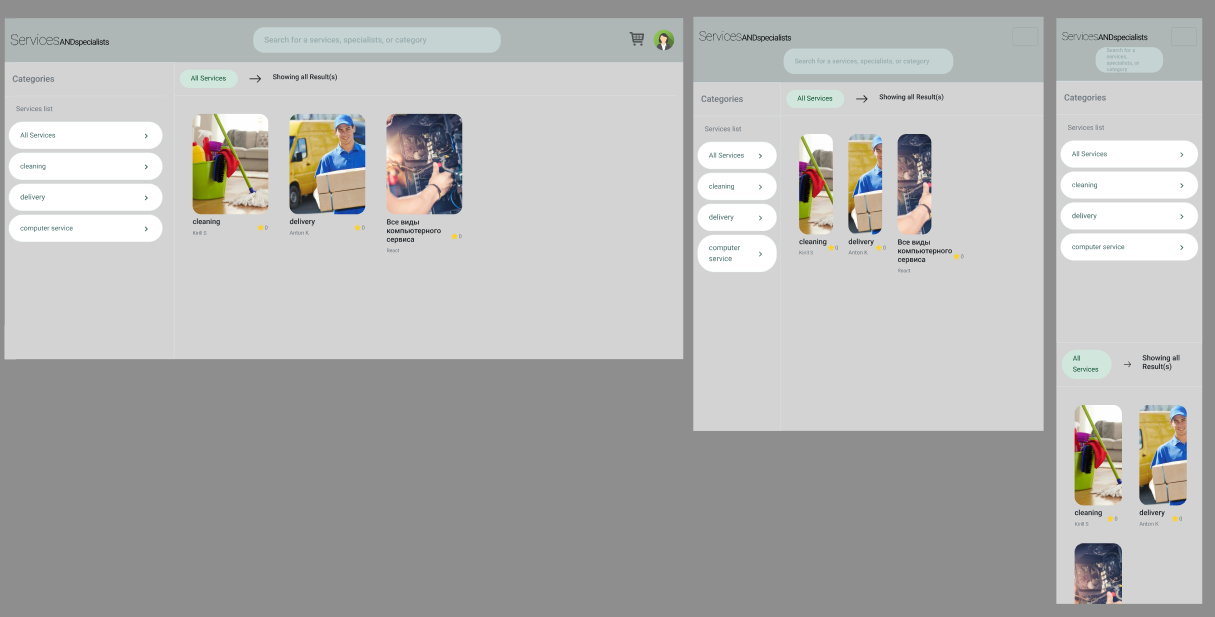


Рисунок 12 – Макет страницы «Главная»



Рисунок 13 – Макет страницы «Корзина»

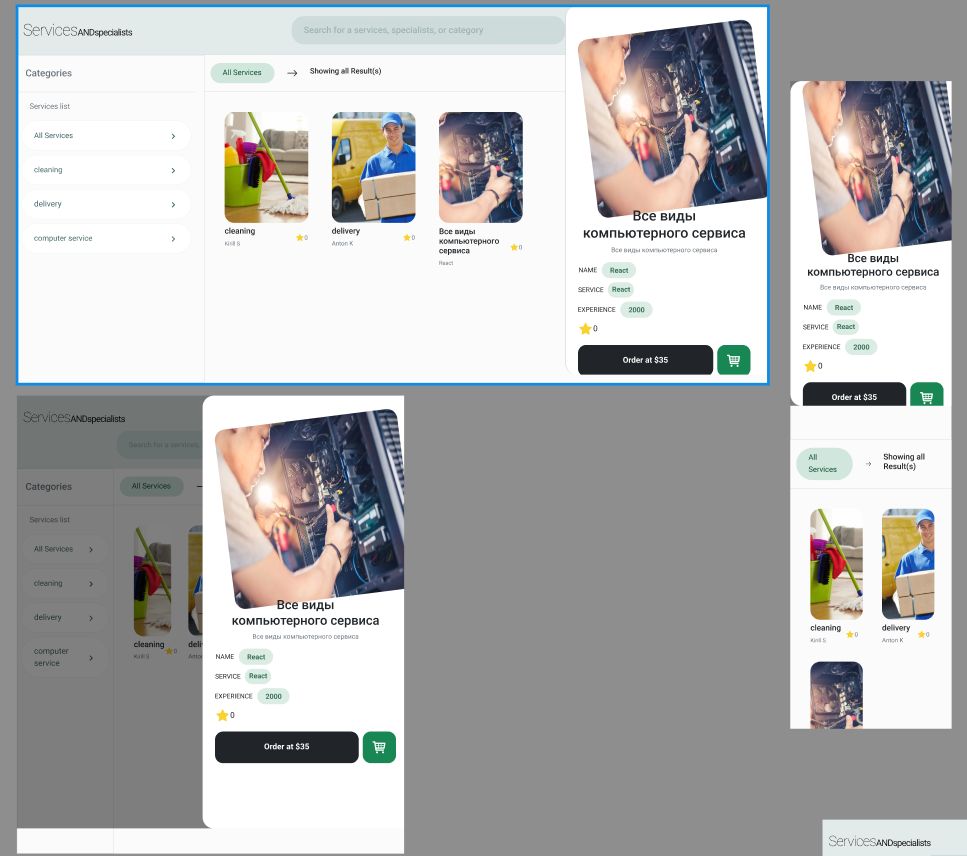


Рисунок 14 – Макет страницы «Выбор услуги»



Рисунок 15 – Макет страницы «Авторизация»

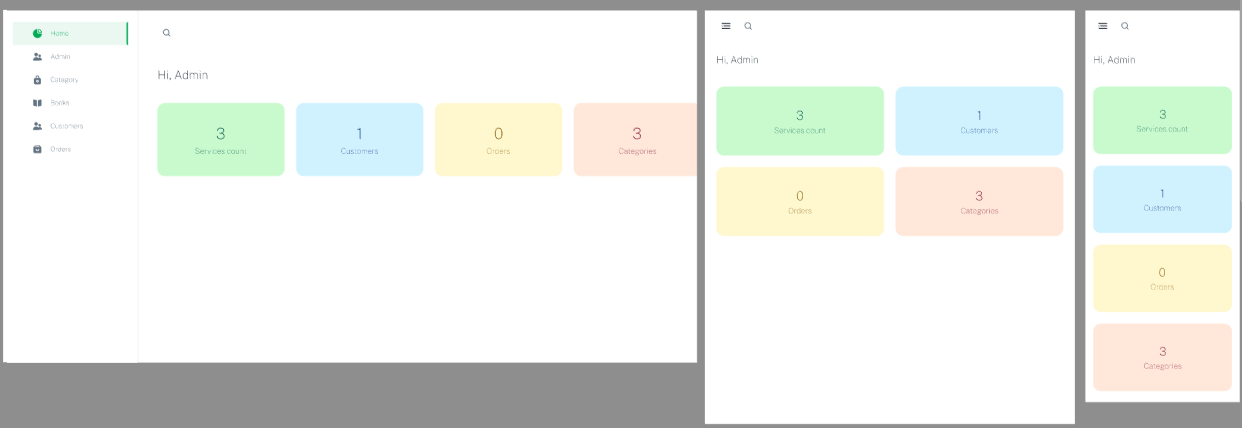


Рисунок 16 – Макет страницы «Admin page»

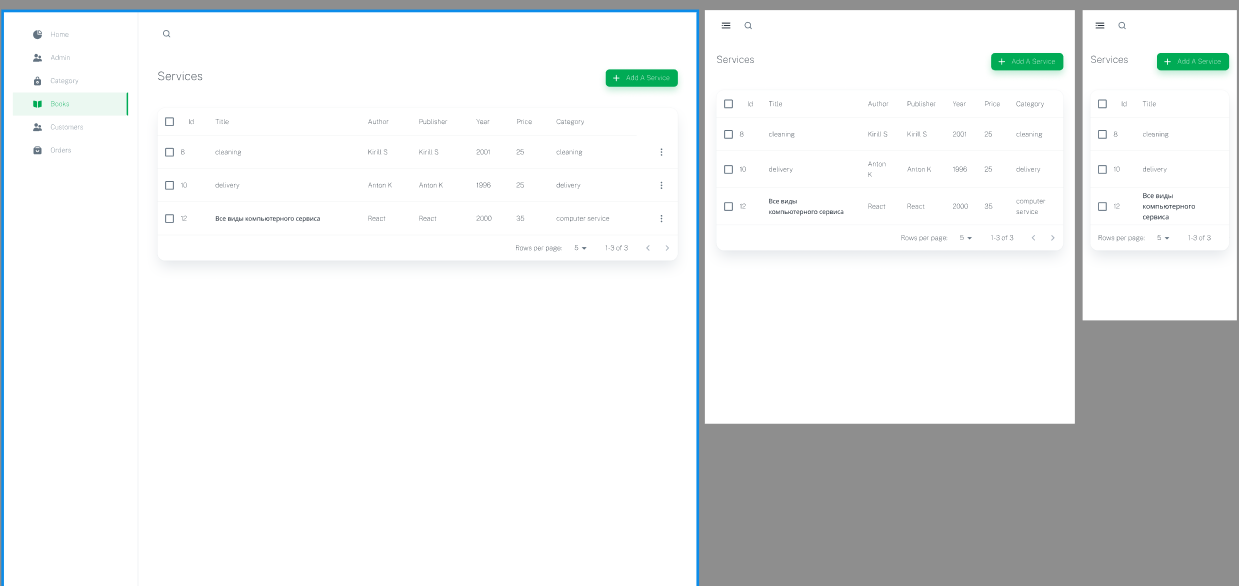


Рисунок 17 – Макет страницы «Услуги admin»

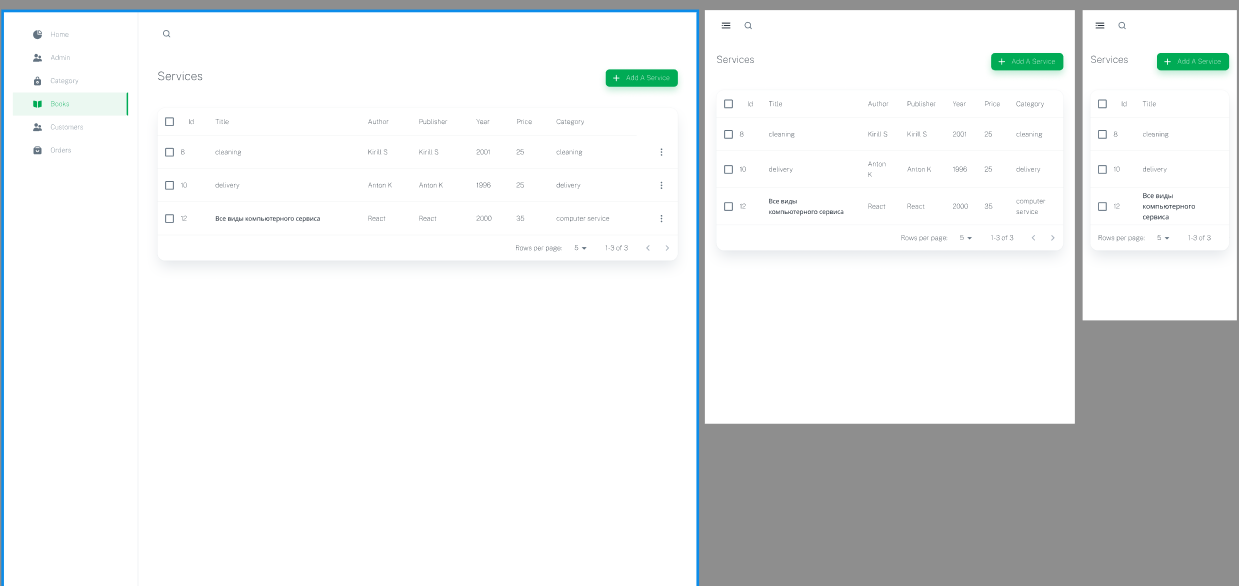


Рисунок 18 – Макет страницы «add Услуги admin»

В результате разработки web-страниц были созданы макеты, соответствующие требованиям и визуальным прототипам. Спроектированный дизайн достаточно современен, а также прост для понимания пользователем контента. Важно заметить, что сайт был выполнен в неярких цветах, что, как правило, располагает к себе и повышает время пребывания пользователя на сайте.

**3 Разработка web-приложения**

**3.1 Выбор технологий и инструментов**

Для разработки клиентской части web-приложения был выбран React – это библиотека JavaScript с открытым кодом для создания внешних пользовательских интерфейсов. В отличие от других библиотек JavaScript, предоставляющих полноценную платформу приложений, React ориентируется исключительно на создание представлений приложений через компоненты, которые сохраняют состояние и генерируют элементы пользовательского интерфейса[12].

Для разработки серверной части web-приложения была выбрана платформа Spring Framework языка Java [3].

По сути Spring Framework представляет собой просто [контейнер внедрения зависимостей](https://en.wikipedia.org/wiki/Dependency_injection), с несколькими удобными слоями (например: доступ к базе данных, прокси, аспектно-ориентированное программирование, RPC, веб-инфраструктура MVC). Это все позволяет вам быстрее и удобнее создавать Java-приложения.

Также при разработке была использована такая библиотека, как MUI React – это библиотека для стилизации приложений React. Она позволяет использовать встроенные компоненты библиотеки для создания компонентов web-страниц. Данные компоненты имеют под собой стилизацию и могут быть изменены посредством изменения характеристик компонента, а не только воздействием на стиль компонента.

Для хранения данных была выбрана база данных на основе PostgreSQL[11].PostgreSQL —  [свободная](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D0%BE%D0%B1%D0%BE%D0%B4%D0%BD%D0%BE%D0%B5_%D0%9F%D0%9E) [объектно-реляционная](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D1%80%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%A1%D0%A3%D0%91%D0%94) [система управления базами данных](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) (СУБД). Существует в реализациях для множества [UNIX-подобных](https://ru.wikipedia.org/wiki/UNIX-%D0%BF%D0%BE%D0%B4%D0%BE%D0%B1%D0%BD%D0%B0%D1%8F_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0) платформ, включая [AIX](https://ru.wikipedia.org/wiki/AIX), различные [BSD-системы](https://ru.wikipedia.org/wiki/BSD), [HP-UX](https://ru.wikipedia.org/wiki/HP-UX), [IRIX](https://ru.wikipedia.org/wiki/IRIX), [Linux](https://ru.wikipedia.org/wiki/Linux), [macOS](https://ru.wikipedia.org/wiki/MacOS), [Solaris](https://ru.wikipedia.org/wiki/Solaris)/[OpenSolaris](https://ru.wikipedia.org/wiki/OpenSolaris), [Tru64](https://ru.wikipedia.org/wiki/Tru64), [QNX](https://ru.wikipedia.org/wiki/QNX), а также для [Microsoft Windows](https://ru.wikipedia.org/wiki/Microsoft_Windows).

**3.2 Разработка клиентской части**

В ходе выполнения курсовой работы для наглядной демонстрации структуры web-приложения была разработана диаграмма компонентов, представленная на рисунке 18. Каждая web-страница представлена в виде одного компонента, логика и данные в который поступают через другие компоненты.

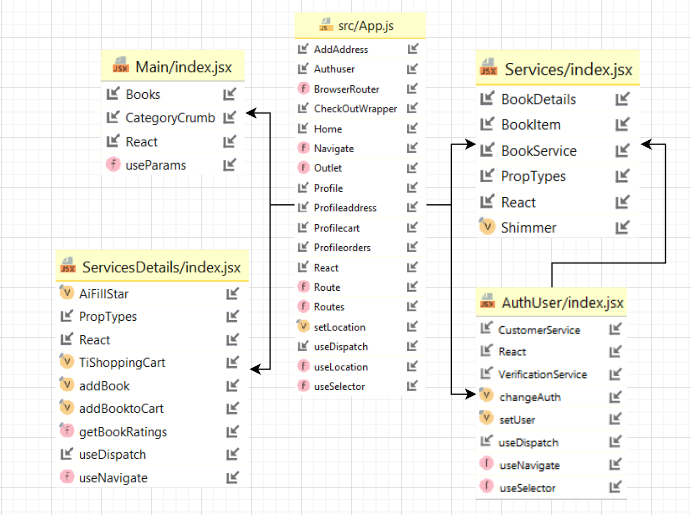


Рисунок 19 – Диаграмма компонентов web-приложения

Для связки с серверной частью в клиентской части используется библиотека axios. Axios — это широко известная JavaScript-библиотека. Она представляет собой HTTP-клиент, основанный на промисах и предназначенный для браузеров и для Node.js.

**3.3 Разработка серверной части**

Для выполнения курсовой работы была создана база данных для хранения всей необходимой информации. Для проектирования базы данных использовалась PostgreSQL. Структура базы данных представлена на рисунке 20.

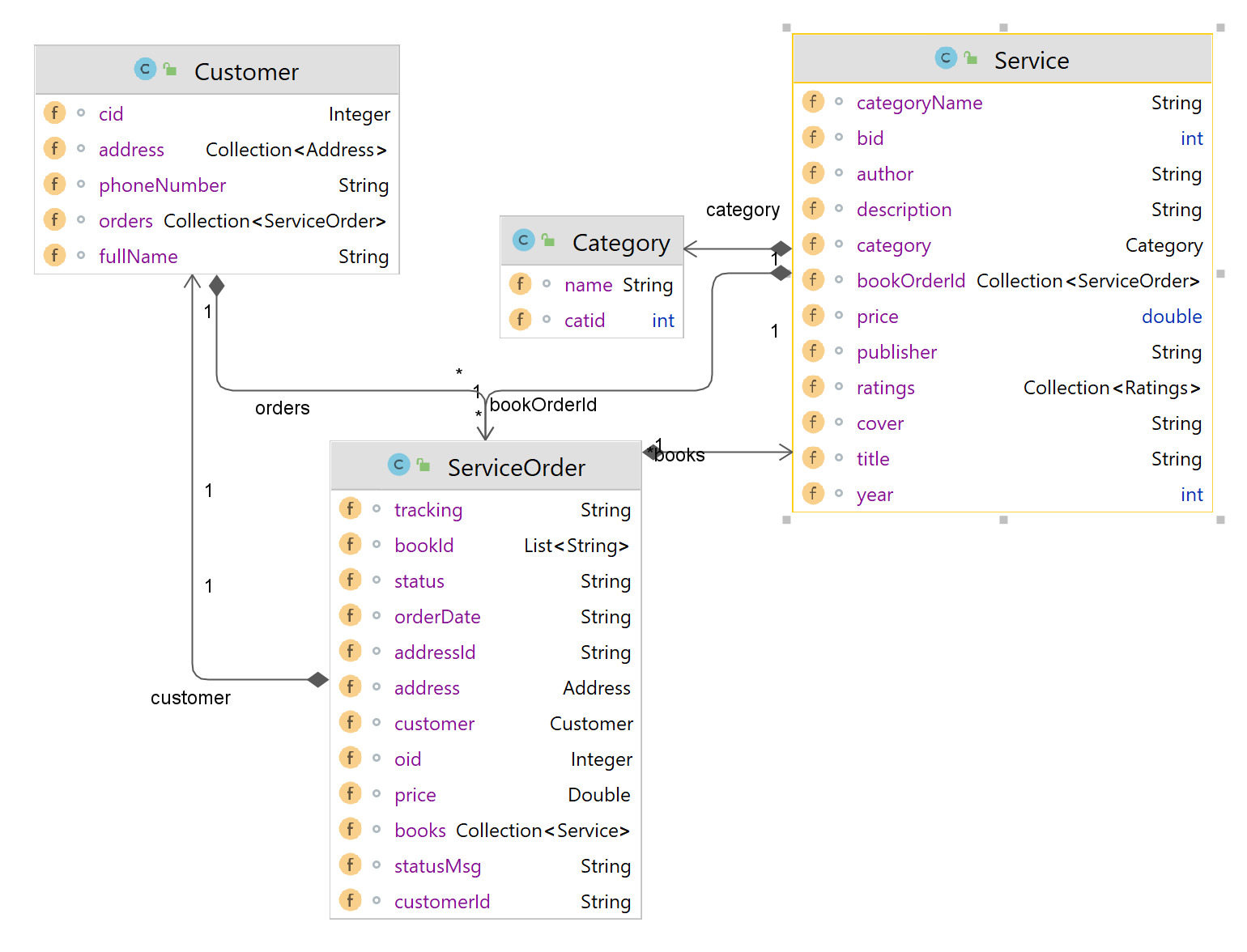


Рисунок 20 – Структура базы данных

База данных содержит пять сущностей: Customer, Service, Category, ServiceOrder.

Сущность Category необходима для хранения информации о категориях, к которым относятся услуги. Для представления этой информации использованы такие атрибуты, как:

– id идентификатор категории;

– name – наименование категории;

Сущность Service необходима для хранения информации о категориях, к которым относятся услуги. Для представления этой информации использованы такие атрибуты, как:

– id идентификатор категории;

– author – имя автора;

– description – описание товара;

– price – цена;

– cover – изображение;

– title – тайтл;

– year – цена;

Реализованы связи между таблицами.

Сущность Customer необходима для хранения информации о заказчиках. Для представления этой информации использованы такие атрибуты, как:

– id идентификатор заказчика;

– address – адрес;

– phoneNumber – номер телефона;

– fullName – Имя;

Сущность ServiceOrder необходима для хранения информации о заказчиках. Для представления этой информации использованы такие атрибуты, как:

– id идентификатор заказчика;

– order\_date – дата заказа;

– price – цена;

– status – стутус заказа;

– status\_msg – сообщение о статусе заказа;

Связь между сущностями Customer и ServiceOrder идентифицирующая, т.к. сущность ServiceOrder не может быть однозначно идентифицирована, если не задана сущность Customer (нельзя разместить на странице заказ, если нет заказчика). Тип связи 1:М, т.к. заказчику может соответствовать множество заказов.

Связь между сущностями Category и Service идентифицирующая, т.к. сущность Service не может быть однозначно идентифицирована, если не задана сущность Category (нельзя разместить на странице элемент услуги если нет категории услуги). Тип связи 1:М, т.к. категории может соответствовать не одна услуга.

Связь между сущностями Service и ServiceOrder идентифицирующая, т.к. сущность ServiceOrder не может быть однозначно идентифицирована, если не задана сущность Service. Тип связи 1:M, т.к. одному заказу может соответствовать множество услуг.

Серверная часть web-приложения была реализована с помощью Spring Framework(Boot, Data, Security)[16-17], основанная на контроллерах, отправляющих http запросы на web-страницу, моделях с базы данных и сервисах, получающих данные из базы данных.

Диаграммы классов приложения представлены на рисунках 20 – 22.

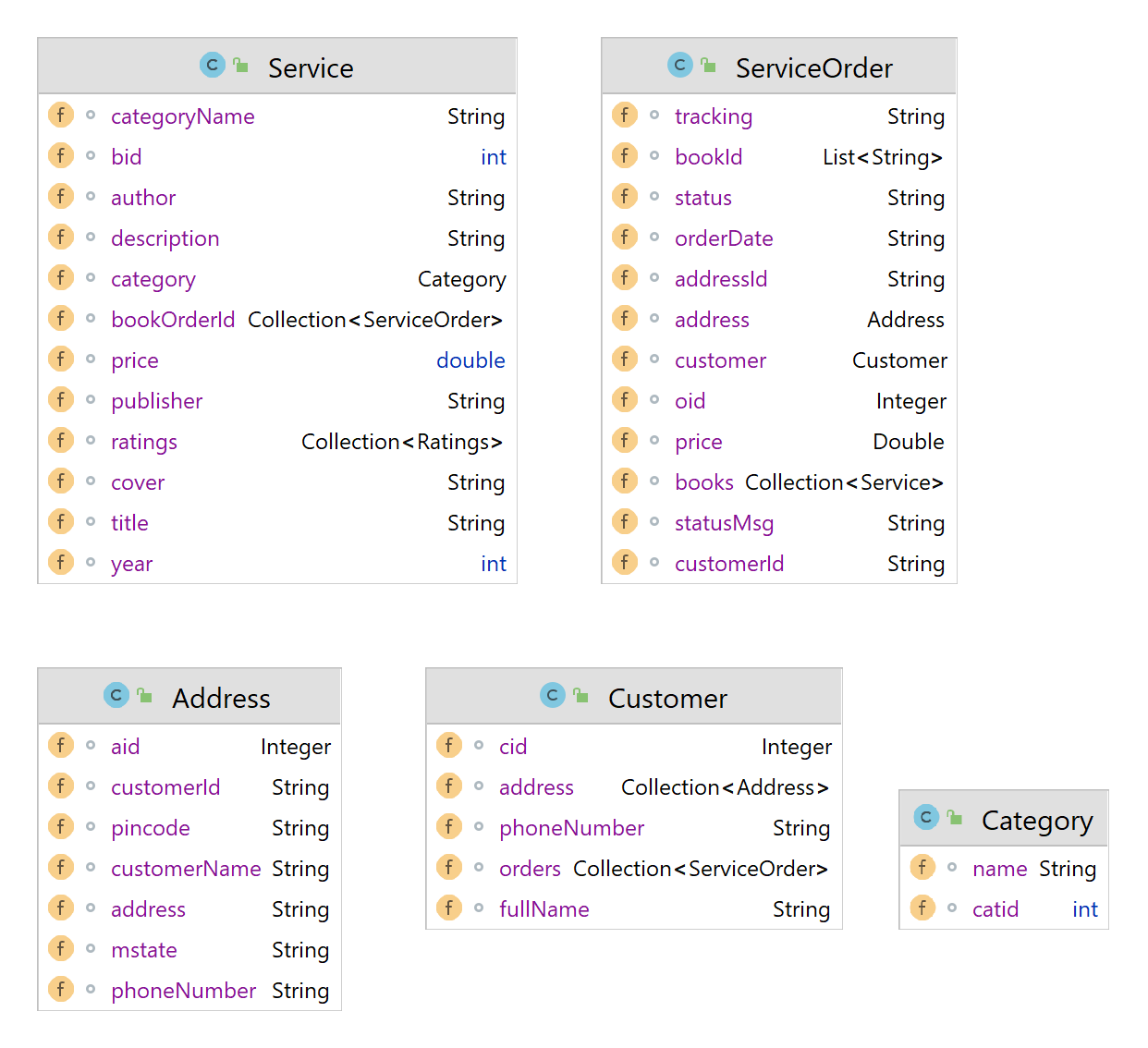


Рисунок 21 – Диаграмма классов моделей приложения

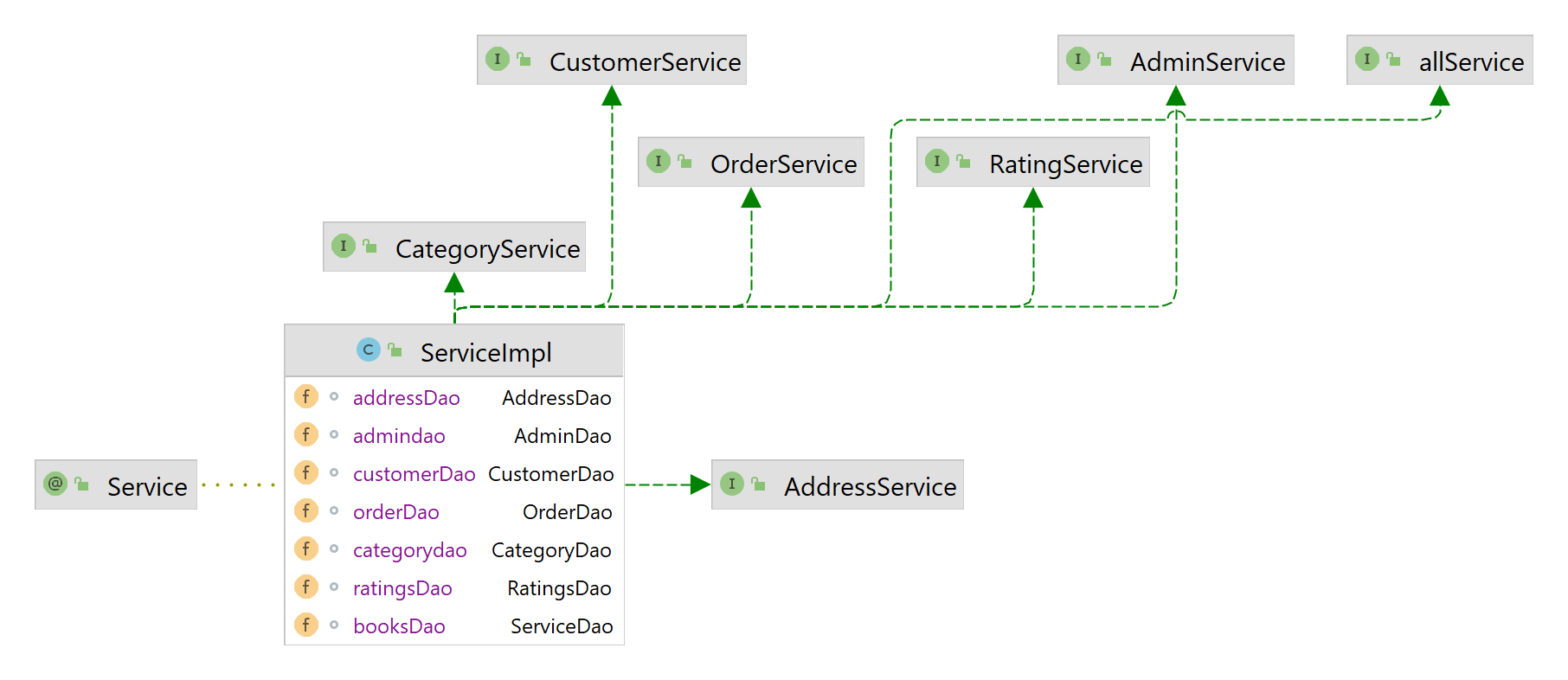


Рисунок 22 – Диаграмма классов сервисов приложения

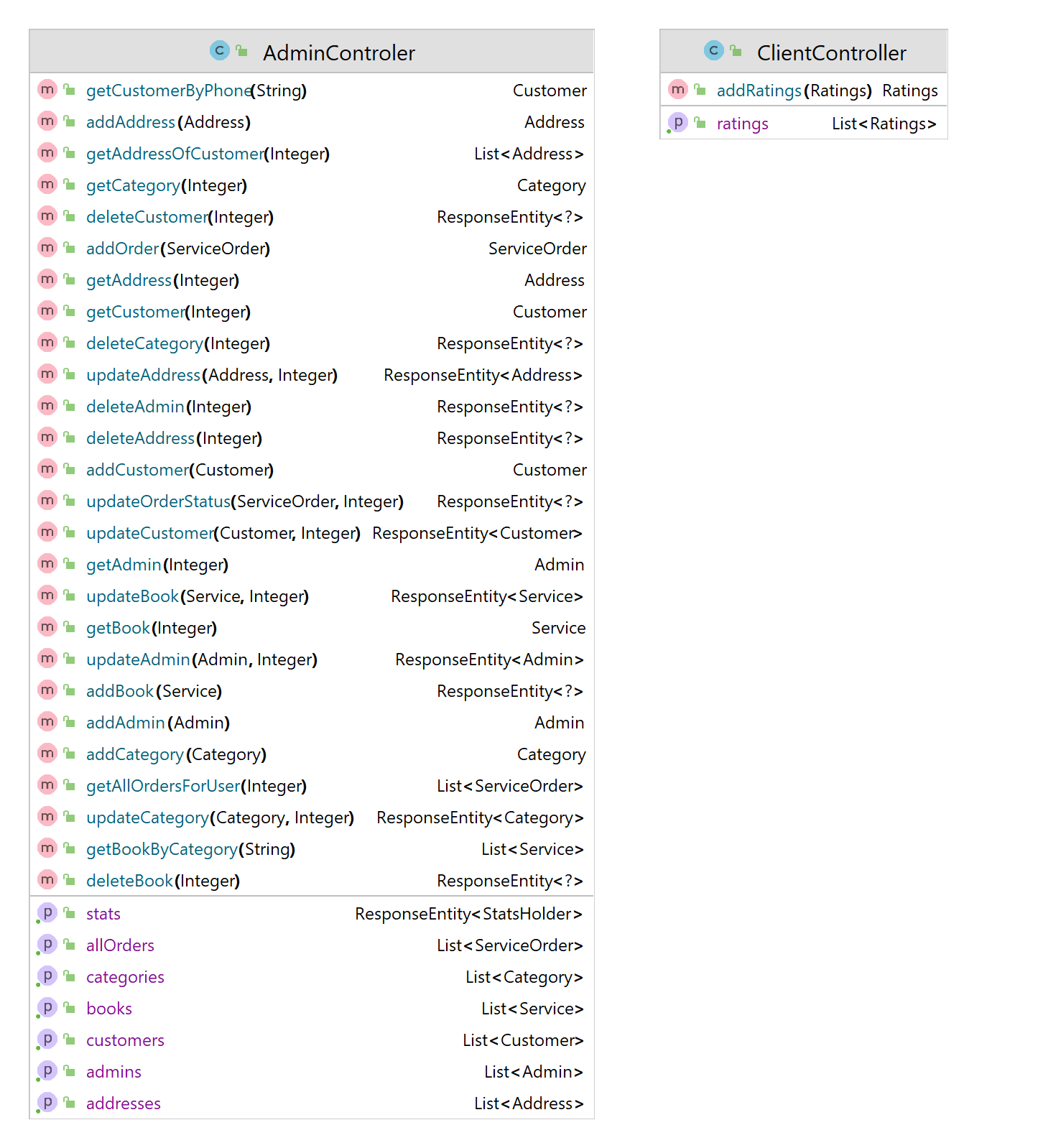


Рисунок 23 – Диаграмма классов контроллеров приложения

**4 Руководство системного программиста**

На компьютере должна быть установлена платформа node.js(LTS) 16.16.0. Официальные установочные пакеты для всех основных платформ можно скачать с сайта https://nodejs.org/en/blog/release/v16.16.0/

Что должно быть установлено:

– Intellij IDEA <https://www.jetbrains.com/idea/download/>;

– JDK 17 <https://corretto.aws/downloads/latest/amazon-corretto-17-x64-windows-jdk.zip>;

– NodeJS <https://nodejs.org/en/>;

– БД PostgreSQL https://www.postgresql.org/download/;

– Любой редактор JS кода (делал через Fleet).

Открыть проект серверной части через IDEA, подождать пока maven загрузит зависимости и запустить проект.

Распаковать и установить PostgreSQL. Перейти в pgAdmin 4. Создать БД с паролем как в application.properties, либо если со своими настройками БД, то изменить файл конфигурации application.properties.

Открыть проект клиентской части через Fleet (клиентский проект находится в папке react-client). Открыть терминал и запустить команду npm install --force. После установки пакетов запустить команду npm start, дождаться загрузки и перейти в браузере на URL localhost:3000.

**5 Описание тестового примера**

Для демонстрации работы web-приложения поиска услуг и специалистов следует рассмотреть процессы добавления услуги в корзину и просмотра всех объявлений.

При запуске приложения откроется страница «Главная», на которой размещено навигационное меню, каждый пункт которого приглашает ознакомится с тем или иным разделом сайта. Результат представлен на рисунке 24.

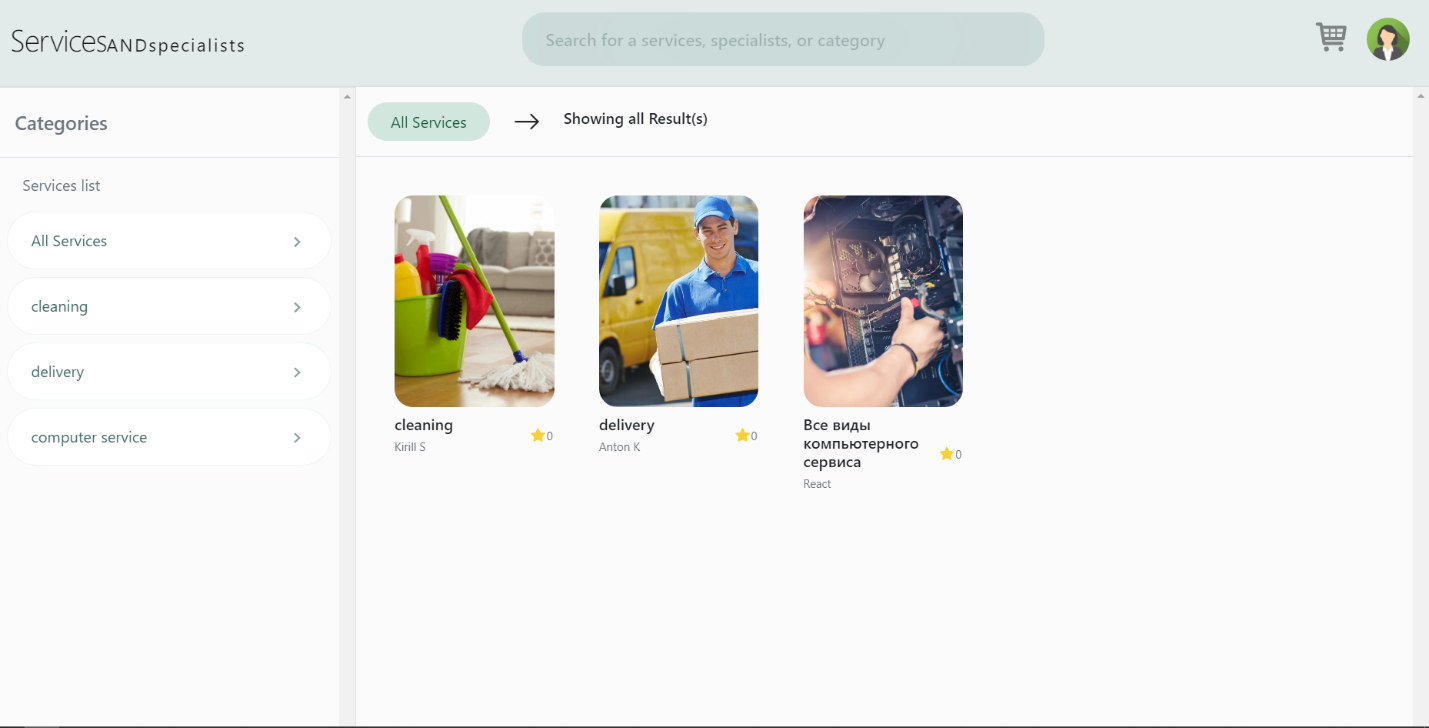


Рисунок 24 – Страница «Главная»

Для того, чтобы добавить услугу в таблицу необходимо выбрать услугу и нажать на значок корзины. Результат продемонстрирован на рисунке 25.

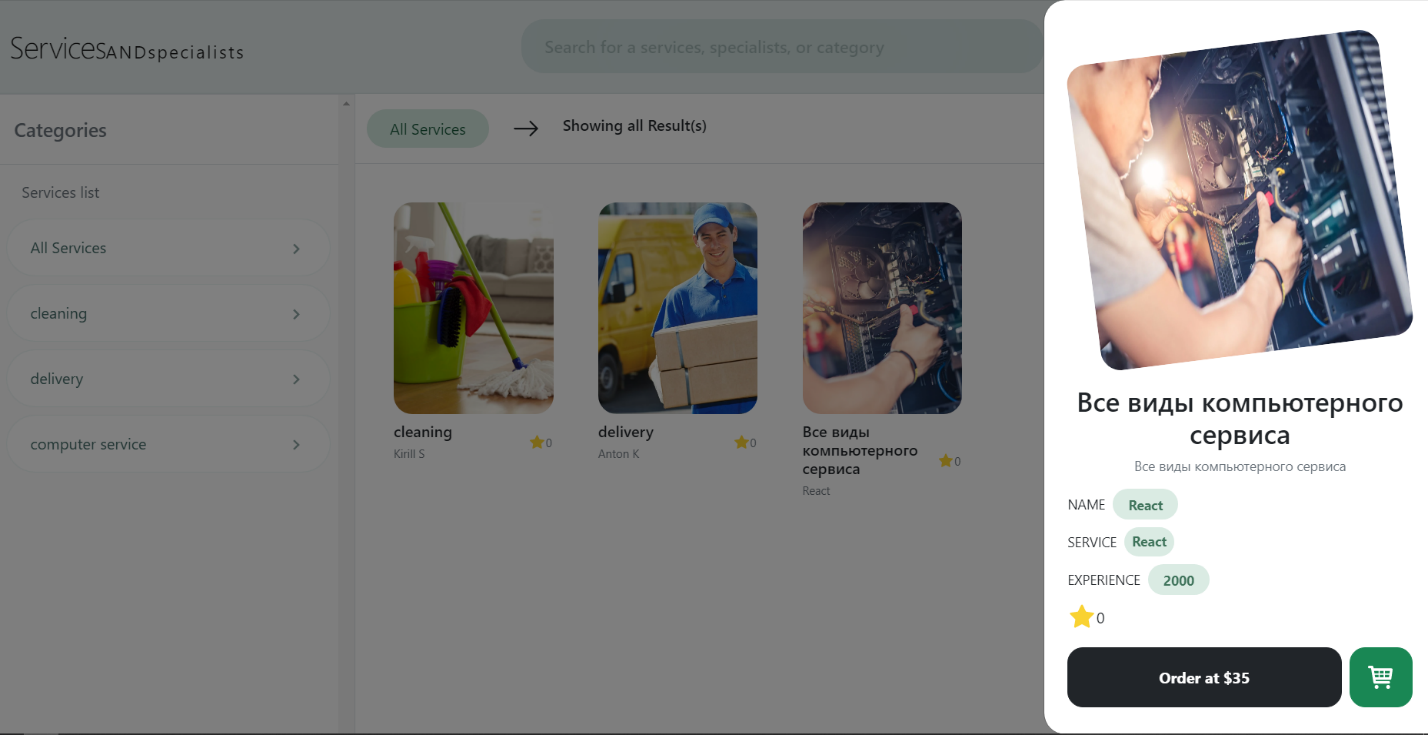


Рисунок 25 – Страница «Добавить в корзину»

Для добавления в корзину необходимо ввести телефон и авторизоваться. Результат продемонстрирован на рисунке 26.

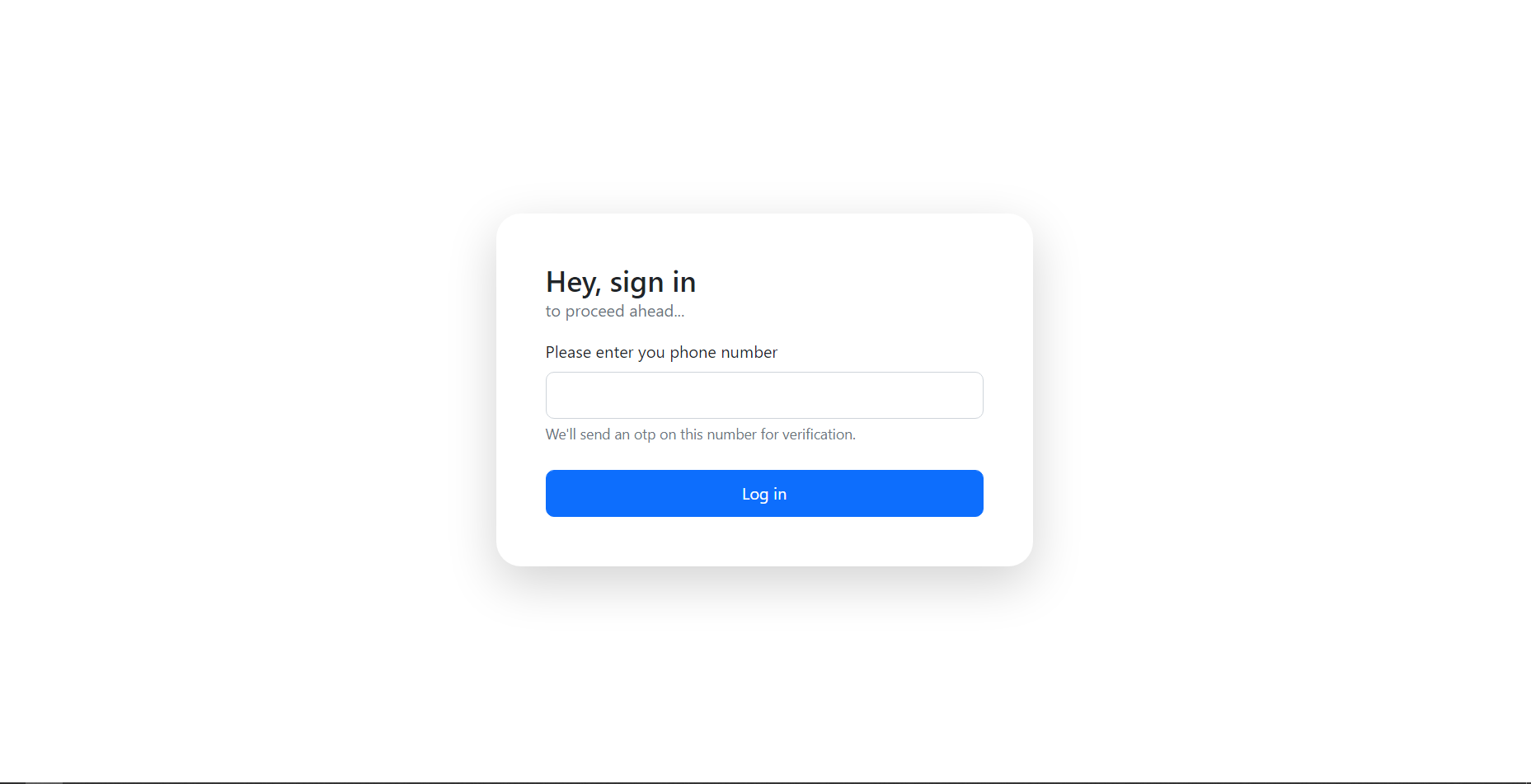


Рисунок 26 – Страница авторизации

После авторизации перебросит в корзину где можно посмотреть выбраные услуги. Результат продемонстрирован на рисунке 27.

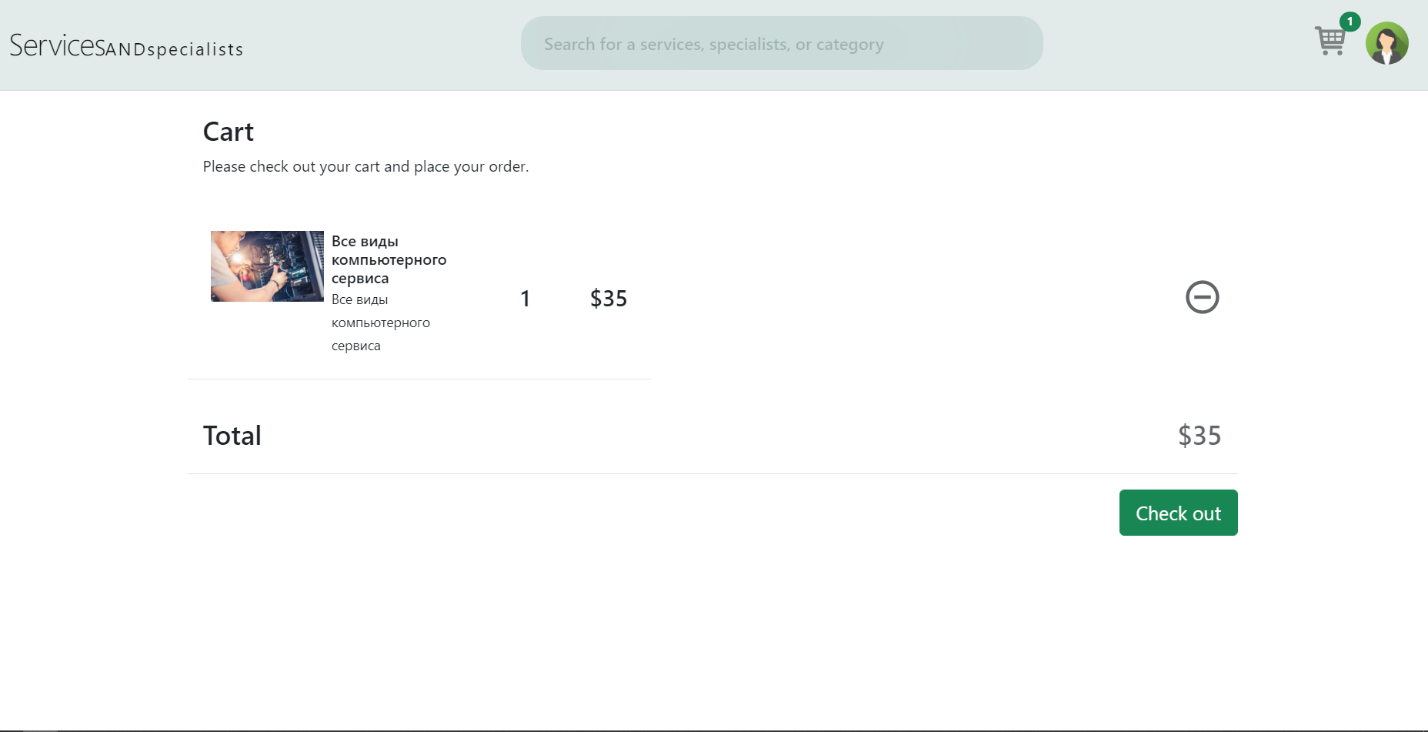


Рисунок 27 – Страница корзины

Представление главной страницы admin page. Результат продемонстрирован на рисунке 28.

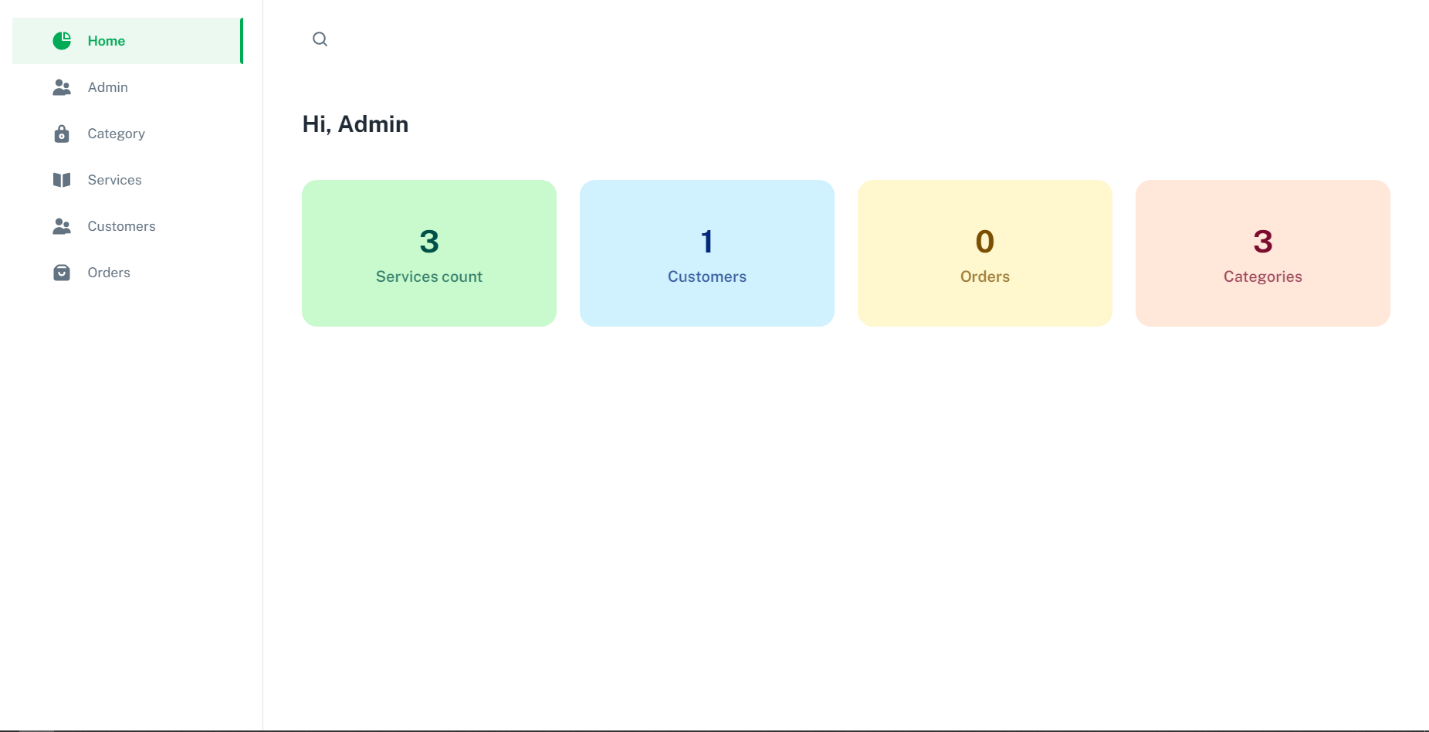


Рисунок 28 – Страница admin

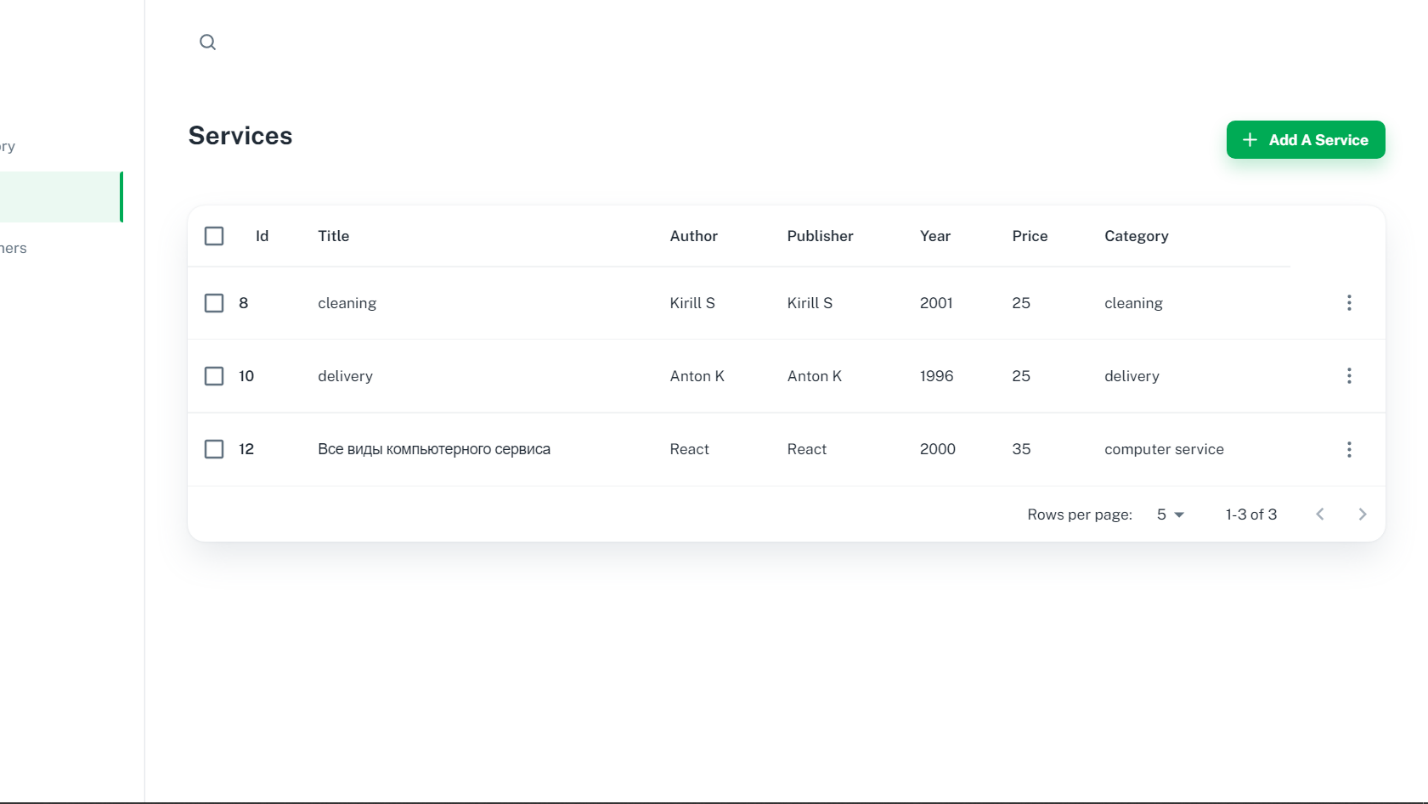
Просмотр услуг в БД. Результат продемонстрирован на рисунке 29.

Рисунок 29 – Страница просмотр услуг

Добавление услуги в БД. Результат продемонстрирован на рисунке 30.

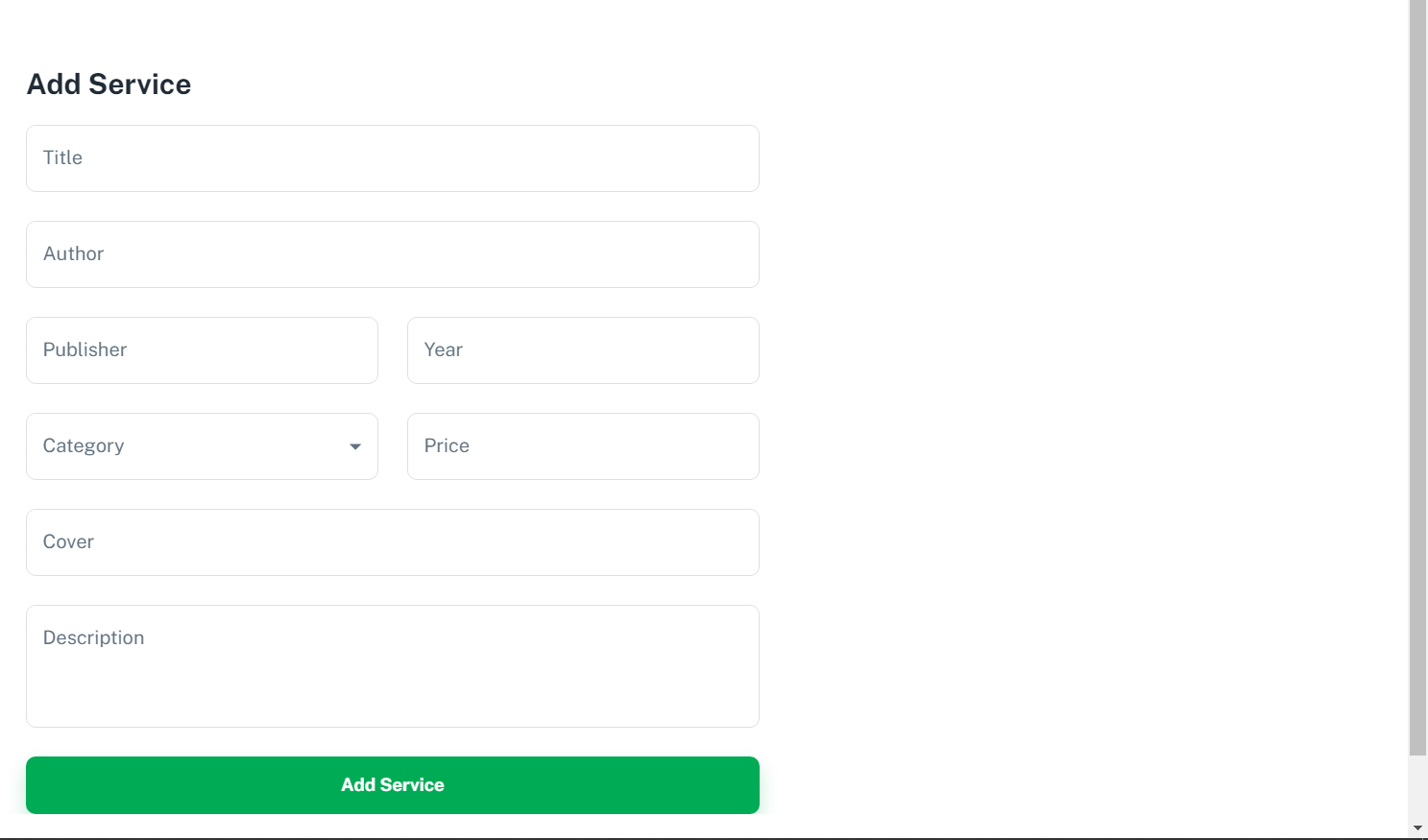


Рисунок 30 – Страница добавления услуги

**Заключение**

В ходе проделанной работы, были выполнены все поставленные задачи, а именно:

– формирование базы данных основных объектов;

– разработка web-приложения (клиентской и серверной части).

В качестве ожидаемых результатов можно выделить:

– создание «дружественного для пользователя» интерфейса;

– вёрстка, адаптированная для трех типов устройств: десктоп, планшет, мобильное устройство;

– корректно функционирующие поиск, выбор по категориям, добавление в корзину.

В результате выполнения данного курсового проекта были закреплены знания в работе с такими библиотеками JavaScript как React и такими инструментами web-программирования, как HTML5, CSS, Java, Spring Framework.

По итогам выполненной работы можно отметить полноту решения сформулированных в техническом задании задач: была проведена конкурентная аналитика: выявлены слабые и сильные стороны приложений-аналогов, задан общий вектор развития разработанного web-приложения; освоен графический редактор Figma, с помощью которого был создан дизайн-макет будущего приложения; создана база данных для реализации приложения.

Среди сильных сторон приложения можно выделить удобство для пользователя, а также корректно функционирующую систему с необходимым функционалом; приятный визуальный дизайн, адаптированный под три типа устройств.

Можно дать следующие рекомендации по дальнейшему развитию:

– добавление корректно работающих ролей, локализации, объединения client-side и admin-side в одну систему;

– добавить историю заказов и автодополнение полей для конкретного пользователя;

– оптимизация поиска по всевозможным параметрам;

– развернуть приложение в докере;

– реализовать микросервисную архитектуру;

– заменить РСУБД на Spring Data MongoDB;

– добавить брокер сообщений Apache Kafka или RabbitMQ;

– добавить балансировщик сетевой нагрузки.

**Список используемых источников**

1 React JavaScript-библиотека для создания пользовательских интерфейсов – Текст : электронный // ReactJS [сайт]. – URL: https://ru.reactjs.org/ (дата обращения: 05.10.2022).

2 Компоненты и пропсы React – Текст : электронный // Docs [сайт]. – URL: https://ru.reactjs.org/docs/components-and-props.html (дата обращения: 05.10.2022).

3 REST with Spring Tutorial – Текст : электронный // Baeldung – блог [сайт]. – URL: <https://www.baeldung.com/rest-with-spring-series> (дата обращения: 15.12.2022).

4 Введение | Axios Docs – Текст : электронный // Axios Docs [сайт]. – URL: https://axios-http.com/ru/docs/intro (дата обращения: 20.09.2022).

5 Figma: the collaborative interface design tool. – Текст : электронный // Figma [сайт]. – URL: https://www.figma.com/ (дата обращения: 15.11.2022).

7 Поиск услуг и специалистов – Текст : электронный // [сайт]. – URL: https://uslugi.yandex.ru/ (дата обращения: 15.12.2022).

8 Сервис поиска надежных специалистов – Текст : электронный // [сайт]. – URL: https://youdo.com/ (дата обращения: 16.12.2022).

9 Сервис поиска специалистов – Текст : электронный // [сайт]. – URL: https://profi.ru/ (дата обращения: 12.12.2022).

10 Основные принципы проектирования пользовательского интерфейса. – Текст : электронный // Файловый архив для студентов. StudFiles [сайт]. – URL: https://studfile.net/preview/423781/ (дата обращения: 29.09.2022).

11 Официальная документация PostgreSQL // PostgreSQL: [сайт]. – URL: https://www.postgresql.org/docs/current/ (дата обращения: 12.12.2022).

12 React Router – Текст : электронный // GeeksForGeeks URL [сайт]. – URL: https://www.geeksforgeeks.org/reactjs-router(дата обращения: 30.11.2022).

14 Руководство по React – Текст : электронный // Metanit URL [сайт]. – URL: https://metanit.com/web/react/ (дата обращения: 31.11.2022).

15 Официальная документация Material UI // Material UI [сайт]. – URL: <https://mui.com/material-ui/getting-started/overview/> (дата обращения: 16.11.2022).

16 Официальная документация Spring. // Spring: [сайт]. – URL: <https://spring.io/guides> (дата обращения: 09.11.2022).

17 Accessing Relational Data using JDBC with Spring // Spring: [сайт]. – URL: https://spring.io/guides/gs/relational-data-access/ (дата обращения: 25.10.2022).