

МИНОБРНАУКИ РОССИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

Факультет компьютерных наук

Кафедра информационных технологий управления

Курсовая работа по дисциплине «Технологии программирования»  
Веб-приложение для проката инвентаря на спортивной базе «SportBox»

Направление 09.03.02 Информационные системы и технологии

Преподаватель \_\_\_\_\_ В.С. Тарасов, ст. преподаватель \_\_\_\_\_.\_\_20\_\_

Обучающийся \_\_\_\_\_ И.Р. Корнилов, 3 курс, д/о

Обучающийся \_\_\_\_\_ Е.А. Бабкина, 3 курс, д/о

Обучающийся \_\_\_\_\_ И.А. Кандауров, 3 курс, д/о

Воронеж 2023

## Содержание

Содержание.....	2
Введение.....	4
1 Постановка задачи.....	6
1.1 Постановка задачи .....	6
1.2 Обзор аналогов.....	7
1.2.1 Спортмастер .....	7
1.2.2 Ялмал.....	8
1.2.3 Робинзон .....	8
2 Анализ предметной области .....	10
2.1 Терминология (гlossарий) предметной области .....	10
2.2 Пользовательские истории.....	11
2.2.1 Пользовательская история 1 .....	11
2.2.2 Пользовательская история 2 .....	11
2.2.3 Пользовательская история 3 .....	11
2.2.4 Пользовательская история 4 .....	12
2.2.5 Пользовательская история 5 .....	12
2.3 Продуктовые воронки.....	13
2.4 Диаграммы, иллюстрирующие работу системы.....	15
2.4.1 Диаграмма IDEF0.....	15
2.4.2 Диаграмма прецедентов .....	16
2.4.3 Диаграмма активностей.....	18
2.4.4 Диаграмма последовательности .....	19
2.4.5 Диаграмма развертывания .....	21
2.4.6 Диаграмма сотрудничества.....	22

2.4.7 Диаграмма классов .....	22
2.4.8 Диаграмма объектов .....	26
2.4.9 Диаграмма состояний .....	26
2.4.10 ER-диаграмма .....	27
2.4.11 Физическая схема базы данных.....	28
3 Реализация.....	30
3.1 Средства реализации.....	30
3.2 Реализация backend .....	32
3.3 Реализация frontend.....	36
3.4 Навигация по приложению .....	39
3.4.1 Для клиента .....	39
3.4.2 Для администратора .....	44
3.5 Тестирование .....	46
3.5.1 Модульное тестирование .....	46
3.5.2 Дымовое тестирование .....	47
3.5.3 Тестирование пользовательского интерфейса .....	49
3.5.4 Usability тестирование .....	51
Заключение .....	53
Список использованных источников .....	54

## **Введение**

Спорт – это жизнь. Врачи постоянно рекомендуют занятия спортом для укрепления здоровья. Подавляющее большинство людей занимаются как минимум одним видом спорта, и не только для собственно здоровья. Для кого-то это работа, а кому-то просто нравится проводить так своё время и для них это отдых. Для некоторых видов спорта нужно лишь желание, например, для бега достаточно просто выйти на улицу. Но для большинства видов спорта необходимо различное оборудование: мячи, копья, луки, лыжи и т.д. Разновидностей спортивного инвентаря очень много. Но далеко не у всех дома есть необходимое оборудование, для таких ситуаций существует аренда спортивного инвентаря. Благодаря такой возможности, люди могут арендовать необходимое оборудование и заниматься выбранным видом спорта.

В случае, когда необходимо приходить в организацию, чтобы арендовать инвентарь, можно столкнуться с очередями. Кроме того, может не оказаться того, что Вы хотели, и окажется, что Вы пришли зря. Эти проблемы решаются созданием для сервиса аренды инвентаря своего сайта или приложения, особенно в наше время, когда у большинства компаний используют информационные технологии в своей работе.

Сайт или приложение может быть также интегрировано в работу сотрудников организации. Ведение бумажного учета инвентаря, заказов и клиентов вручную затрачивает денежные (оплата бумаги и труда работников) и временные ресурсы. Кроме того, общение с клиентами посредством телефонных звонков или личного обращения в организацию также затрачивает денежные (оплата телефонных звонков и оплаты труда персонала) и временные ресурсы.

Таким образом, целью нашей работы является разработка сайта для сервиса по сдаче в аренду спортивного инвентаря, чтобы пользователи могли арендовать инвентарь удалённо, по приходе в организацию не тратить время на очереди и сразу начать заниматься выбранным видом спорта.

Но также стоит отметить, что приложение разрабатывается не только для обычных пользователей, но и для работников сервиса. Поэтому существуют специальные аккаунты для работников, в которых они могут просматривать и редактировать информацию о заказах, инвентаре и клиентах.

## **1 Постановка задачи**

### **1.1 Постановка задачи**

Целью данного проекта, поставленной заказчиком перед разработчиками, является сокращение материальных расходов и затраченного времени относительно приема заказов:

- Посредством телефонных звонков клиентов на спортивную базу на стоимость и время телефонных звонков;
- Посредством личного обращения клиентов на спортивную базу на стоимость оплаты работы персонала и время обслуживания клиентов.

Кроме того, целью является сокращение материальных расходов и затраченного времени относительно ручного бумажного ведения учета спортивного оборудования, мероприятий, заказов, клиентов на стоимость бумажных ресурсов, оплаты работы персонала, а также на время, необходимое для учета.

Система будет представлять собой backend (серверную) часть и frontend (клиентскую) часть, которые взаимодействуют с помощью REST API. Backend часть отвечает за обработку запросов, работу с базой данных, управление доступом пользователей, логику расчета стоимости проката, составлением рекомендаций. Frontend часть отвечает за отображение информации на сайте в браузере.

Система решает следующие задачи:

- Формирование клиентом заказа на прокат спортивного инвентаря;
- Расчет стоимости проката;
- Создание формы для оплаты заказа;
- Предоставление клиенту рекомендаций по посещению спортивных мероприятий, формируемых на основе данных о заказе;
- Сохранение истории заказов клиента;
- Просмотр и редактирование сотрудником компании данных, относящихся к прокату оборудования;

— Добавление клиентов в «черный список» за нарушение условий аренды (порчу оборудования и/или неуплату аренды).

## 1.2 Обзор аналогов

### 1.2.1 Спортмастер

Спортмастер — это крупная компания, продающая спортивные товары и имеющая множество магазинов по всей стране. Совместно с сервисом «Арентер», который предоставляет услуги аренды самых различных товаров, имеет договор о возможности предоставления аренды на спортивные товары и сайт для этого. Сайт предлагает широкий выбор товаров и обладает возможностью поиска нужного спортивного оборудования.

Из минусов стоит отметить, что регистрация на сайте возможна только через оформление заказа. Кроме того, при заказе пользователю не предлагаются спортивные мероприятия, которые он может посетить. Также отсутствует система блокировки пользователей, сотрудничество с которыми приносит компании только убытки из-за того, что они нарушают условия аренды.

На рисунке 1 представлен скриншот сайта Спортмастер.

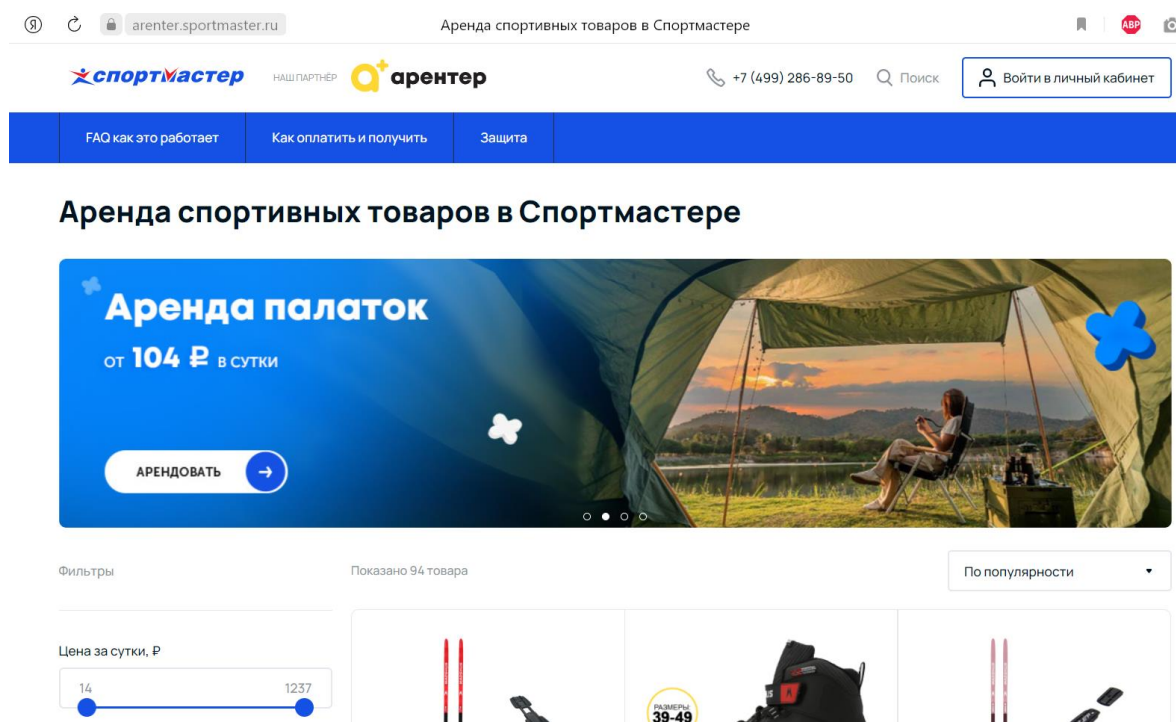


Рисунок 1 - Спортмастер и Арентер

### 1.2.2 Ялмад

Ялмад — сервис для аренды различных вещей в Воронеже. Сайт предлагает оборудование от инструментов и мебели до лодок и предоставляет возможность заказывать аренду в различные города. Количество единиц товара ограничено, поэтому высока вероятность не найти то, что Вам нужно.

Из минусов стоит отметить, на сайте отсутствует авторизация. Это означает, что клиенту недоступна история своих заказов. Кроме того, при заказе пользователю не предлагаются спортивные мероприятия, которые он может посетить. Также отсутствует система блокировки пользователей, сотрудничество с которыми приносит компании только убытки из-за того, что они нарушают условия аренды.

На рисунке 2 представлен скриншот сайта Ялмад.

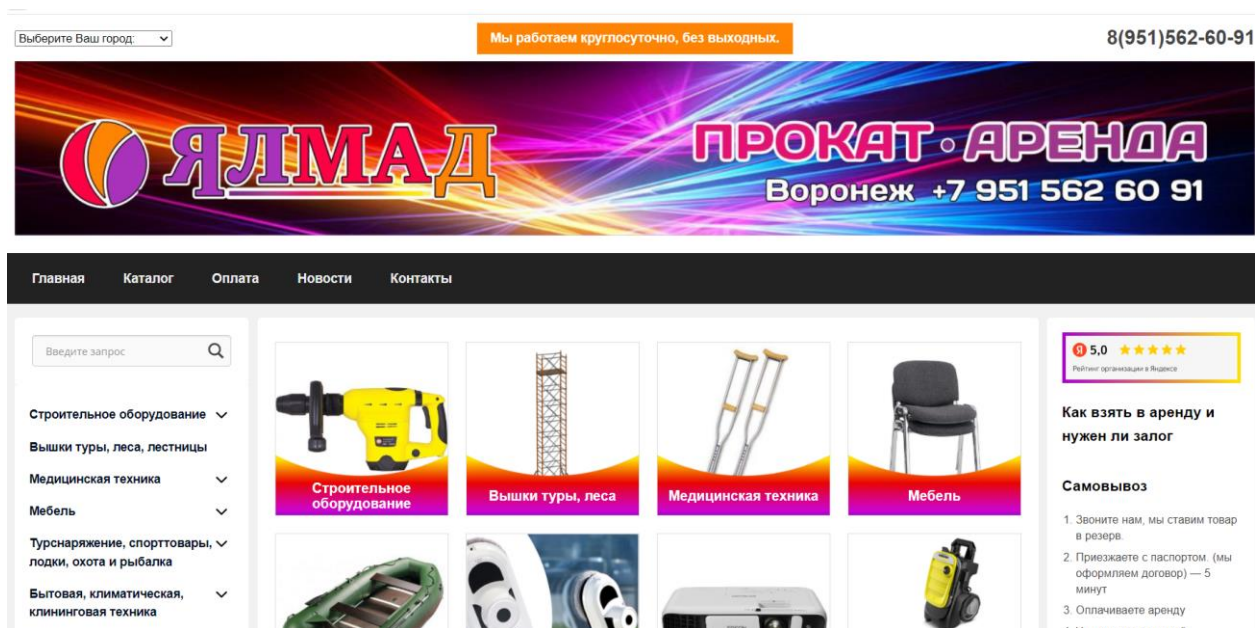


Рисунок 2 - Ялмад

### 1.2.3 Робинзон

Робинзон — сервис для аренды оборудования, направленного на активный отдых на природе, такой как рыбалка, охота или туризм. Сервис сконцентрирован на конкретной теме и не перегружен посторонними вещами, но ассортимент товаров ограничен. Клиентам предлагаются гибкие условия



аренды, включая длительность и стоимость аренды, а также возможность отмены заказа.

Из минусов стоит отметить, что на сайте отсутствует авторизация. Заказы возможны только путем телефонного звонка. Соответственно, нельзя посмотреть историю заказов. Кроме того, отсутствует синхронизация с мероприятиями и система блокировки пользователей.

На рисунке 3 представлен скриншот сайта Робинзон.

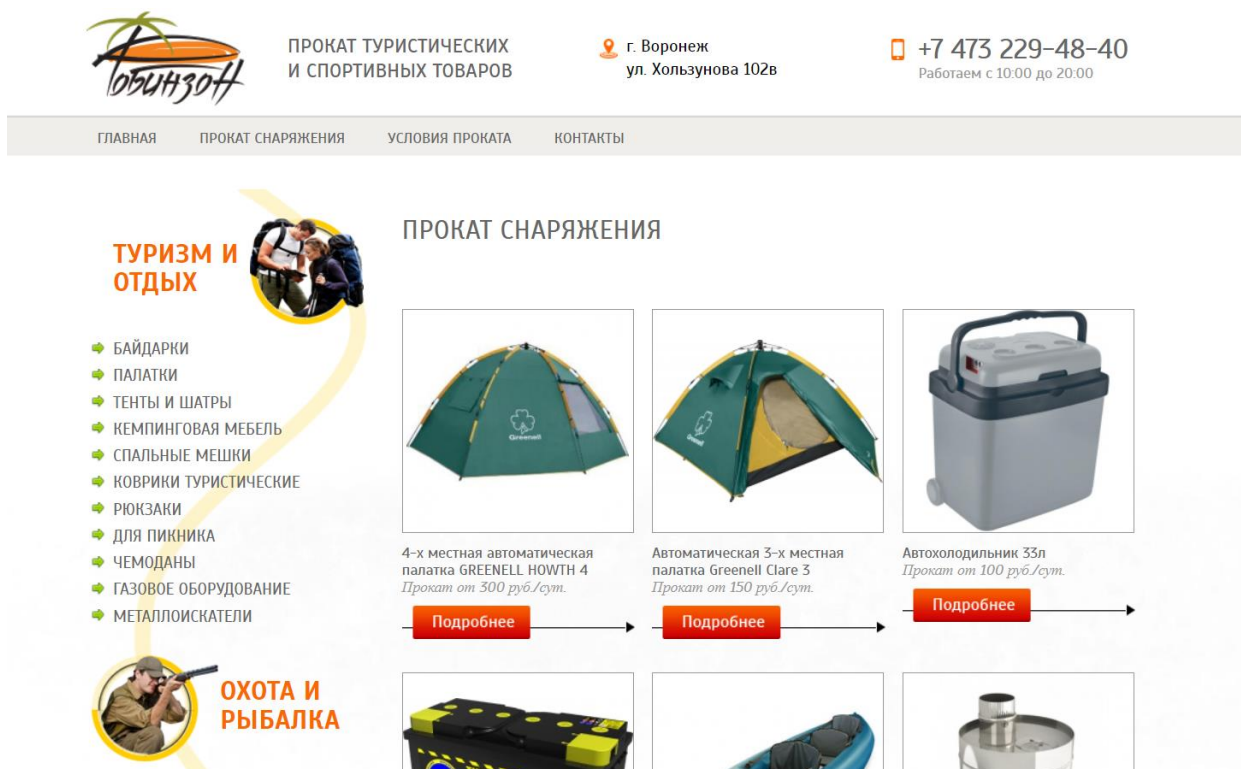


Рисунок 3 - Робинзон

## **2 Анализ предметной области**

### **2.1 Терминология (гlossарий) предметной области**

Спортивная база — организация, имеющая в собственности сооружение или комплекс зданий и сооружений, предоставляющая услуги по прокату спортивного инвентаря, а также занимающаяся организацией спортивных мероприятий.

Спортивный инвентарь — устройство, приспособление узкоспециального назначения, используемое при занятии различными видами спорта. [1]

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

MVC (Model-View-Controller) — схема разделения данных приложения и управляющей логики на три отдельных компонента: модель, представление и контроллер — таким образом, что модификация каждого компонента может осуществляться независимо. Контроллер обрабатывает входящие запросы. Модель отвечает за данные, которые хранятся и обрабатываются на сервере. Представление определяет результат запроса, который получает пользователь. [2]

REST (Representational state transfer) API — это архитектурный стиль, который определяет правила обмена данными между клиентом и сервером.

Backend — логика работы сайта, внутренняя часть продукта, которая находится на сервере и скрыта от пользователя. [3]

Frontend — презентационная часть информационной или программной системы, ее пользовательский интерфейс и связанные с ним компоненты. [4]

Splash screen — это экран, который появляется при запуске приложения или загрузке веб-страницы и предназначен для показа пользователю информации о процессе инициализации приложения или загрузки контента.

Эндпоинт (endpoint) — это конечная точка или URL веб-сервиса или API, к которой можно обратиться для выполнения определенного запроса или получения определенной информации. Эндпоинты определяют, какие

операции и ресурсы доступны в системе и каким образом они могут быть использованы.

## **2.2 Пользовательские истории**

### **2.2.1 Пользовательская история 1**

Пользователь: Анатолий Соловьев, 45 лет

Описание: У Анатолия есть сын. Анатолий любит провести зимние выходные, катаясь на лыжах. У него есть собственные лыжи, но его сын никогда не пробовал лыжный спорт. По совету друга Анатолий первый раз обратился на базу проката спортивного инвентаря.

Пользовательская история: Анатолию необходимо взять в прокат набор лыж для своего сына, с целью приобщить его к лыжному спорту. Анатолий не хочет пока покупать лыжи для сына, так как ребенку может не понравиться лыжный спорт после того, как он попробует. Выгоднее взять набор лыж в прокат.

### **2.2.2 Пользовательская история 2**

Пользователь: Михаил Рыков, 26 лет

Описание: Михаил — опытный велосипедист. У него есть собственный велосипед в городе, в котором он живет. Михаил на некоторое время уехал к друзьям в другой город и оказался без своего велосипеда. Михаил предложил своим друзьям-велосипедистам провести время, просто катаясь на велосипедах, или принять участие в соревнованиях, если они будут проходить в даты его присутствия в городе.

Пользовательская история: Михаилу необходимо взять в прокат велосипед, чтобы прокатиться с друзьями или принять участие в соревнованиях, если они будут проходить в эти даты.

### **2.2.3 Пользовательская история 3**

Пользователь: Оксана Котельникова, 31 год

Описание: Оксана — администратор, работающий на базе проката спортивного инвентаря. Она работает в компании 5 лет, отличается трудолюбием, ответственным подходом к своей работе.

Пользовательская история: Оксане необходимо добавить в черный список пользователя с плохой репутацией клиента, чтобы компания в будущем не понесла убытки, если он повторно сломает или не вернет полученный в аренду спортивный инвентарь. Данный пользователь зарекомендовал себя как человек с низкой социальной ответственностью; компании не выгодно давать ему в прокат оборудование.

#### **2.2.4 Пользовательская история 4**

Пользователь: Сергей Федоров, 32 года

Описание: Сергей — администратор, работающий на базе проката спортивного инвентаря. Он работает в компании 3 года, занимается контролем и инвентаризацией спортивного оборудования.

Пользовательская история: Сергею после обновления некоторых позиций инвентаря на базе необходимо изменить данные о них в таблице базы данных, чтобы пользователи получали актуальную информацию об оборудовании, находящемся в наличии в организации. Если он не сделает этого, возникнут логистические ошибки, что приведет к появлению недовольных клиентов и, соответственно, убытков.

#### **2.2.5 Пользовательская история 5**

Пользователь: Мария Семенова, 31 год

Описание: Мария — пользователь, до этого не обращавшийся на спортивную базу. Она решила попробовать себя в зимних видах спорта и ищет организацию, предлагающую возможности, которые удовлетворят ее запросы. Мария — ответственный покупатель, серьезно подходящий к вопросу выбора спортивного инвентаря.

Пользовательская история: Марии необходимо исследовать ассортимент спортивной базы, цены и предлагаемые возможности, чтобы

решить, будет ли она брать оборудование в прокат на этой базе или поищет другую, более подходящую. До того, как авторизоваться и начать заказывать инвентарь, Мария хочет ознакомиться с каталогом и информацией об организации.

### **2.3 Продуктовые воронки**

Для неавторизованного пользователя:

- Просмотр оборудования: зайти на сайт, перейти на страницу с оборудованием;
- Просмотр мероприятий: зайти на сайт, перейти на страницу с мероприятиями;
- Авторизация: зайти на сайт, перейти на страницу входа, войти;
- Регистрация: зайти на сайт, перейти на страницу входа, перейти на страницу регистрации, зарегистрироваться.

Для авторизованного пользователя:

- Просмотр оборудования: зайти на сайт, перейти на страницу с оборудованием;
- Просмотр мероприятий: зайти на сайт, перейти на страницу с мероприятиями;
- Оплата заказа: зайти на сайт, перейти на страницу с оборудованием, перейти на страницу аренды конкретного оборудования, перейти на страницу оплаты, оплатить;
- Просмотр истории заказов: зайти на сайт, перейти на страницу профиля;
- Отмена заказа: зайти на сайт, перейти на страницу профиля, отменить заказ.

Для администратора:

- Просмотр оборудования: зайти на сайт, перейти на страницу с оборудованием;

- Просмотр мероприятий: зайти на сайт, перейти на страницу с мероприятиями;
- Просмотр заказов: зайти на сайт, перейти на страницу с заказами;
- Просмотр клиентов: зайти на сайт, перейти на страницу с клиентами;
- Добавление клиента в черный список: зайти на сайт, перейти на страницу с клиентами, заблокировать клиента;
- Вынесение клиента из черного списка: зайти на сайт, перейти на страницу с клиентами, разблокировать клиента;
- Изменение оборудования: зайти на сайт, перейти на страницу с оборудованием, перейти на страницу с изменением оборудования, изменить;
- Изменение мероприятий: зайти на сайт, перейти на страницу с мероприятиями, перейти на страницу с изменением мероприятия, изменить;
- Изменение заказов: зайти на сайт, перейти на страницу с заказами, перейти на страницу с изменением заказа, изменить;
- Добавление оборудования: зайти на сайт, перейти на страницу с оборудованием, перейти на страницу с добавлением оборудования, добавить;
- Добавление мероприятий: зайти на сайт, перейти на страницу с мероприятиями, перейти на страницу с добавлением мероприятия, добавить;
- Добавление клиентов: зайти на сайт, перейти на страницу с клиентами, перейти на страницу с добавлением клиента, добавить;
- Удаление оборудования: зайти на сайт, перейти на страницу с оборудованием, удалить;
- Удаление мероприятий: зайти на сайт, перейти на страницу с мероприятиями, удалить;

— Удаление клиентов: зайти на сайт, перейти на страницу с клиентами, удалить.

Для заблокированного пользователя:

— Попытка войти в профиль: зайти на сайт, перейти на страницу входа, попытаться войти.

## 2.4 Диаграммы, иллюстрирующие работу системы

### 2.4.1 Диаграмма IDEF0

Диаграмма IDEF0 используется для создания функциональной модели, отображающей структуру и функции системы, а также потоки информации и материальных объектов, связывающие эти функции.

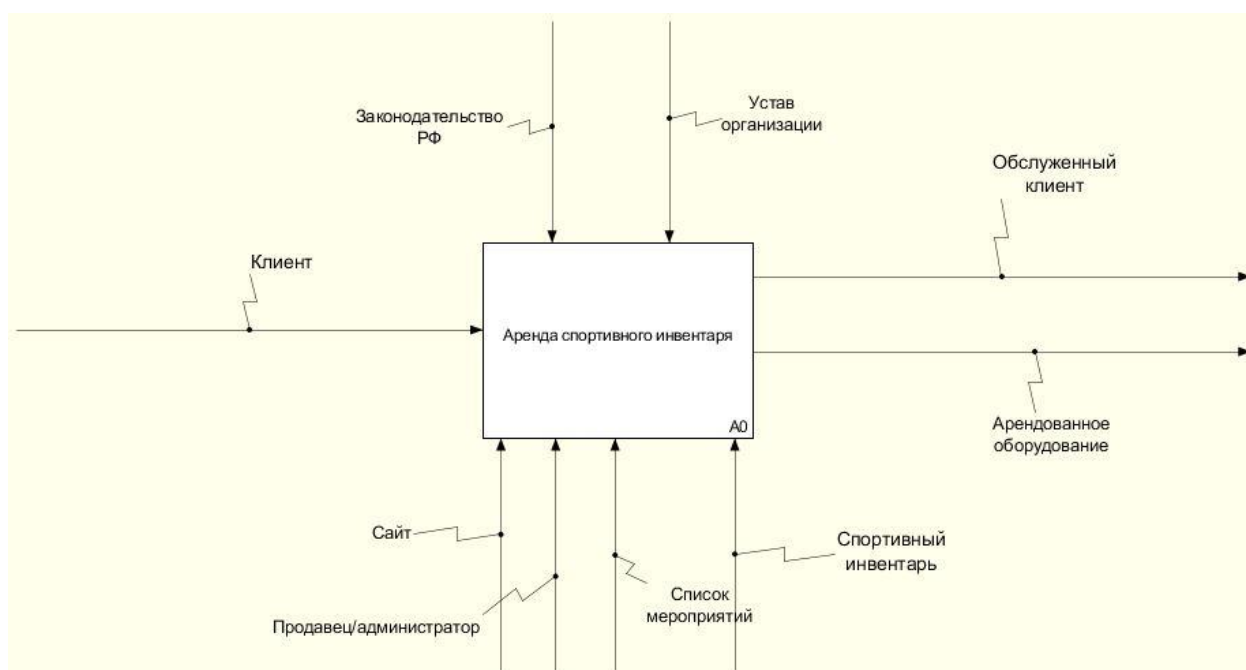


Рисунок 4 - Диаграмма IDEF0 (0 уровень декомпозиции)

На рисунке 4 изображена диаграмма IDEF0. На вход системе поступает клиент, желающий произвести аренду спортивного оборудования. Клиент может быть, как авторизованным, так и неавторизованным. Работу системы регулируют законодательство РФ и устав организации. Как ресурсы, необходимые для работы системы, в неё поступают сайт, продавец/администратор, спортивный инвентарь и список мероприятий. На

выходе системы мы имеем обслуженного клиента и арендованное оборудование.

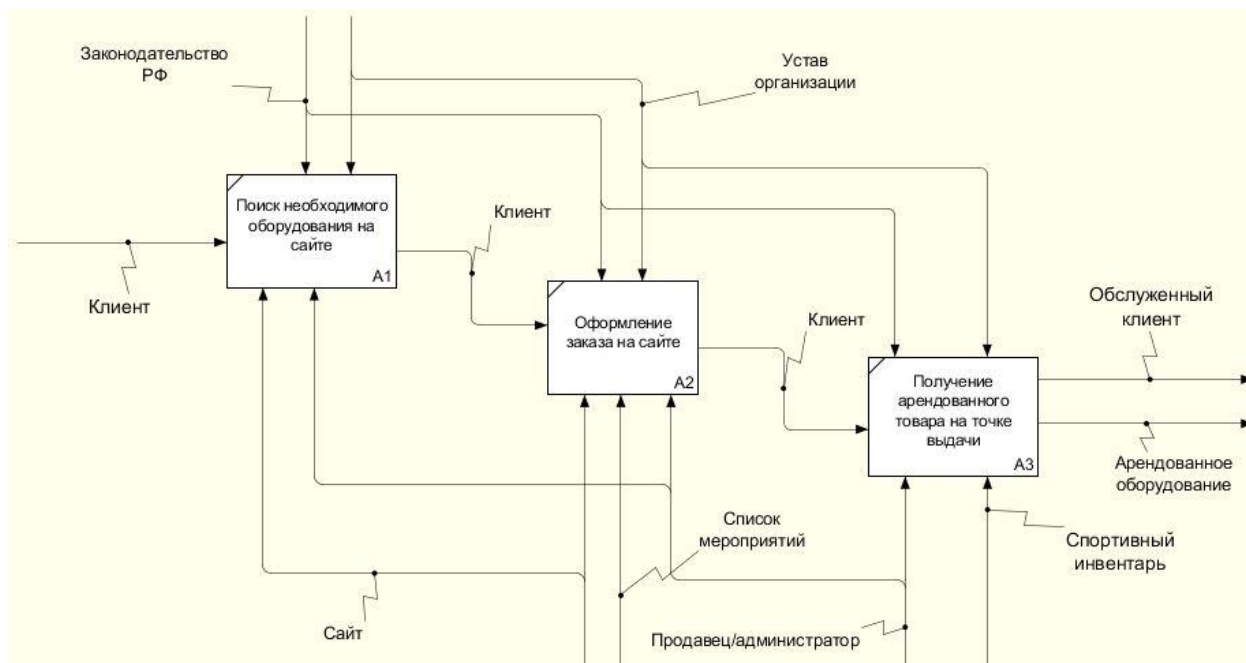


Рисунок 5 - Диаграмма IDEF0 (1 уровень декомпозиции)

На рисунке 5 изображен 1 уровень декомпозиции для диаграммы IDEF0, раскрывающий более подробно процесс аренды спортивного инвентаря.

#### 2.4.2 Диаграмма прецедентов

Диаграмма прецедентов позволяет визуализировать различные типы ролей в системе и то, как эти роли взаимодействуют с системой. На диаграмме, изображенной на рисунке 6, присутствует 4 актёра: неавторизованный, авторизованный, заблокированный пользователи, администратор.



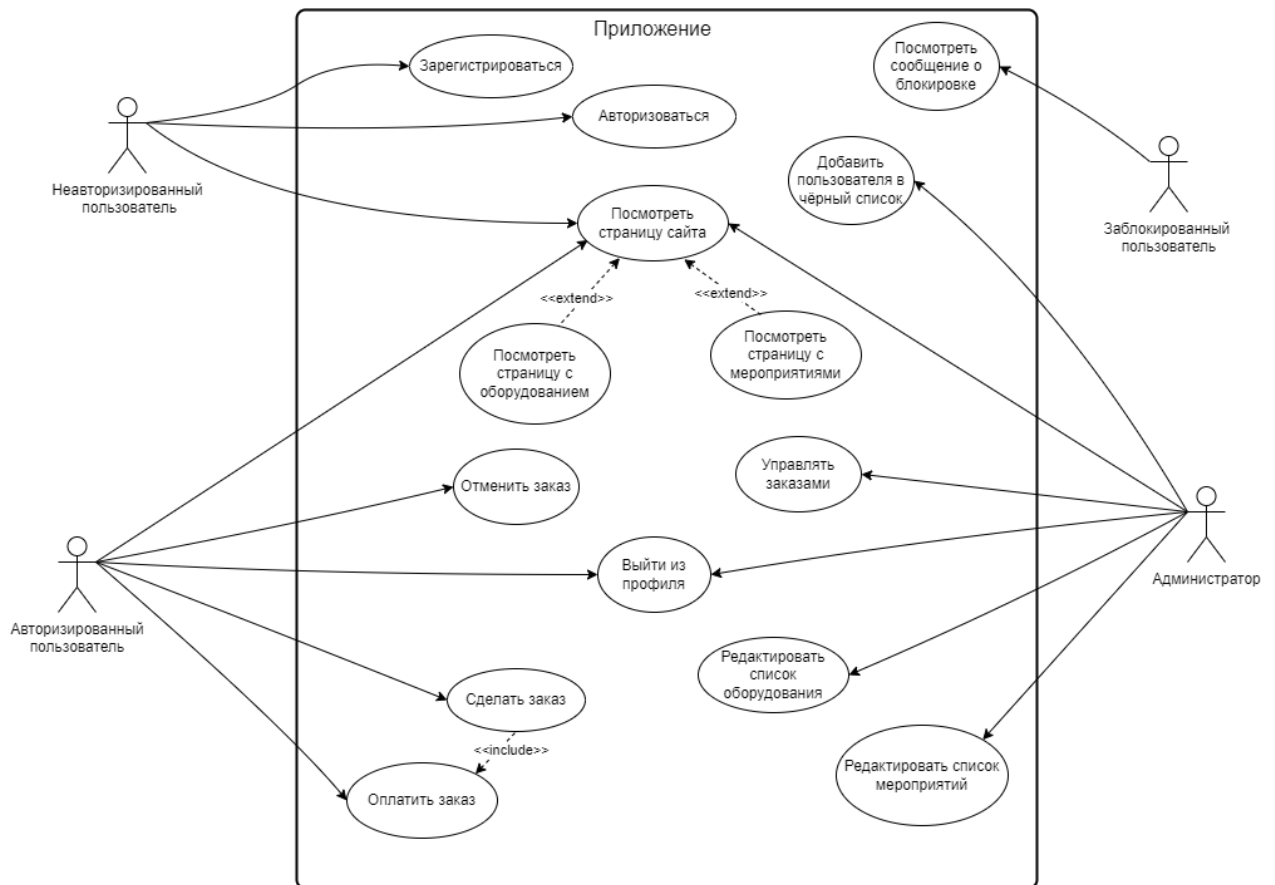


Рисунок 6 - Диаграмма прецедентов

Функции неавторизованного пользователя:

- Зарегистрироваться;
- Авторизоваться;
- Посмотреть страницу сайта.

Функции авторизованного пользователя:

- Посмотреть страницу сайта;
- Сделать заказ;
- Оплатить заказ;
- Отменить заказ.

Функции администратора (пользователя системы, обладающего особыми правами):

- Авторизоваться;
- Посмотреть страницу сайта;
- Управлять заказами;

- Редактировать список оборудования;
- Редактировать список мероприятий;
- Добавить пользователя в чёрный список.

### 2.4.3 Диаграмма активностей

Диаграмма активностей — это поведенческая диаграмма, которая иллюстрирует поток деятельности через систему. В данном случае диаграмма, представленная на рисунке 7, позволяет проиллюстрировать действия незарегистрированного и зарегистрированного пользователя при аренде инвентаря.

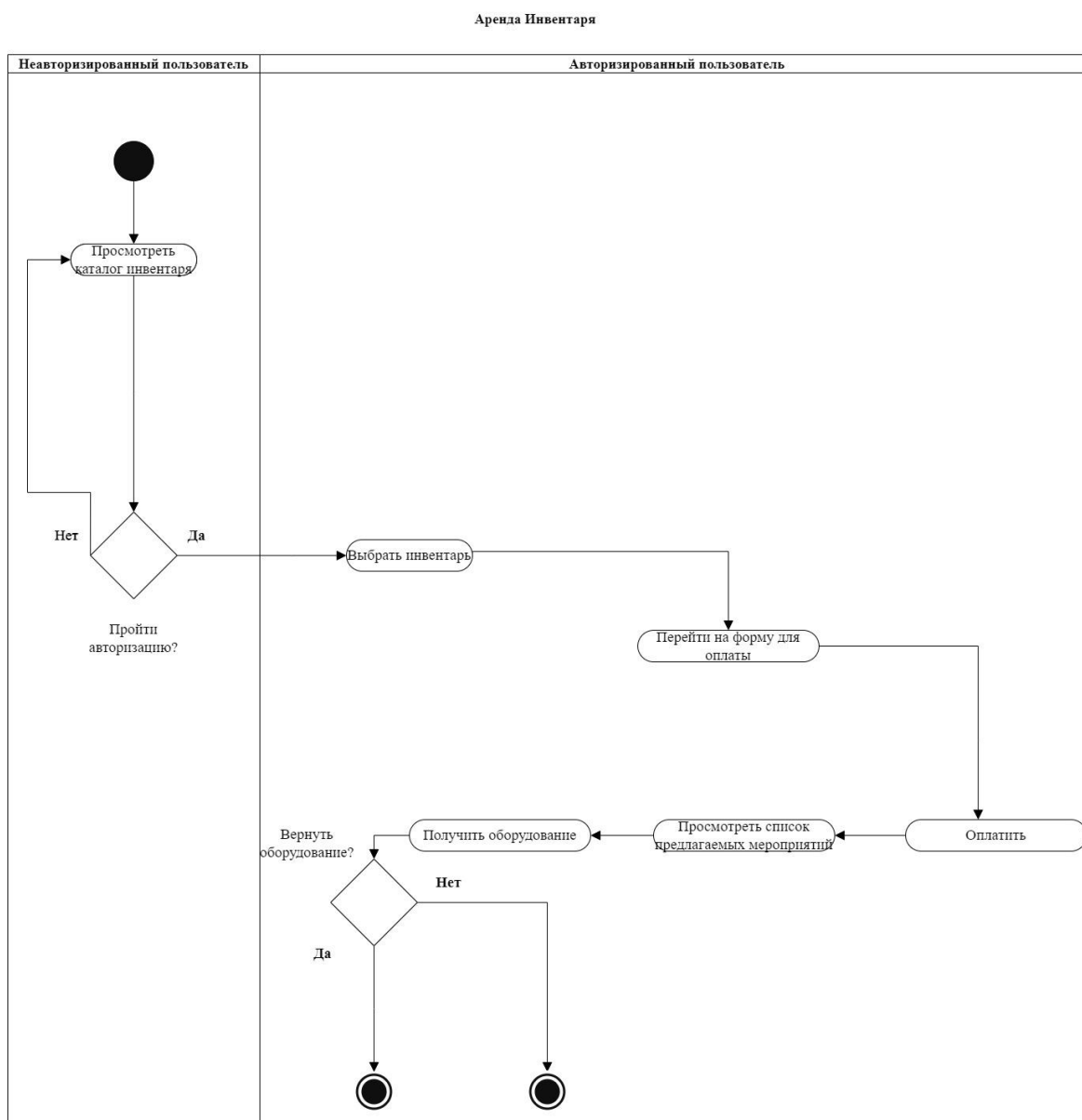


Рисунок 7 - Диаграмма активностей

Из диаграммы видно, что при входе на сайт пользователь может пройти авторизацию и получить возможность арендовать оборудование или продолжить взаимодействовать с сайтом в роли неавторизованного пользователя, имеющего только возможность просмотра.

#### 2.4.4 Диаграмма последовательности

Диаграмма последовательности иллюстрирует, как различные части системы взаимодействуют друг с другом для выполнения функции, а также порядок, в котором происходит взаимодействие при выполнении конкретного случая использования.

На рисунках 8-10 представлены диаграммы последовательности для трех основных актеров системы.

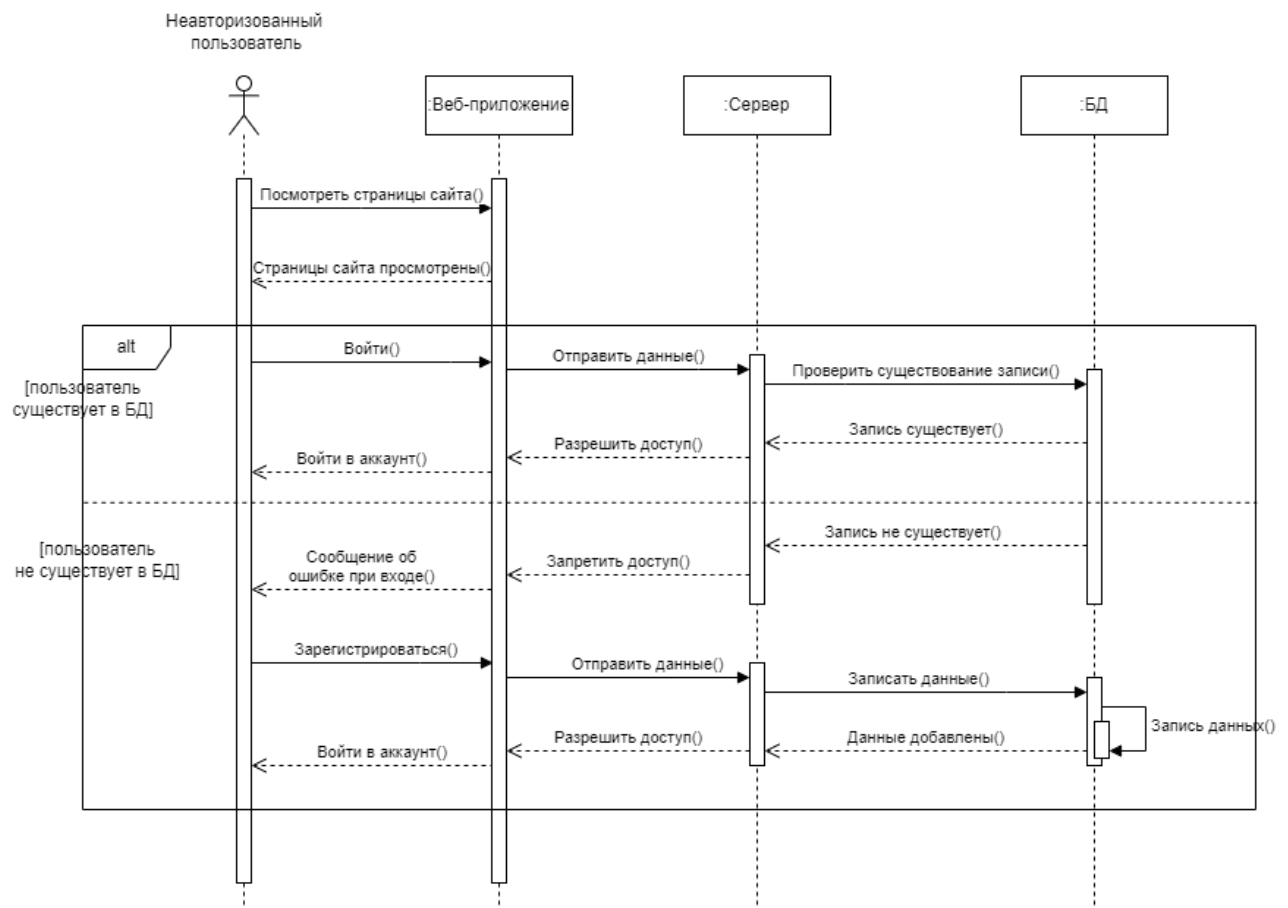


Рисунок 8 - Диаграмма последовательности для неавторизованного пользователя

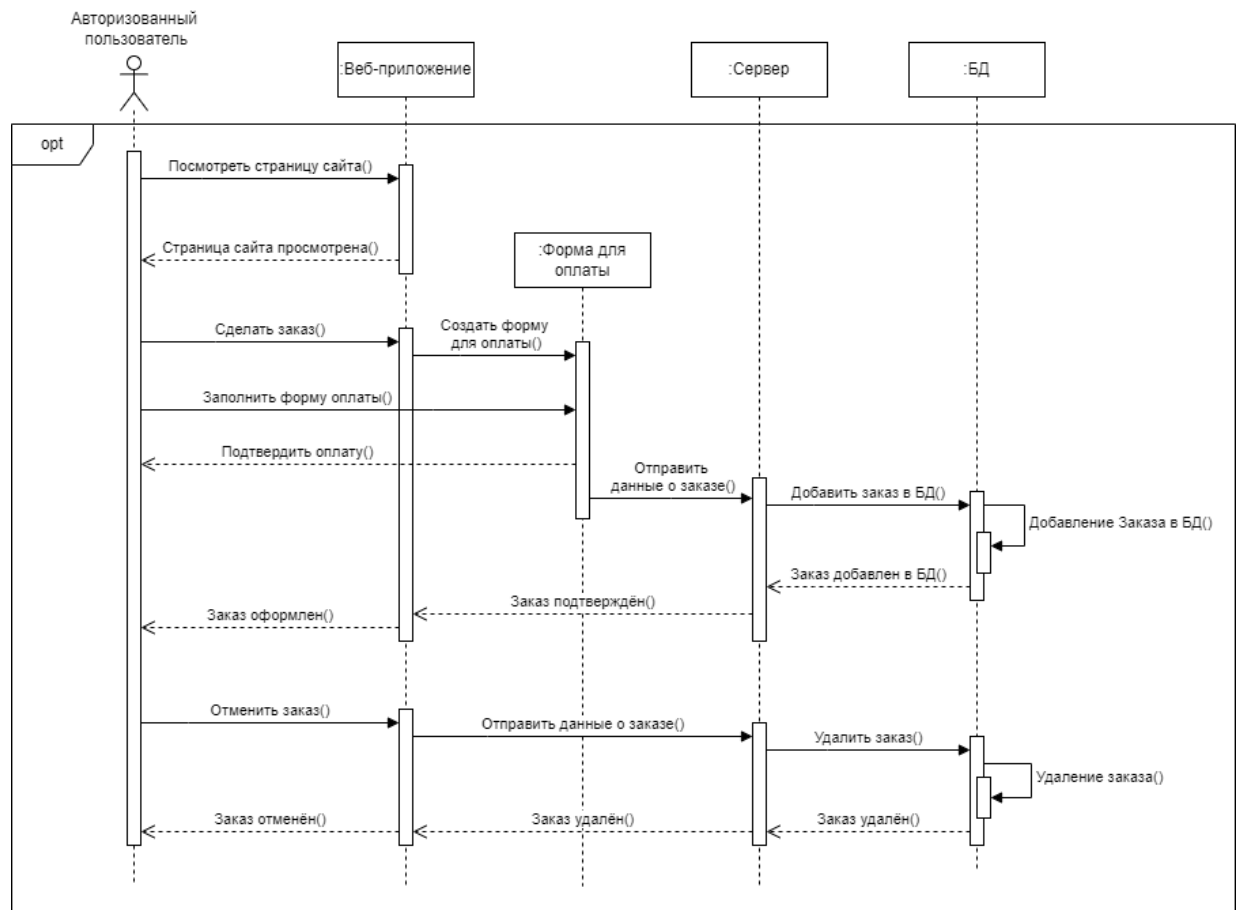


Рисунок 9 - Диаграмма последовательности для авторизованного пользователя

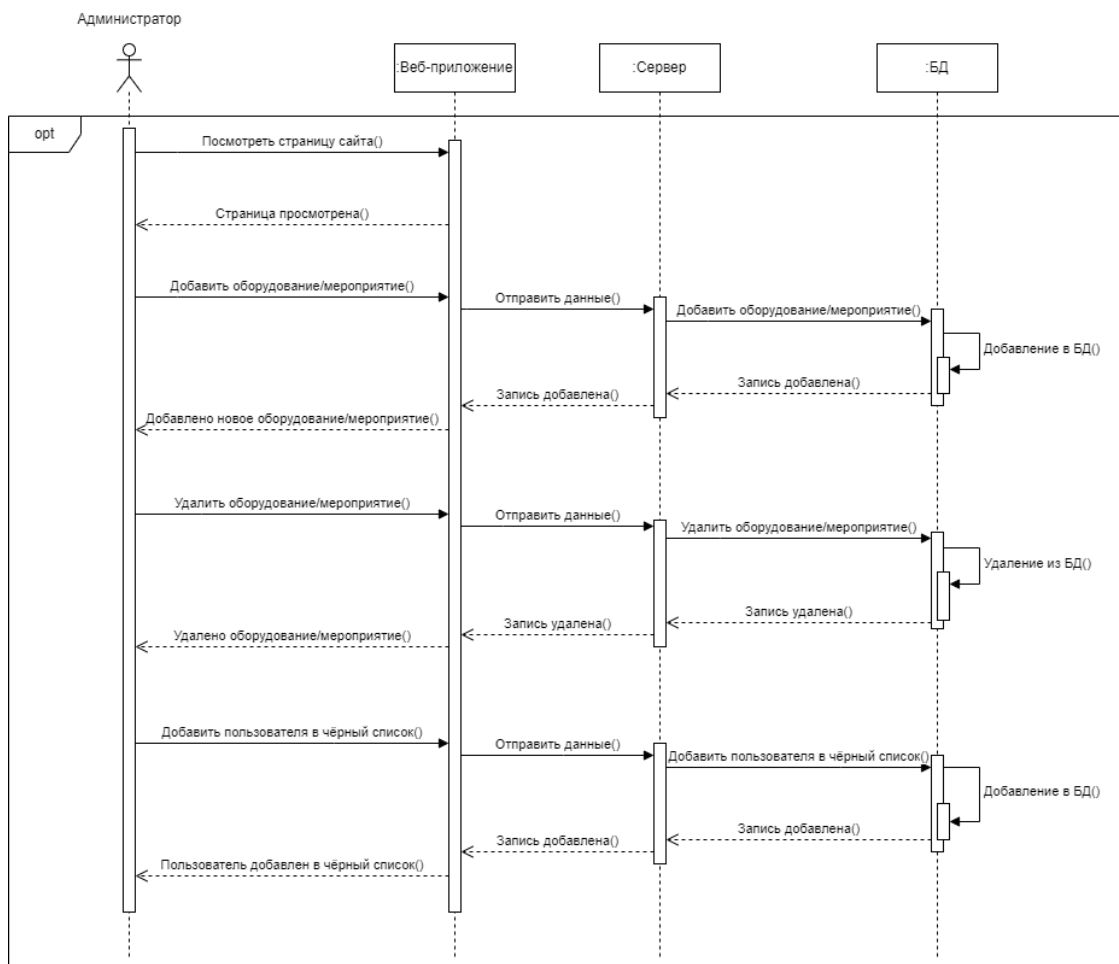


Рисунок 10 - Диаграмма последовательности для администратора

## 2.4.5 Диаграмма развертывания

Диаграмма развертывания предназначена для представления общей конфигурации или топологии распределенной программной системы.

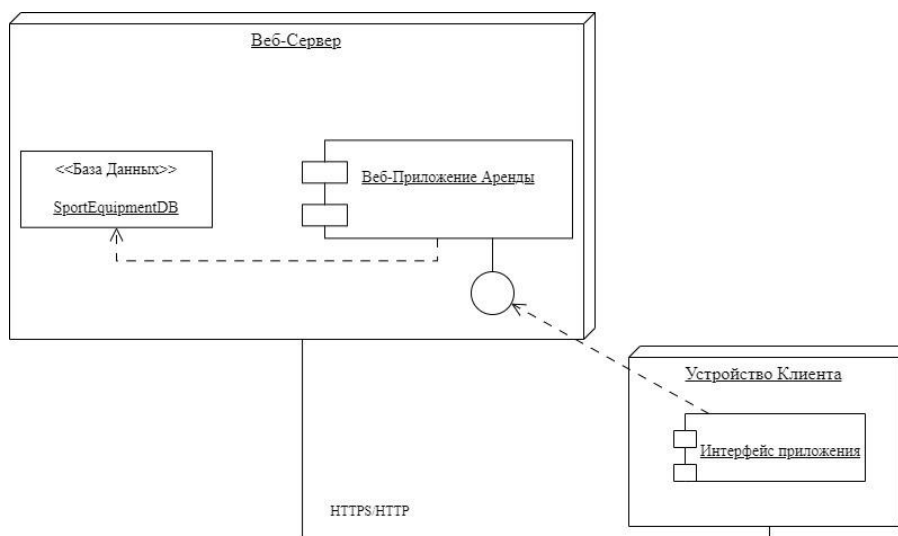


Рисунок 11 - Диаграмма развертывания

На рисунке 11 изображена топология разрабатываемой системы и маршруты передачи информации между ее компонентами.

#### 2.4.6 Диаграмма сотрудничества

Диаграмма сотрудничества — это вид диаграммы взаимодействия, в котором основное внимание сосредоточено на структуре взаимосвязей объектов, принимающих и отправляющих сообщения. [5]



Рисунок 12 - Диаграмма сотрудничества

На рисунке 12 изображена диаграмма, иллюстрирующая взаимодействия актеров системы с приложением, а также процессы отправки запросов и получения результатов.

#### 2.4.7 Диаграмма классов

Диаграмма классов предназначена для представления внутренней структуры программы в виде классов и связей между ними.

На рисунках 13-18 изображены диаграммы классов для различных категорий объектов. На диаграммах можно проследить закономерность, присущую процессу формирования классов: создание сущности, репозитория, сервиса, контроллера.

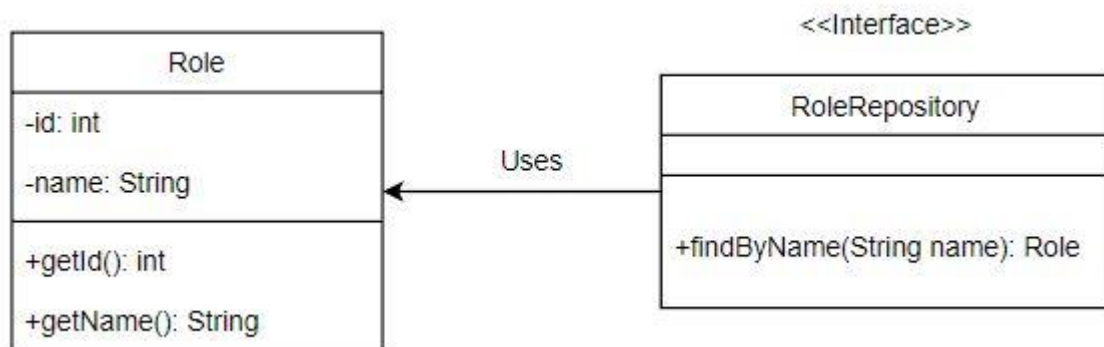


Рисунок 13 - Диаграмма классов для ролей

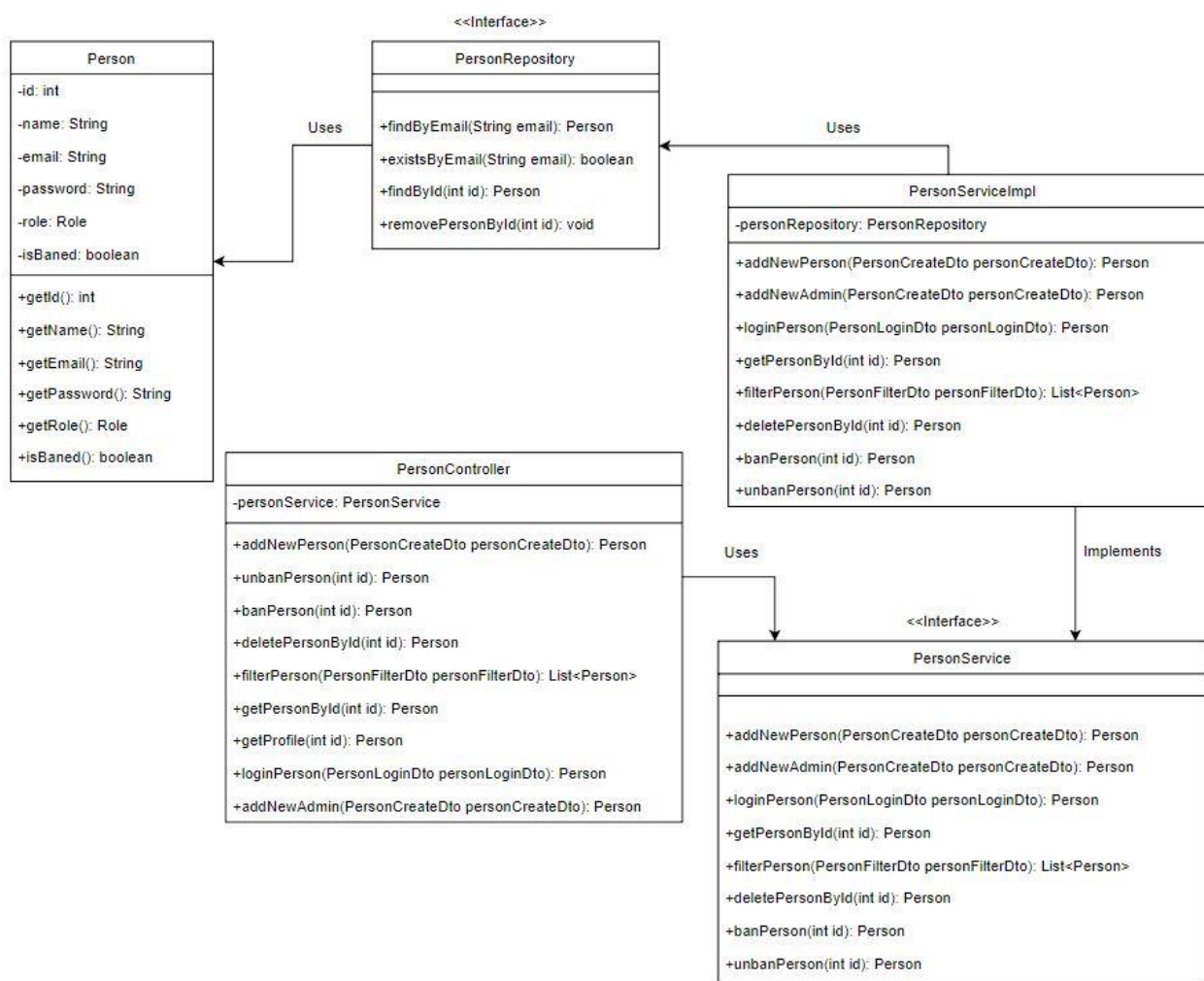


Рисунок 14 - Диаграмма классов для пользователей

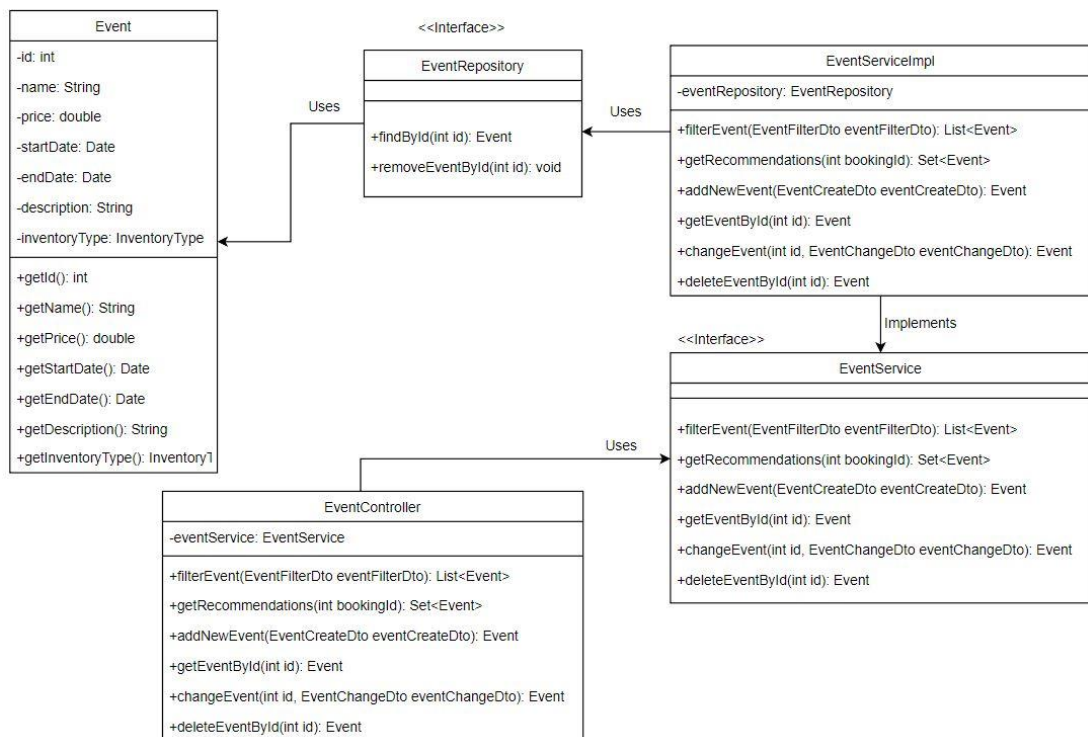


Рисунок 15 - Диаграмма классов для мероприятий

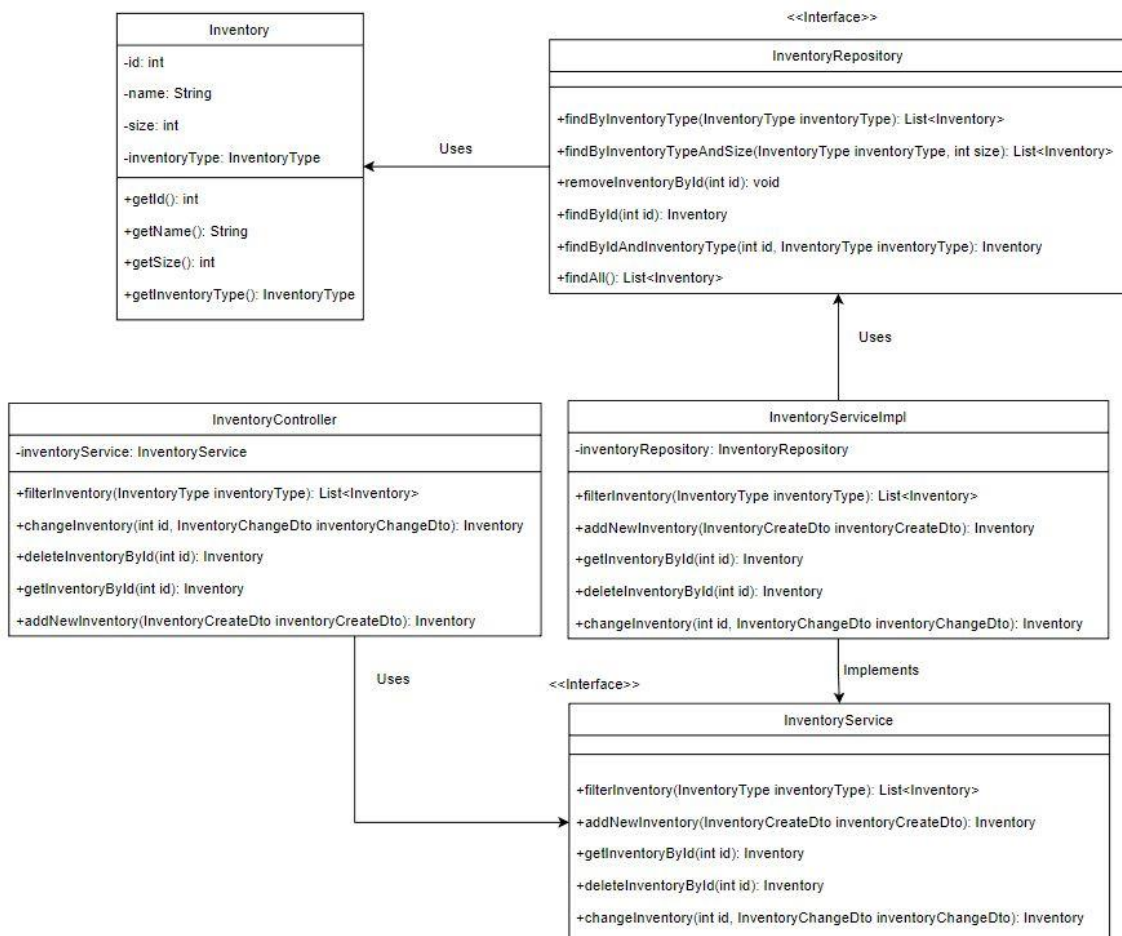


Рисунок 16 - Диаграмма классов для инвентаря



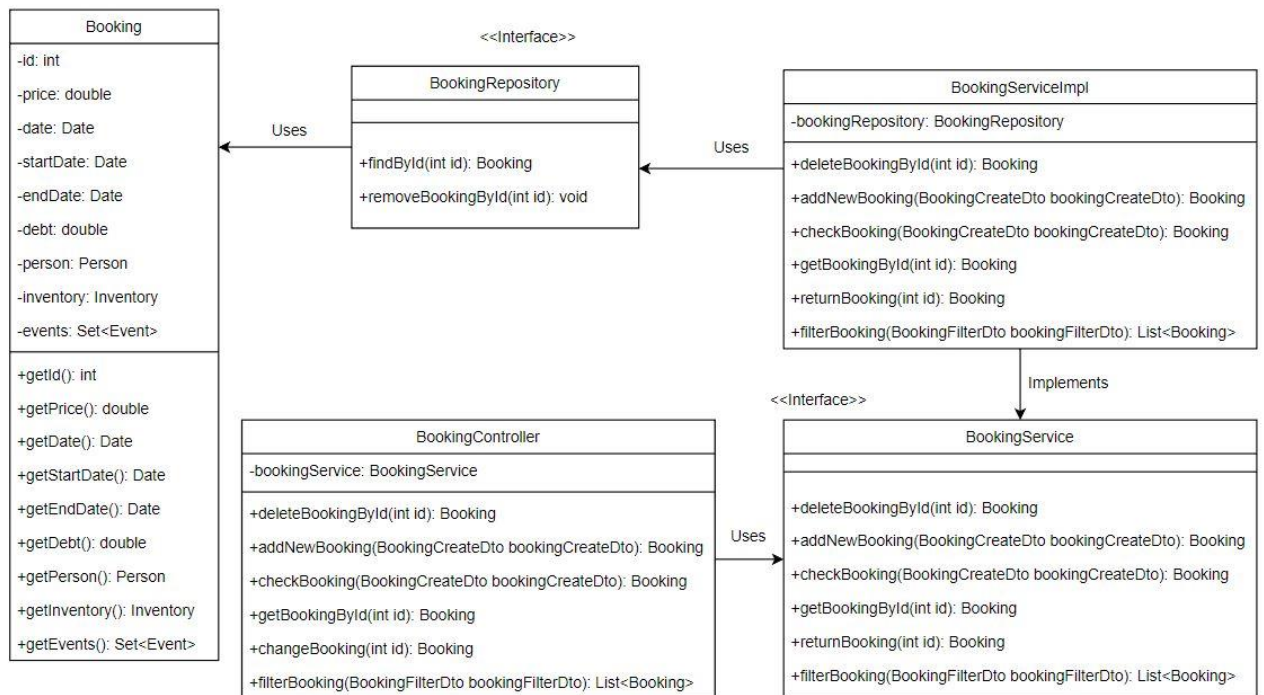


Рисунок 17 - Диаграмма классов для заказов

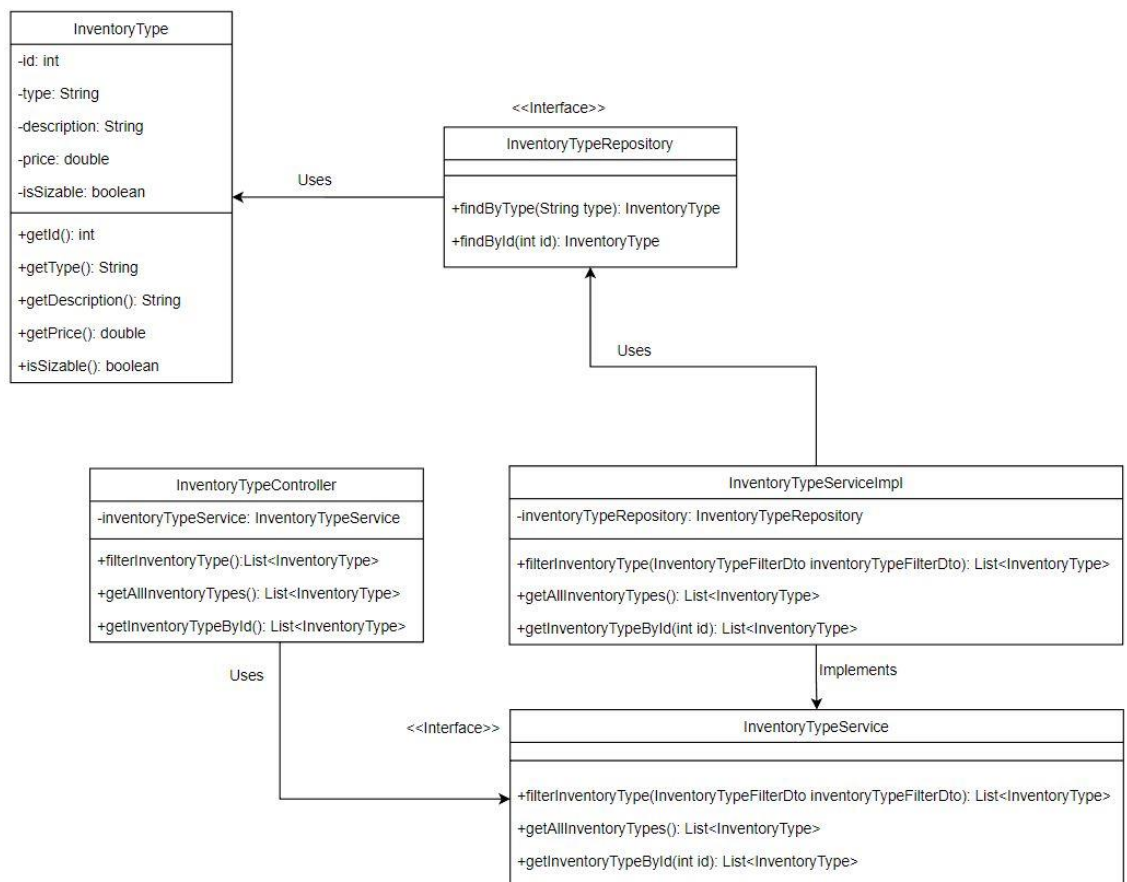


Рисунок 18 - Диаграмма классов для типов оборудования

### 2.4.8 Диаграмма объектов

Диаграмма объектов в языке моделирования UML предназначена для демонстрации совокупности моделируемых объектов и связей между ними в фиксированный момент времени.

На рисунке 19 изображена диаграмма объектов для разрабатываемой системы.

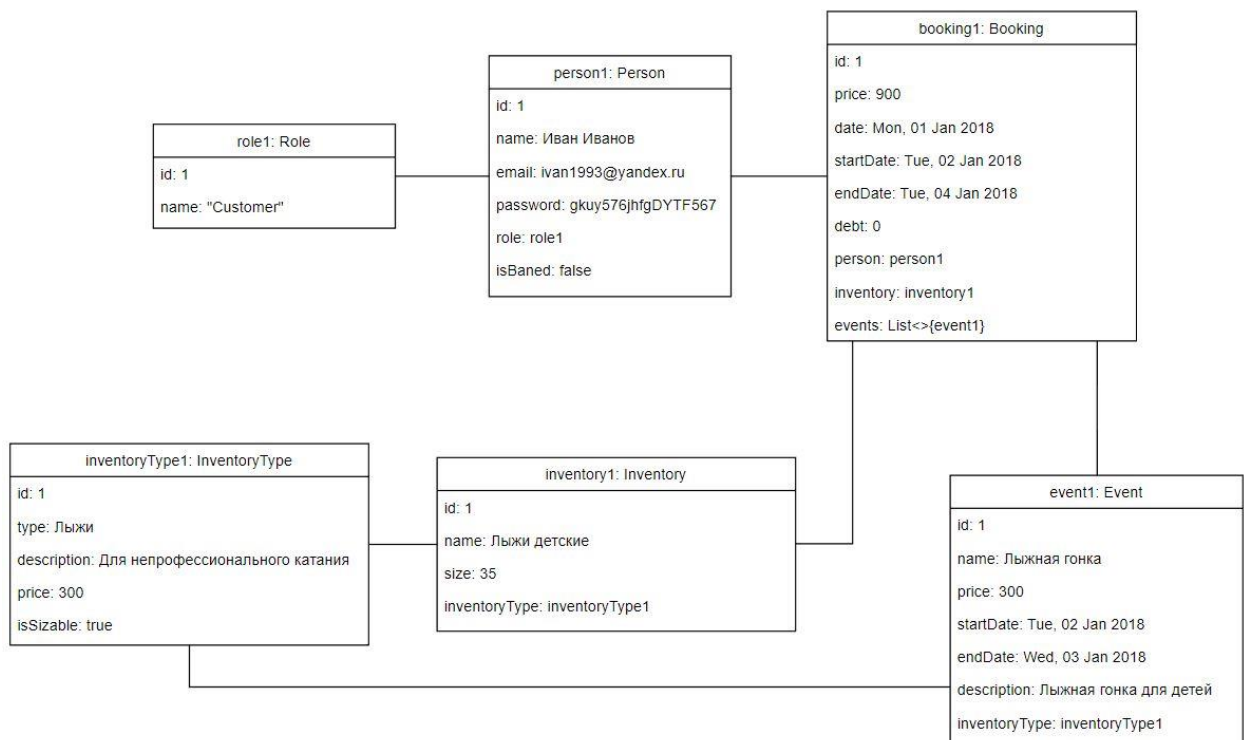


Рисунок 19 - Диаграмма объектов

### 2.4.9 Диаграмма состояний

На диаграмме состояний отображаются все переходы между состояниями изменяющегося объекта системы.

На рисунках 20-22 изображены диаграммы состояний для изменяющихся во времени объектов системы, а именно для заказа, клиента и инвентаря.



Рисунок 20 - Диаграмма состояний для заказа



Рисунок 21 - Диаграмма состояний для клиента



Рисунок 22 - Диаграмма состояний для инвентаря

#### 2.4.10 ER-диаграмма

Диаграмма «Сущность-связь» (ER-диаграмма) — визуальное представление базы данных, которое показывает, как связаны элементы

внутри. В данном случае она иллюстрирует, какие есть сущности и как они связаны внутри системы нашего веб-приложения.

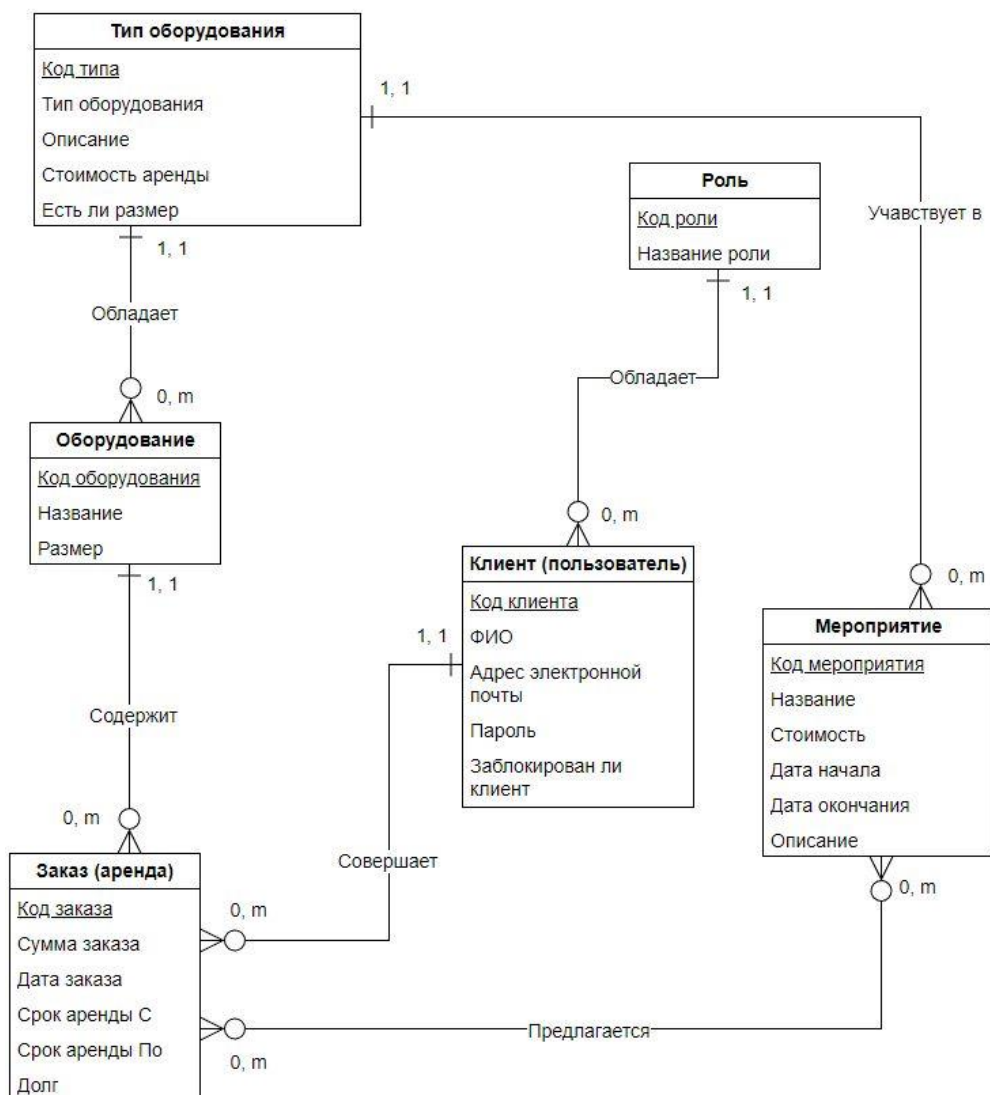


Рисунок 23 - ER-диаграмма

На рисунке 23 можно увидеть, какую информацию хранят в себе сущности, а также взаимодействия между ними.

#### 2.4.11 Физическая схема базы данных

Физическая схема базы данных — это модель данных, которая определяет, каким образом представляются данные, и содержит все детали, необходимые СУБД для создания базы данных.

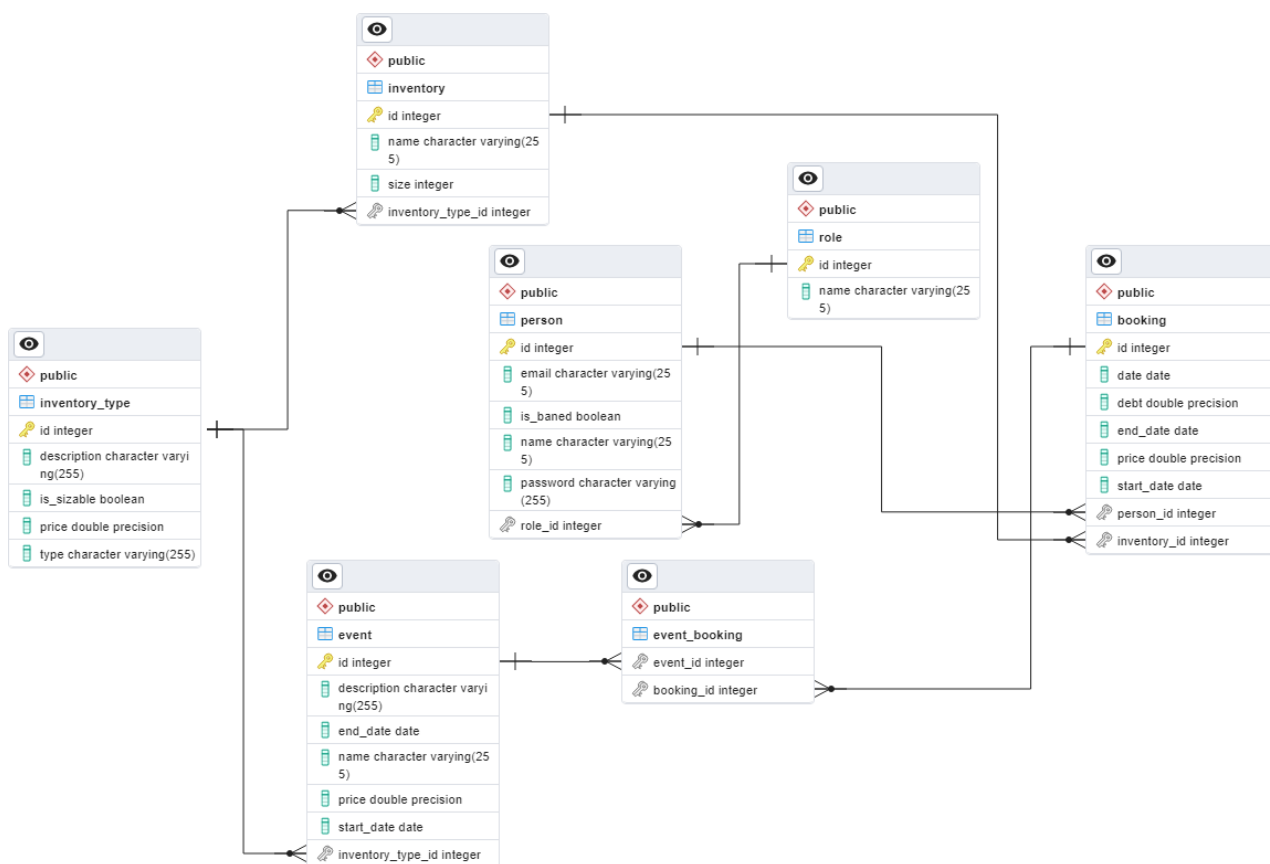


Рисунок 24 - Физическая схема базы данных

На рисунке 24 можно увидеть, как ER-диаграмма преобразовалась в физическую схему базы данных.

## 3 Реализация

### 3.1 Средства реализации

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

- Java — строго типизированный объектно-ориентированный язык программирования, характеризующийся платформенезависимостью. Он обладает следующими преимуществами: масштабируемость, безопасность, богатая стандартная библиотека, поддержка большого сообщества разработчиков.
- Spring Boot — фреймворк для разработки приложений на языке Java. Он используется для создания самостоятельных, готовых к работе приложений, которые легко масштабировать и поддерживать. Он обладает следующими преимуществами: ускорение разработки благодаря автоматической настройке и управлению зависимостями, встроенный контейнер сервлетов, интеграция с другими модулями и библиотеками Spring.
- PostgreSQL — объектно-реляционная система управления базами данных с открытым исходным кодом, основанная на языке SQL. [6] Она обладает следующими преимуществами: расширяемость, надежность, поддержка транзакций, масштабируемость, гибкость модели данных.
- Swagger — инструмент для документирования и тестирования API. [7] Он позволяет создавать интерактивную документацию для веб-сервисов, что упрощает их использование и интеграцию. Swagger автоматически генерирует документацию на основе аннотаций и комментариев в коде, что позволяет разработчикам сосредоточиться на написании логики приложения, а не на создании и поддержке документации. Благодаря Swagger, разработчики могут изучить доступные эндпоинты, параметры, модели данных и примеры запросов и ответов.

- Railway — хостинговый сервис, предназначенный для развертывания веб-приложений. Он обладает следующими преимуществами: автоматическая настройка среды выполнения, интеграция с популярными сервисами и базами данных, масштабирование и мониторинг приложений, возможность обновления кода, средства для работы в команде с другими разработчиками, безопасность (предоставляет возможности для настройки доступа, обработки аутентификации и авторизации).
  - bit.io — хостинг баз данных, предоставляющий средства для управления данными в облаке. Он обладает следующими преимуществами: масштабируемость, надежность (средства резервного копирования и восстановления данных), возможность доступа к данным из различных приложений и сервисов, средства для работы в команде с другими разработчиками, гибкие тарифные планы.
- Для разработки клиентской (frontend) части приложения был выбран следующий стек технологий:

- JavaScript — высокоуровневый интерпретируемый язык программирования, который применяется веб-разработке для создания интерактивных элементов на веб-страницах. Он обладает следующими преимуществами: широкая поддержка, гибкость, возможность создания динамических и интерактивных веб-страниц, доступ к различным библиотекам и фреймворкам.
- React.js — JavaScript-библиотека для создания пользовательских интерфейсов. [8] Она позволяет разработчикам создавать масштабируемые и переиспользуемые компоненты, которые обновляются только при изменении данных, что обеспечивает высокую производительность и эффективность. Она обладает следующими преимуществами: модульность, компонентная архитектура, виртуальный DOM для эффективного обновления интерфейса,

возможность создания одностраничных приложений (SPA), широкую поддержку сообщества разработчиков.

- CSS — язык стилей, используемый для оформления веб-страниц. Он определяет внешний вид и расположение элементов на странице, включая цвета, шрифты, размеры, отступы, границы и другие стилевые свойства. Он обладает следующими преимуществами: возможность изменения внешнего вида всего сайта путем изменения одного файла CSS, улучшение доступности и поддержки мобильных устройств, возможность переиспользования стилей благодаря созданию классов и переопределения стилей для различных элементов.
- HTML — язык разметки, используемый для создания структуры и содержимого веб-страниц. Он обладает следующими преимуществами: читаемость для разработчиков, возможность создания структурированного контента с использованием семантических элементов, доступность и адаптивности для различных устройств и браузеров. HTML является основным языком для создания веб-страниц и работает в сочетании с CSS для определения внешнего вида и JavaScript для добавления интерактивности.
- Vercel — облачная платформа для развертывания веб-приложений. Она обладает следующими преимуществами: масштабируемость, интеграция с CI/CD, автоматическое развертывание.

### **3.2 Реализация backend**

Серверная (backend) часть приложения была написана на языке Java с использованием фреймворка Spring Boot. Spring Boot подразумевает разделение приложения на модули (прослойки), которые будут описаны далее. Каждый модуль организован в виде отдельного пакета. Структура приложения и иерархия пакетов модулей изображена на рисунке 25.



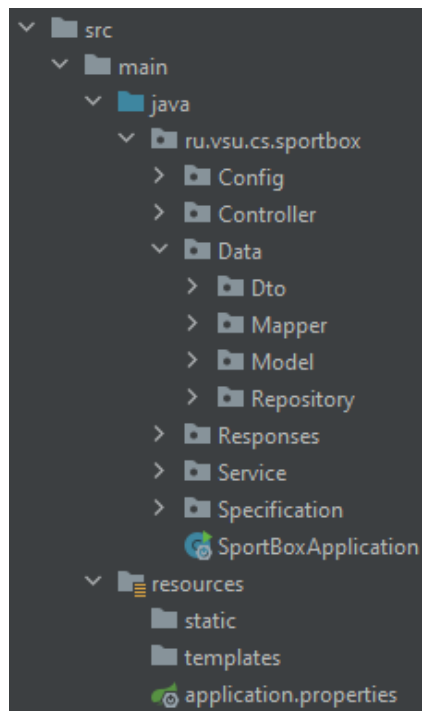


Рисунок 25 - Структура серверной части приложения

Приложение разделено на следующие модули:

- Модуль приложения (Application module). Это основной модуль, который содержит классы и настройки, связанные с запуском и конфигурацией приложения Spring Boot. В этом модуле находится главный класс приложения с аннотацией `@SpringBootApplication` и файл конфигурации `application.properties`, содержащий параметры подключения к базе данных. Для защиты базы данных эти параметры не объявляются явно в коде, а передаются в программу через переменные окружения.
- Модуль контроллеров (Controller Module). В этом модуле находятся классы контроллеров, отвечающие за обработку HTTP-запросов и взаимодействие с клиентом (браузером). Контроллеры содержат аннотации, такие как `@RestController` и `@RequestMapping`, для определения эндпоинтов и обработки запросов.
- Модуль конфигурации (Configuration Module). В этом модуле находятся классы конфигурации Spring, которые содержат настройки и бины, связанные с приложением. В случае разрабатываемого

приложения в модуле конфигурации только один класс `SecurityConfig`, в котором определены и настроены основные компоненты `Spring Security` для обеспечения безопасности серверной части приложения. Данный файл конфигурации настраивает основные компоненты безопасности `Spring Security`, а именно шифрование паролей (используется `BCryptPasswordEncoder`, который предоставляет безопасное хеширование паролей), правила авторизации (позволяют определить права доступа пользователя к различным ресурсам), фильтры безопасности и управление пользователями.

- Модуль моделей (`Model Module`). Этот модуль содержит классы моделей данных, которые отображают структуру таблиц в базе данных и представляют объекты бизнес-логики. Модели аннотированы аннотациями JPA, такими как `@Entity` и `@Table`, и содержат поля, геттеры и сеттеры. При работе с базой данных использовался фреймворк `Hibernate`. Он обеспечил автоматическое отображение классов и свойств на таблицы и столбцы в базе данных.
- Модуль репозитория (`Repository Module`). В этом модуле находятся классы репозитория, отвечающие за взаимодействие с базой данных. Репозитории используют `Spring Data JPA` и содержат аннотацию `@Repository`. Они предоставляют методы для выполнения операций CRUD (создание, чтение, обновление, удаление) и других запросов к базе данных.
- Модуль сервисов (`Service Module`). Этот модуль содержит бизнес-логику приложения, которая обрабатывает запросы, полученные от контроллеров. Здесь находятся классы сервисов с аннотацией `@Service`, которые выполняют определенные операции и взаимодействуют с репозиториями.

Кроме этих основных модулей в приложении используются такие паттерны, как:

- DTO (Data Transfer Object) — компонент в архитектуре приложения, который используется для обмена данными между клиентской и серверной частями приложения. DTO представляют основные сущности или объекты приложения. Целью DTO является упрощение передачи данных и уменьшение количества запросов, необходимых для обмена информацией. DTO пересылаются между клиентской и серверной частями приложения в формате JSON.
- Response — классы, которые описывают форматы ответов, возвращаемых backend-частью приложения в ответ на запросы от клиентской части приложения (frontend).
- Спецификации (Specifications) представляют собой механизм для построения сложных запросов к базе данных. Они используются для динамического создания условий и критериев поиска данных, основываясь на различных параметрах или предикатах. Основная цель использования спецификаций состоит в том, чтобы создать гибкий и масштабируемый способ формирования запросов без необходимости явного написания SQL-запросов. Они позволяют строить запросы в коде, используя объектно-ориентированный подход.

База данных была развернута на хостинге bit.io. После создания базы данных были предоставлены данные для подключения, такие как хост, порт, имя базы данных, имя пользователя и пароль. Эти данные будут использоваться для настройки подключения к базе данных из приложения.

Сервер был развернут (загружен) на хостинге Railway. Этот процесс был выполнен с использования системы контроля версий Git. Railway будет автоматически обнаруживать изменения и запускать процесс развертывания сервера. После загрузки потребовалось настроить переменные окружения, представляющие собой конфиденциальные настройки, такие как данные аутентификации для базы данных.

Для описания спецификации API использовался Swagger. Для интеграции его в проект потребовалось добавить соответствующую

зависимость (springdoc-openapi) и аннотации к контроллерам, методам, моделям. На основе этих аннотаций и структуры приложения составляется интерактивная документация API.

### **3.3 Реализация frontend**

Клиентская часть приложения (frontend) разработана на языке JavaScript с использованием Фреймворка React.js. Для взаимодействия с серверной частью приложения на клиентской части используется библиотека axios, которая предоставляет интерфейс для выполнения HTTP-запросов из браузера.

Разрабатываемое приложение на React представляет собой SPA (single page application), то есть одностраничное приложение. Приложение загружает одну HTML страницу и динамически обновляет её без необходимости перезагрузки страницы при взаимодействии пользователя с приложением. Вместо традиционного подхода, когда каждый переход на новую страницу вызывает полную загрузку HTML, CSS и JavaScript, SPA загружает и инициализирует приложение один раз, а затем динамически обновляет только необходимую часть страницы при переходе между разделами или выполнении определенных действий.

Особенностью разработки приложения на React является разбиение на компоненты. Все приложение разбито на компоненты, которые в необходимый момент загружаются на страницу. Компоненты представляют собой функции, возвращающие определённый HTML код. Эти функции могут хранить в себе переменные или другие функции.

Разработанное React-приложение имеет базовую структуру: директорию public, в которой хранятся статические ресурсы (HTML файлы), и директорию src (source), в которой хранятся все исходные файлы приложения. Директория src хранит в себе поддиректорию components (для хранения всех компонентов приложения), css (для хранения всех CSS файлов, описывающих стили), images (в ней хранятся необходимые приложению изображения) и store (в этой

директории находится хранилище состояния приложения). Кроме того, в директории src находятся основные .js файлы: index.js и App.js.

Описанная структура приложения изображена на рисунке 26.

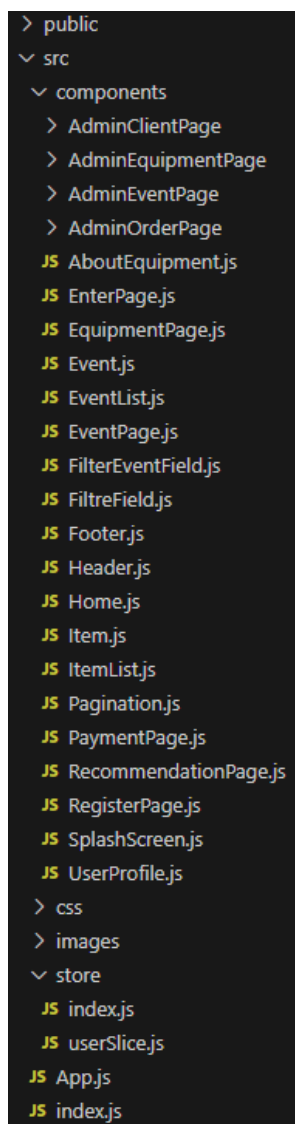


Рисунок 26 - Структура React-приложения

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

Компоненты могут состоять из других компонентов. Поэтому в приложении помимо компонентов, описывающих конкретные страницы, есть компоненты, из которых состоят эти страницы.

Так, среди файлов для компонентов, обозначающих страницы, можно выделить:

- UserProfile.js;
- SplashScreen.js;
- RegisterPage.js;
- RecommendationPage.js;
- PaymentPage.js;
- Home.js;
- EventPage.js;
- EquipmentPage.js;
- EnterPage.js;
- AboutEquipment.js;
- AdminOrderManagerPage.js;
- AdminOrderChangePage.js;
- AdminEventManagerPage.js;
- AdminChangeEventPage.js;
- AdminAddEventPage.js;
- AdminEquipmentManagerPage.js;
- AdminChangeEquipment.js;
- AdminAddEquipmentPage.js;
- AdminClientManagerPage.js;
- AdminAddClient.js.

Остальные компоненты являются частями страниц приложения.

Приложение имеет некоторые глобальные переменные, которые могут понадобиться в различных частях приложения, например, данные об авторизованном пользователе необходимы для отображения его профиля, при

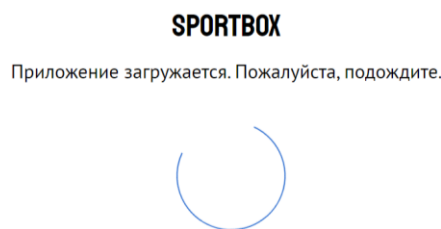
создании заказа и т.д. Для хранения этих переменных и для более удобного их использования в приложении создано хранилище (store), в котором хранится всё состояние приложения. Оно было сделано при помощи Redux. Redux — это библиотека управления состоянием приложения в JavaScript. Она предоставляет предсказуемый и однонаправленный поток данных, который упрощает управление состоянием приложения и обеспечивает единообразие доступа к данным для всех компонентов.

Хранилище состоит из 2 файлов: `index.js` и `userSlice.js`. `index.js` является конфигурацией хранилища, а `userSlice.js` представляет само хранилище.

### **3.4 Навигация по приложению**

#### **3.4.1 Для клиента**

При запуске приложения пользователя встречает Splash screen.



Первое, что видит пользователь после завершения загрузки, — главный экран. У пользователя есть выбор: войти в существующий аккаунт, пройти регистрацию или продолжить пользоваться приложением без возможности сделать заказ.





SPORTBOX

ГлавнаяОборудованиеМероприятияВход

Для регистрации введите данные

ФИО:

Иванов Иван Иванович

Адрес эл. почты:

example@example.ru

Пароль:

\*\*\*\*\*

Зарегистрироваться

[Войти, если есть аккаунт](#)

КОНТАКТНАЯ ИНФОРМАЦИЯ:

394018, Россия, г. Воронеж,  
Университетская площадь, 1  
+7 (493) 900-56-56  
blatys5377@gmail.com  
ignat\_kandaurov@mail.ru  
kornilovilja@rambler.ru

РАЗРАБОТЧИКИ:

И.Р. Корнилов,  
Е.А. Бабкина,  
И.А. Кандауров

Рисунок 29 - Экран регистрации нового аккаунта

На страницах для просмотра оборудования и мероприятий можно осуществить фильтрацию списков оборудования и мероприятий по предлагаемым параметрам.

SPORTBOX

ГлавнаяОборудованиеМероприятияВход

Фильтры поиска

Тип оборудования:

Выберите тип

Дата начала аренды:

ДД.ММ.ГГГГ

Дата окончания аренды:


ДД.ММ.ГГГГ

Цена за день:

от:


до:

Найти




Мяч баскетбольный  
70 руб./день

Арендовать




Лыжи беговые  
200 руб./день

Арендовать



Тюбинг  
280 руб./день

Арендовать



Сноуборд  
900 руб./день

Арендовать

1 2 3

Рисунок 30 - Экран для просмотра доступного оборудования

41

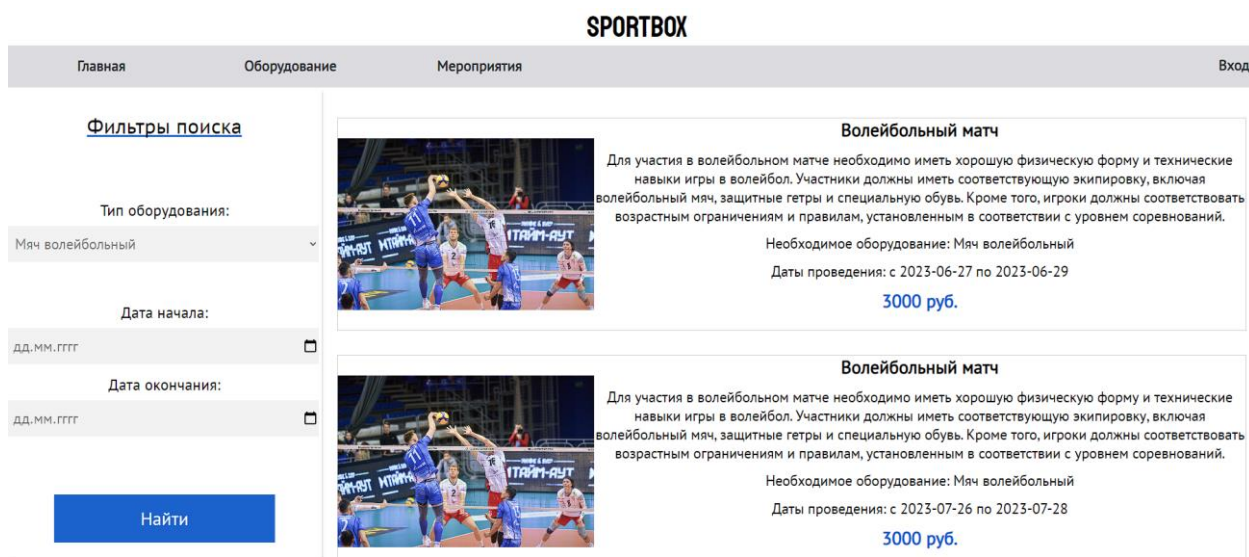


Рисунок 31 - Экран для просмотра доступных мероприятий

После авторизации пользователю становится доступной возможность арендовать конкретное оборудование. Для этого на странице для просмотра ассортимента оборудования необходимо нажать на кнопку «Арендовать». При аренде пользователю предлагается ввести даты аренды и размер, если он есть у инвентаря.

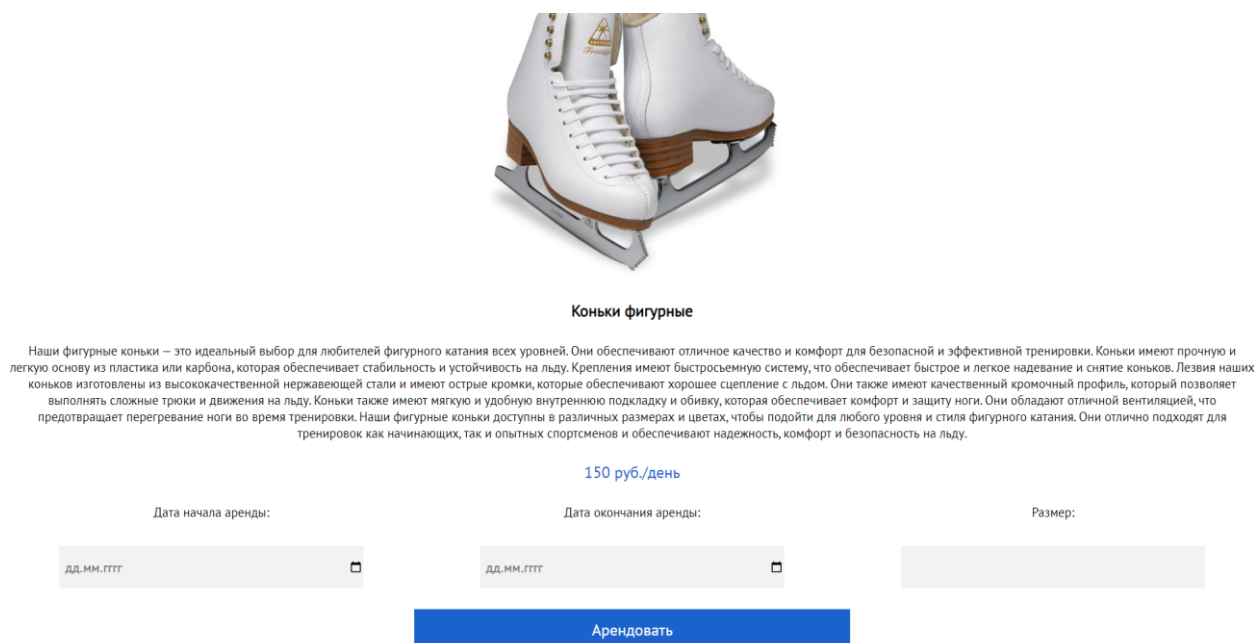


Рисунок 32 - Экран аренды оборудования

После нажатия на кнопку «Арендовать» происходит переход на экран оплаты, где необходимо ввести данные карты. На этом экране также

выводится информация о заказе, чтобы пользователь мог проверить ее перед оплатой.

Коньки фигурные Nordway Emily  
С 2023-05-25 по 2023-05-26  
40 размер  
150 руб.

Для оплаты введите данные карты или отсканируйте QR-код на экране.


maestro

VISA

Номер карты

ММ / ГГ Три цифры на обороте карты CVC

Оплатить



Оплатить по QR-коду

Рисунок 33 - Страница оплаты

После завершения оплаты создается заказ и предлагаются мероприятия к посещению.

SPORTBOX

Главная

Оборудование

Мероприятия

Профиль

Заказ № 20 успешно оплачен.

Рекомендуем принять участие в следующих мероприятиях:



Соревнования по фигурному катанию

Для участия необходимо иметь высокую физическую подготовку и технические навыки на льду. Участники должны иметь определенный уровень мастерства, который определяется в соответствии с возрастом и опытом. Необходима подходящая экипировка, включая коньки, костюмы и протекторы.

Необходимое оборудование: Коньки фигурные

Даты проведения: с 2023-06-10 по 2023-06-20

4000 руб.

Рисунок 34 - Экран с рекомендациями после совершения заказа

Историю своих заказов, как и информацию о себе, введенную при регистрации, пользователь может посмотреть в своем профиле.

#### Информация о пользователе:

ФИО: Бабкина Екатерина  
Александровна

Адрес эл. почты: bkatya5577@gmail.com

#### История заказов:

№	Оборудование	Цена, руб.	Дата заказа	Дата начала	Дата окончания	Долг, руб.	
18	Коньки фигурные Nordway Emily	1200	2023-05-19	2023-06-10	2023-06-18	0	<a href="#">Отменить</a>
15	Велосипед взрослый Stern Angel	800	2023-05-19	2023-06-01	2023-06-03	0	<a href="#">Отменить</a>
20	Коньки фигурные Nordway Emily	150	2023-05-21	2023-06-19	2023-06-20	0	<a href="#">Отменить</a>
8	Коньки фигурные Nordway Emily	150	2023-05-18	2023-06-01	2023-06-02	0	<a href="#">Отменить</a>

Рисунок 35 - Профиль клиента

### 3.4.2 Для администратора

Администратору предлагаются возможности для работы со списком оборудования, мероприятий, клиентов, заказов. В качестве примера рассмотрим работу с оборудованием. На странице оборудования выводится список доступного оборудования, который можно отфильтровать по типу или идентификатору оборудования.

**SPORTBOX**

[Главная](#) [Оборудование](#) [Мероприятия](#) [Клиенты](#) [Заказы](#) [Профиль](#)

[Фильтры поиска](#)

Тип оборудования:  
Выберите тип

Идентификатор оборудования:

[Найти](#)

[Добавить оборудование](#)

Id	Название	Тип	Размер	Цена в день, руб.		
1	Велосипед взрослый Stern Angel	Велосипед взрослый	-	400	<a href="#">Удалить</a>	<a href="#">Изменить</a>
2	Велосипед взрослый Stern Angel	Велосипед взрослый	-	400	<a href="#">Удалить</a>	<a href="#">Изменить</a>
3	Велосипед взрослый Stern Angel	Велосипед взрослый	-	400	<a href="#">Удалить</a>	<a href="#">Изменить</a>

Рисунок 36 - Страница со списком оборудования

При нажатии на кнопку «Добавить оборудование» предлагается ввести данные для нового оборудования.

**SPORTBOX**

Главная	Оборудование	Мероприятия	Клиенты	Заказы	Профиль
---------	--------------	-------------	---------	--------	---------

Для добавления оборудования введите данные

Тип оборудования:

Название:

Размер:

КОНТАКТНАЯ ИНФОРМАЦИЯ: 394018, Россия, г. Воронеж, Университетская площадь, 1 +7 (495) 900-56-56 bkatya5577@gmail.com ignat_kandaurov@mail.ru	РАЗРАБОТЧИКИ: И.Р. Корнилов, Е.А. Бабкина, И.А. Кандауров
--	--

Рисунок 37 - Страница с добавлением нового оборудования

В таблице в строке с каждым оборудованием выводится кнопка «Удалить» (при нажатии выбранное оборудование удаляется из базы данных), а также кнопка «Изменить» (при нажатии происходит переход на страницу изменения выбранного оборудования). На странице изменения оборудования предлагается ввести новые данные об оборудовании.

**SPORTBOX**

Главная	Оборудование	Мероприятия	Клиенты	Заказы	Профиль
---------	--------------	-------------	---------	--------	---------

Для изменения оборудования введите новые данные

Тип оборудования:

Название:

Рисунок 38 - Страница для изменения существующего оборудования

В профиле администратора выводится информация о нем: его ФИО и адрес электронной почты.

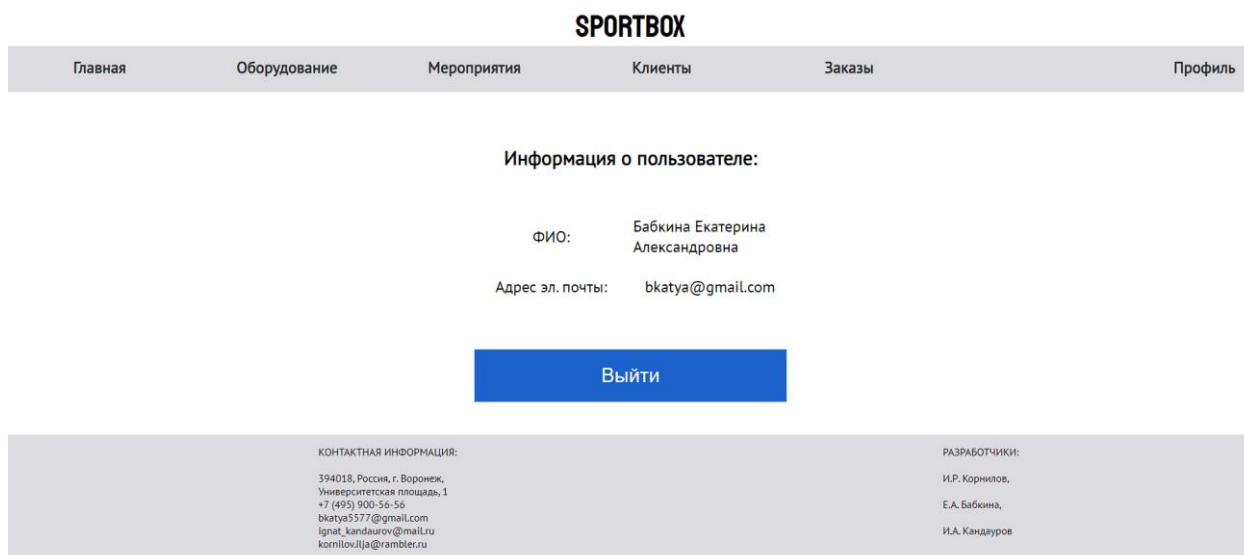


Рисунок 39 - Профиль администратора

### 3.5 Тестирование

Для более полного тестирования разработанной системы были проведены тесты основных типов:

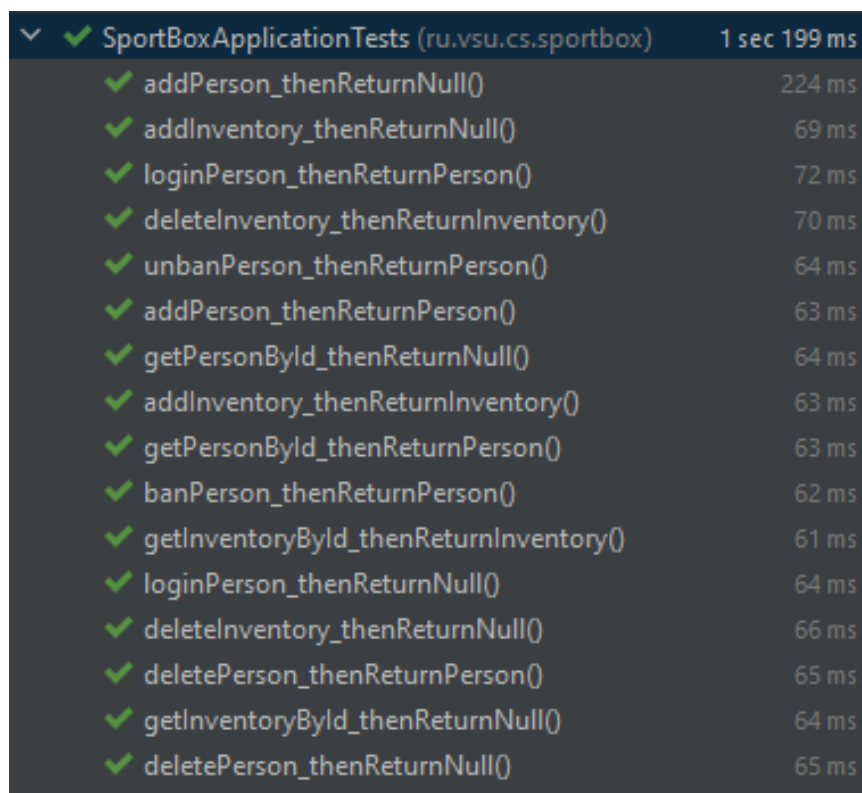
- Модульное тестирование (unit-тесты)
- Тесты, связанные с изменениями (дымовое тестирование);
- Функциональные тесты (тестирование пользовательского интерфейса);
- Нефункциональные тесты (usability тестирование).

#### 3.5.1 Модульное тестирование

Unit-тесты являются частью процесса разработки программного обеспечения и используются для проверки отдельных модулей или компонентов программы. Они предназначены для тестирования функциональности отдельных единиц кода, таких как отдельные классы, методы или функции.

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

На рисунке 40 представлены результаты unit-тестов серверной части приложения. Проверялась работа контроллеров. Тесты были написаны в классе с аннотацией @WebMvcTest.



✓ SportBoxApplicationTests (ru.vsu.cs.sportbox)	1 sec 199 ms
✓ addPerson_thenReturnNull()	224 ms
✓ addInventory_thenReturnNull()	69 ms
✓ loginPerson_thenReturnPerson()	72 ms
✓ deleteInventory_thenReturnInventory()	70 ms
✓ unbanPerson_thenReturnPerson()	64 ms
✓ addPerson_thenReturnPerson()	63 ms
✓ getPersonById_thenReturnNull()	64 ms
✓ addInventory_thenReturnInventory()	63 ms
✓ getPersonById_thenReturnPerson()	63 ms
✓ banPerson_thenReturnPerson()	62 ms
✓ getInventoryById_thenReturnInventory()	61 ms
✓ loginPerson_thenReturnNull()	64 ms
✓ deleteInventory_thenReturnNull()	66 ms
✓ deletePerson_thenReturnPerson()	65 ms
✓ getInventoryById_thenReturnNull()	64 ms
✓ deletePerson_thenReturnNull()	65 ms

Рисунок 40 - Результаты unit-тестов

### 3.5.2 Дымовое тестирование

Дымовое тестирование (smoke testing) — это вид тестирования программного обеспечения, который выполняется для быстрой проверки основной функциональности системы или компонента после внесения изменений или внедрения новой версии. Основная цель дымового тестирования — обнаружить критические ошибки или проблемы, которые могут повлиять на работоспособность системы в целом.

В ходе дымового тестирования выполняются базовые операции или сценарии, которые предполагаются как наиболее важные и часто используемые пользователем.

В таблицах 1, 2 представлены результаты дымового тестирования для основных сценариев клиента и администратора.

Таблица 1 - Результаты дымового тестирования для клиента

Тестовый сценарий	Результат теста
Регистрация	Пройден
Авторизация	Пройден
Просмотр ассортимента оборудования	Пройден
Просмотр мероприятий	Пройден
Аренда оборудования	Пройден
Оплата заказа	Пройден
Отмена заказа	Пройден
Получение информации о пользователе в профиле	Пройден

Таблица 2 - Результаты дымового тестирования для администратора

Тестовый сценарий	Результат теста
Авторизация	Пройден
Просмотр списка оборудования	Пройден
Добавление оборудования	Пройден
Удаление оборудования	Пройден
Изменение оборудования	Пройден
Просмотр списка мероприятий	Пройден
Добавление мероприятия	Пройден
Удаление мероприятия	Пройден
Изменение мероприятия	Пройден
Просмотр списка пользователей	Пройден
Добавление пользователя	Пройден
Блокировка пользователя	Пройден
Разблокировка пользователя	Пройден
Удаление пользователя	Пройден
Просмотр списка заказов	Пройден



Изменение (принятие) заказа	Пройден
Получение информации о пользователе в профиле	Пройден

### 3.5.3 Тестирование пользовательского интерфейса

Тестирование пользовательского интерфейса (UI-тестирование) — это процесс тестирования элементов управления в приложении, который помогает убедиться, что интерфейс соответствует ожидаемой функциональности. Задача проведения UI-тестов — убедиться, что в функциях пользовательского интерфейса отсутствуют дефекты.

В таблице 3 представлена часть результатов тестирования пользовательского интерфейса. В ней отражены тестовые сценарии для неавторизованного пользователя.

Таблица 3 - Результаты UI-тестирования для неавторизованного пользователя

Тестовый сценарий	Ожидаемый результат	Статус теста
Нажатие на кнопку «Оборудование» в меню	Переход на страницу оборудования	Пройден
Нажатие на кнопку «Мероприятия» в меню	Переход на страницу мероприятий	Пройден
Нажатие на кнопку «Главная» в меню	Переход на главную страницу	Пройден
Нажатие на логотип	Переход на главную страницу	Пройден
Нажатие на кнопку «Вход» в меню	Переход на страницу входа	Пройден
Нажатие на ссылку «Зарегистрироваться, если нет аккаунта» на странице входа	Переход на страницу регистрации	Пройден

Нажатие на ссылку «Войти, если есть аккаунт» на странице регистрации	Переход на страницу входа	Пройден
Нажатие на кнопку «Арендовать» на странице оборудования	Переход на страницу входа	Пройден
Нажатие на кнопку «Найти» на странице оборудования с пустыми полями для ввода	Вывод всех типов оборудования, хранящихся в базе данных	Пройден
Нажатие на кнопку «Найти» на странице оборудования с выбранным типом оборудования «Велосипед взрослый»	Вывод карточки конкретного типа оборудования — взрослого велосипеда	Пройден
Нажатие на кнопку «Найти» на странице оборудования с заполненными полями для цены: от 100 до 150	Вывод карточек оборудования, соответствующего указанному диапазону цены	Пройден
Попытка входа с незаполненным полем для адреса электронной почты	Вывод предупреждения «Заполните это поле»	Пройден
Попытка входа с незаполненным полем для пароля	Вывод предупреждения «Заполните это поле»	Пройден
Попытка регистрации с незаполненным полем для адреса электронной почты	Вывод предупреждения «Заполните это поле»	Пройден

Попытка регистрации с незаполненным полем для пароля	Вывод предупреждения «Заполните это поле»	Пройден
Попытка регистрации с незаполненным полем для ФИО	Вывод предупреждения «Заполните это поле»	Пройден

### 3.5.4 Usability тестирование

Тестирование удобства пользования (Usability testing) — это процесс оценки, который выполняется для определения того, насколько легко и удобно пользователи могут взаимодействовать с продуктом. Целью тестирования удобства пользования является выявление проблем, с которыми сталкиваются пользователи при использовании продукта, а также сбор обратной связи и предложений для улучшения пользовательского опыта.

В таблице 4 представлены результаты тестирования удобства пользования для 4 респондентов, перед которыми были поставлены указанные задания. Оценка успешности выполнения задания производилась с помощью подхода Нильсена, где выделяют три значения показателя:

- Если респондент справился с заданием почти без проблем — 100% (оценка 1);
- Если у респондента возникли некоторые проблемы, но он сделал задание — 50% (оценка 0,5);
- Если респондент не выполнил задание — 0% (оценка 0).

Таблица 4 - Результаты тестирования удобства пользования

Задание	Респондент 1	Респондент 2	Респондент 3	Респондент 4
Регистрация	1	1	1	1
Авторизация	1	1	1	1

Просмотр ассортимента оборудования	1	1	1	1
Просмотр мероприятий	1	1	1	1
Аренда оборудования	1	0,5	1	1
Оплата заказа	1	1	1	0,5
Отмена заказа	1	1	1	1
Получение информации о пользователе в профиле	1	1	1	1

Расчет средней успешности по заданиям:

$$\frac{30 * 1 + 2 * 0,5}{32} * 100\% = \frac{31}{32} * 100\% \approx 97\%$$

## Заключение

В ходе данной курсовой работы были выполнены все поставленные задачи. Было разработано веб-приложение для аренды спортивного инвентаря. Благодаря нему у клиентов организации появилась возможность онлайн:

- Ознакомиться с ассортиментом;
- Сделать заказ;
- Оплатить заказ;
- Посмотреть историю своих заказов;
- Получить рекомендации по посещению мероприятий (рекомендации составляются на основе данных о времени аренды и арендованном оборудовании);
- Отменить заказ при необходимости.

Кроме того, благодаря приложению у сотрудников организации появилась возможность онлайн:

- Просматривать, изменять, добавлять, удалять данные об оборудовании;
- Просматривать, изменять, добавлять, удалять данные о мероприятиях;
- Просматривать, изменять данные о заказах;
- Просматривать, добавлять, удалять данные о клиентах;
- Добавлять клиентов в черный список/выносить их из черного списка при необходимости.

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

## Список использованных источников

1. Дизайн как элемент конкурентоспособности спортивного инвентаря [Электронный ресурс]. — Режим доступа: [https://studbooks.net/888936/marketing/ponyatie\\_sportivnyy\\_inventar](https://studbooks.net/888936/marketing/ponyatie_sportivnyy_inventar). — Заглавие с экрана. — (Дата обращения: 15.03.2023).
2. Что такое MVC: рассказываем простыми словами [Электронный ресурс]. — Режим доступа: <https://ru.hexlet.io/blog/posts/что-такое-mvc-rasskazyvaem-prostymi-slovami>. — Заглавие с экрана. — (Дата обращения: 16.03.2023).
3. Backend разработка [Электронный ресурс]. — Режим доступа: <https://mobile-erp.ru/backend-razrabotka/>. — Заглавие с экрана. — (Дата обращения: 16.03.2023).
4. Фронтенд [Электронный ресурс]. — Режим доступа: <https://www.gorkilib.ru/events/frontend>. — Заглавие с экрана. — (Дата обращения: 17.03.2023).
5. Диаграммы сотрудничества [Электронный ресурс]. — Режим доступа: <https://helpiks.org/9-53448.html>. — Заглавие с экрана. — (Дата обращения: 16.03.2023).
6. Система управления объектно-реляционными базами данных PostgreSQL [Электронный ресурс]. — Режим доступа: <https://webcreator.ru/technologies/webdev/postgresql>. — Заглавие с экрана. — (Дата обращения: 20.04.2023).
7. Тестирование API с помощью Swagger: особенности и преимущества [Электронный ресурс]. — Режим доступа: <https://blog.ithillel.ua/ru/articles/api-testing-with-swagger>. — Заглавие с экрана. — (Дата обращения: 20.04.2023).
8. Что такое React и как его освоить? [Электронный ресурс]. — Режим доступа: <https://academy.yandex.ru/journal/что-такое-react-i-kak-ego-osvoit>. — Заглавие с экрана. — (Дата обращения: 20.04.2023).