Пермский филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский университет «Высшая школа экономики»

*Факультет социально-экономических и компьютерных наук*

Лазаренко Александр Сергеевич

**РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ УЧЁТА ПРОКАТА АВТОМОБИЛЕЙ**

*Курсовая работа*

студента образовательной программы «Разработка информационных систем для бизнеса» по направлению подготовки 09.03.04 Программная инженерия

Руководитель проектной работы

к.ф.-м.н., доцент, доцент кафедры информационных технологий в бизнесе

\_\_\_\_\_\_\_\_\_\_\_\_Л. Н. Лядова

Пермь, 2025

АННОТАЦИЯ

Лазаренко А. С. Тема: Курсовая работа на тему «Разработка информационной системы для учёта проката автомобилей» по направлению подготовки «Программная инженерия» / Лазаренко Александр Сергеевич. – Пермь: НИУ ВШЭ – Пермь, 2025.

Работа посвящена разработке информационной системы, предназначенной для учёта проката автомобилей.

Во "Введении" обосновывается актуальность работы, ставятся цель проекта и задачи.

В первой главе приведены результаты анализа объекта автоматизации, показаны проблемы учёта проката автомобилей и выявлены их причины. Представлен анализ существующих решений, их оценка. Глава заканчивается анализом требований к информационной системе.

Во второй главе представлены результаты проектирования информационной системы. Выполнено проектирование базы данных и графического интерфейса. Задачи проектирования решены на основе объектно-ориентированного подхода, при построении моделей использован язык UML. Результатами главы являются физическая модель базы данных и интерфейс приложения.

Третья глава содержит описание реализации и тестирования информационной системы. Результатом главы является отлаженное стабильно работающие веб-приложение.

В "Заключении" содержится описание результатов, полученных в ходе выполнения курсовой работы.

Работа содержит 68 страниц основного текста, 3 таблицы, 45 рисунков. Библиографический список включает 6 наименований. Работа включает 3 приложения.

**Оглавление**

[ВВЕДЕНИЕ 5](#_Toc198587082)

[Глава 1. Анализ объекта автоматизации 7](#_Toc198587083)

[1.1 Анализ предметной области 7](#_Toc198587084)

[1.1.1 Анализ процесса бронирования 8](#_Toc198587085)

[1.1.2 Анализ автомобилей 8](#_Toc198587086)

[1.2 Анализ существующих решений 9](#_Toc198587087)

[1.2.1 Выделение критериев оценки 10](#_Toc198587088)

[1.2.2 Анализ существующих решений 10](#_Toc198587089)

[1.3 Анализ требований 15](#_Toc198587090)

[1.3.1 Анализ процесса учёта проката автомобилей 16](#_Toc198587091)

[1.3.2 Анализ процесса учёта автомобилей 18](#_Toc198587092)

[1.3.3 Анализ процесса учёта аренд 20](#_Toc198587093)

[1.3.4 Анализ процесса учёта технического состояния автомобиля 22](#_Toc198587094)

[1.3.5 Анализ процесса бронирование автомобиля 23](#_Toc198587095)

[1.3.6 Анализ процесса аутентификации и авторизации 24](#_Toc198587096)

[1.4 Результаты анализа 25](#_Toc198587097)

[Глава 2. Проектирование информационной системы 26](#_Toc198587098)

[2.1 Проектирование базы данных 26](#_Toc198587099)

[2.1.1 Описание операций 26](#_Toc198587100)

[2.1.2 Описание атрибутов для проектирования БД 29](#_Toc198587101)

[2.1.3 Функциональные зависимости 35](#_Toc198587102)

[2.1.4 Универсальное отношение 36](#_Toc198587103)

[2.1.5 Приведение к 2НФ 38](#_Toc198587104)

[2.1.6 Приведение к 3НФ 40](#_Toc198587105)

[2.2 Проектирование архитектуры приложения 42](#_Toc198587106)

[2.3 Результаты проектирования 44](#_Toc198587107)

[Глава 3. Реализация системы 45](#_Toc198587108)

[3.1 Описание классов серверной части 45](#_Toc198587109)

[3.2 Схема графического интерфейса 52](#_Toc198587110)

[3.3 Характеристики поведения интерфейса 53](#_Toc198587111)

[3.4 Результаты реализации системы 66](#_Toc198587112)

[ЗАКЛЮЧЕНИЕ 67](#_Toc198587113)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 68](#_Toc198587114)

ВВЕДЕНИЕ

В мире существует большое количество предпринимателей, которые занимаются прокатом автомобилей. Только в одном городе Пермь открыты десятки учреждений, которые сдают автомобили в аренду.

Такой бизнес требует ведения учёта за арендами, хранения данных о них, об автомобилях, их техническом состоянии и другими подобными факторами. Без какой-либо информационной системы хранение данной информации может быть неудобным, а сохранение новых данных трудоёмким. В свою очередь, клиентам сервиса может быть неудобно или некомфортно бронировать автомобиль по телефону или лично. Существующие сайты прокатов автомобилей, очевидно, не предоставляют возможности выкладывать там объявления другого предпринимателя и вести там учёт аренд. Для объявлений, однако, можно использовать какой-либо популярный интернет-сервис для размещения объявлений, но подобные сервисы по-прежнему не позволяют вести учёт автомобилей и аренд, тем более автоматический.

Таким образом, создание информационной системы для учёта проката автомобилей должно быть актуальным решеннием.

Информационная система должна решать следующие задачи:

* предоставление клиентам возможности забронировать авто онлайн с автоматическим сохранением данных об аренде;
* предоставление администраторам функций для просмотра, добавления, редактирования и удаления необходимых данных об арендах и автомобилях, их техническом состоянии.

Объектом исследования является учёт проката автомобилей.

Предметом исследования является информационная система для учёта проката автомобилей.

Цель проекта – создание информационной системы для учёта проката автомобилей, которая должна облегчить ведение учёта администраторам проката и сделать более удобным бронирование автомобилей для клиентов.

Для достижения поставленной цели должны быть решены следующие задачи:

1. Провести анализ объекта автоматизации – процесса учета проката автомобилей.
2. Выполнить проектирование информационной системы.
3. Выполнить реализацию информационной системы.
4. Протестировать разработанную информационную систему.

В ходе анализа объекта автоматизации должна быть построена модель «as-is». Требования должны быть оформлены в виде технического задания по ГОСТ 19.201-78.

Проектирование выполняется в рамках объектно-ориентированного подхода с использованием диаграмм UML. Разработка базы данных основана на теории реляционных баз данных. При реализации системы будет использоваться язык программирования C# с использованием платформы ASP.NET Core для серверной части, а также библиотеки React для языка программирования JavaScript для клиентской части. Должно быть выполнено компонентное и системное тестирование приложения базы данных с использованием критериям черного ящика.

1. Анализ объекта автоматизации

В главе должны быть решены следующие задачи:

1. Анализ объекта автоматизации, в ходе которого должны быть выявлены проблемы, связанные с отсутствием или недостатками автоматизации учёта проката автомобилей, определены их причины, которые должны быть устранены при создании информационной системы, определяются общие требования к информационной системе.
2. Анализ существующих решений, в ходе которого оценивается их соответствие общим требованиям, возможность использования при реализации проекта.
3. Анализ требований к системе, в ходе которого должны быть определены как функциональные, так и нефункциональные требования. Результаты должны быть зафиксированы в техническом задании, оформленном в соответствии с ГОСТ 19.201-78.
   1. Анализ предметной области

Объектом автоматизации является учёт проката автомобилей, а именно их аренд, технических характеристик и состояния автомобилей.

Аренда автомобиля — это соглашение, по которому арендатору за плату предоставляется транспортное средство во временное владение и пользование. При аренде автомобилей есть необходимость в хранении большого количества информации. Во-первых, необходимо хранить некоторые персональные данные клиента, которые можно использовать в случае, если он совершит какие-либо противоправные действия, такие как нарушения ПДД КоАП РФ, или же хищения имущества проката. Также нужно хранить все характеристики каждого автомобиля, их идентификационные номера, а также актуальные данные о его техническом состоянии. Это важно, так как, обращаясь к статье 634 ГК РФ, арендодатель обязан осуществлять текущий и капитальный ремонт транспортного средства, а также по статье 639 ГК РФ в случае повреждения арендованного транспортного средства арендатор обязан возместить арендодателю причиненные убытки, если последний докажет, что повреждение транспортного средства произошли по обстоятельствам, за которые арендатор отвечает в соответствии с законом или договором аренды. Решением данных проблем могут стать бумажные носители, однако это крайне неудобный и устаревший способ хранения информации. Решением должна послужить разрабатываемая информационная система.

* + 1. Анализ процесса бронирования

При бронировании автомобиля от клиента может потребоваться следующая информация:

* ФИО;
* электронная почта;
* номер телефона;
* паспортные данные;
* данные водительского удостоверения;
* даты начала и конца аренды.

При бронировании автомобиля клиент должен указать эти данные. При этом, если автомобиль уже забронирован каким-либо клиентом в определённое время, другой клиент не должен иметь возможности забронировать этот же автомобиль в это же время. Также клиенту должен выдаваться какой-либо идентификационный номер, по которому при выдаче автомобиля можно определить, что бронь действительно принадлежит данному человеку. Эти данные помогут администраторам проката составить договор аренды, который поможет в случае каких-либо непредвиденных ситуаций. В случае какого-либо ущерба состоянию автомобиля или же при нарушении клиентом ПДД КоАП РФ, клиенту должен быть назначен штраф.

* + 1. Анализ автомобилей

Чтобы клиент мог выбрать подходящий ему автомобиль, он должен иметь доступ к следующей информации:

* марка;
* модель;
* год выпуска;
* мощность двигателя в л.с. (лошадиных силах);
* объём двигателя;
* тип двигателя (бензиновый или дизельный);
* тип трансмиссии (вариатор, роботизированная коробка передач, автоматическая коробка передач или механическая коробка передач);
* количество сидений;
* привод (полный, задний или передний);
* стоимость за сутки в рублях.

Клиент, вероятнее всего, будет выбирать автомобиль для аренды исходя из этих факторов.

Однако, перед выдачей автомобиля администратор должен убедиться в исправности транспортного средства и в случае каких-либо технических неполадок запретить выдачу автомобиля. Также администратор должен проверить состояние автомобиля, технический осмотр которого должен провести механик и заполнить информацию о его состоянии. Это важно, чтобы принять решение о том, возникла ли техническая неисправность после клиента или же клиент не виноват в ней. От этого будет зависеть назначение штрафа клиенту. Механику важно проверить состояние необходимых для работы узлов и агрегатов и вынести вердикт по состоянию автомобиля. Если состояние критическое, то это будет означать, что автомобиль не должен быть допущен к эксплуатации. Если в прокате имеется более одного подразделения, клиент должен быть проинформирован о том, где находится автомобиль.

* 1. Анализ существующих решений

Для анализа существующих решений необходимо выполнить следующие задачи:

1. Выделить критерии оценки решений.
2. Определить список рассматриваемых решений.
3. Провести краткий обзор отобранных решений и оценить их по выделенным критериям.
   * 1. Выделение критериев оценки

Функции, которые доолжна выполнять информационная система, представлены далее.

* взаимодействие с данными: обеспечение выполнения CRUD-операций для данных об автомобилях и арендах;
* удобный и понятный интерфейс, для использования системы не нужно дополнительное обучение;
* возможность бронировать автомобиль прямо в системе;
* можно увидеть все важные характеристики автомобиля;
* возможность выложить объявление об автомобиле;
* мониторинг технического состояния автомобиля;
* возможность просмотра всех аренд и назначение штрафов;
* есть аутентификация и авторизация для разграничения доступа к некоторым функциям.

Для обеспечения этих функций требуется приложение, которое позволяло бы работать с какой-либо базой данных, при этом эта база данных должна быть открыта для всех пользователей, чтобы в неё можно было вносить изменения.

Система должна содержать детальную информацию об автомобилях и обеспечивать полностью автоматизированный процесс бронирования без необходимости дополнительных звонков или личного взаимодействия.

Интерфейс системы должен быть интуитивно понятным, упрощающим ввод и управление данными, их просмотр и редактирование, а также выполнение системных запросов. При этом пользователям не требуются специальные знания SQL или других технических аспектов для выполнения базовых операций в рамках предметной области.

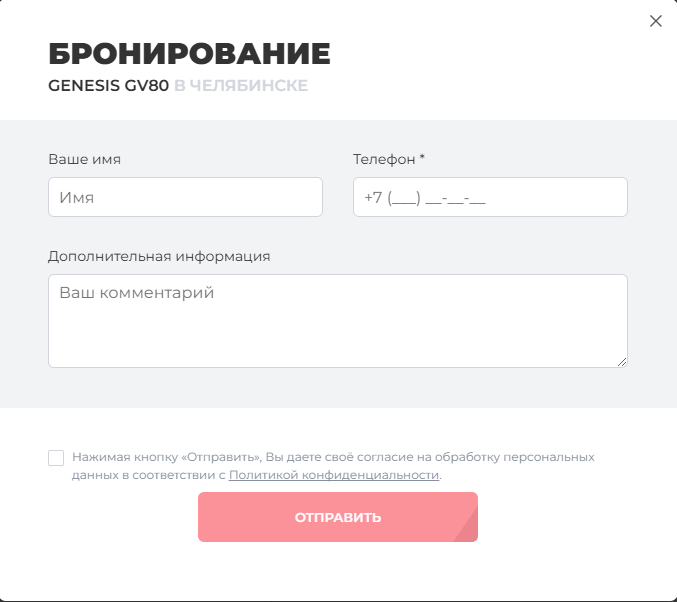
* + 1. Анализ существующих решений

Будут анализироваться следующие ресурсы и инструменты:

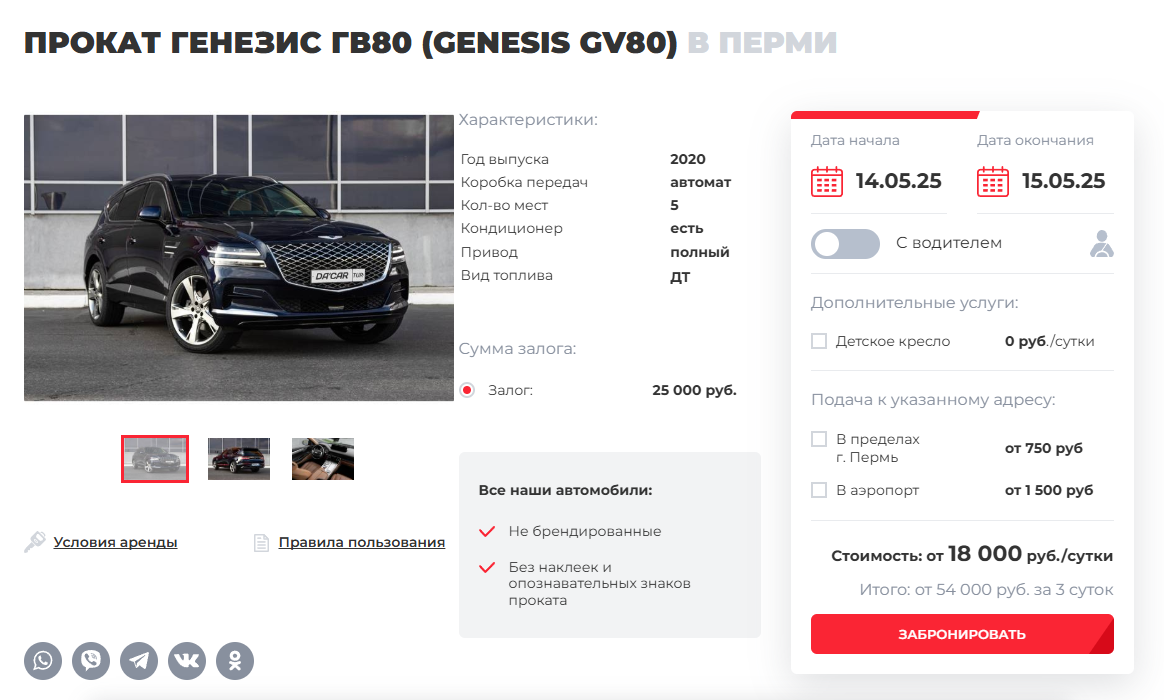
1. Сайт компании «perm.dacartur.com», занимающееся прокатом автомобилей.
2. «Авито» – интернет-сервис для размещения объявлений.
3. «1С» – программа для автоматизации бизнес-процессов.

Выбор именно этих решений обосновывается следующим: данные ресурсы так или иначе имеют возможность хранить данные об арендах, автомобилях и других сущностях. Это также наиболее популярные ресурсы, которые относятся («dacartur») или теоретически могут относится к прокату автомобилей (Авито, 1С).

Сайт «dacartur» является самым посещаемым сайтом по прокату автомобилей в городе Пермь. Он имеет понятный и удобный интерфейс, фильтрацию, клиент имеет возможность забронировать автомобиль прямо на сайте (рисунок 1) и увидеть все важные характеристики данного транспортного средства (рисунок 2), однако данный сайт принадлежит только одной компании, а это значит, что нет никакой возможности выкладывать там свои объявления и вести учёт аренд с использованием этой системы, и есть ли вообще функция по учёту аренд и технического состояния автомобилей – неизвестно, так как доступ к панели администратора имеет только сотрудник данной компании. Там также нет аутентификации, которая могла бы быть удобной при бронировании авто.

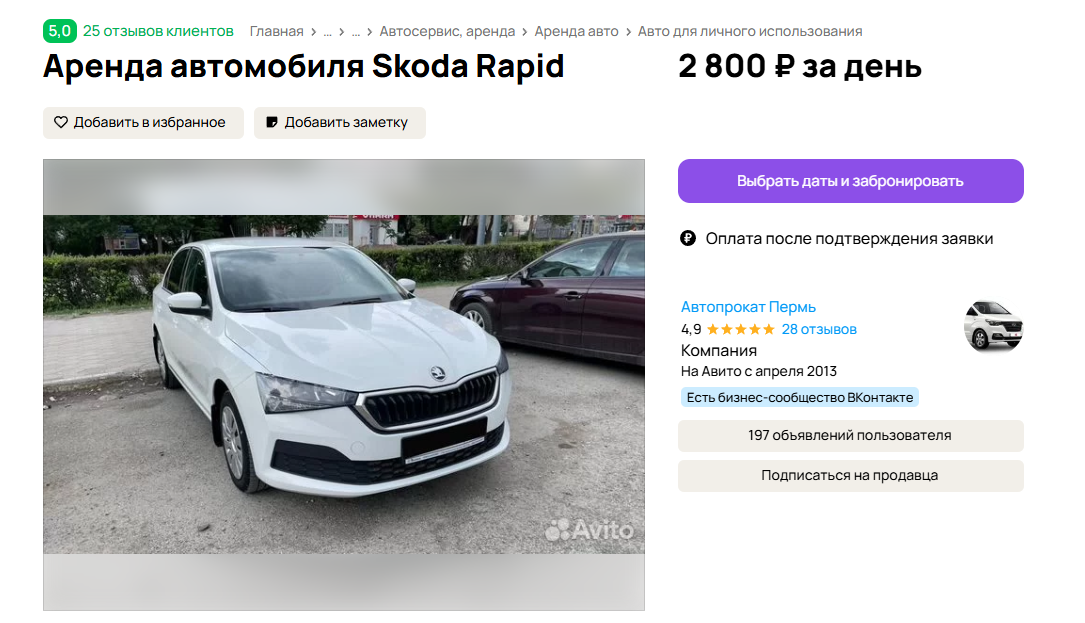


***Рисунок 1 - модальное окно для бронирования автомобиля «dacartur»***



***Рисунок 2 - страница характеристик автомобиля «dacartur»***

Интернет-сервис для размещения объявлений «Авито» – это один из самых популярных сервисов для объявлений с удобным и понятым интерфейсом (рисунок 3). Здесь можно разместить всю необходимую информацию об автомобиле, а клиент сможет забронировать автомобиль посредством функций данного сервиса, однако это все, что позволяет делать данный сервис., а прокат, в свою очередь, не сможет собрать никакой информации об аренде, а также не сможет хранить данные об автомобиле и его техническом состоянии. С другой стороны, этот сервис предоставляет возможность аутентификации, которая, однако, полезна скорее только самому «Авито» и не несёт никакой пользы для проката автомобилей.



***Рисунок 3 - объявление об аренде на «Авито»***

Что касается «1С», эта программа подойдёт только для учёта аренд и хранения данных об автомобилях и их технических состояниях. Данное программное обеспечение, в рамках выбранной предметной области, не получится использовать для размещения объявлений и бронирования автомобилей. Да и информацию об арендах, которая могла быть получена автоматически после бронирования клиентом автомобиля, пришлось бы заносить в базу данных самостоятельно. Также программа требует хотя бы базового обучения для работы с ней, так как данное программное обеспечение включает в себя большое количество функций, схема использования которого может быть неочевидна. Для соответствия всем требуемым свойствам можно использовать «1С» совместно с «Авито», однако это не самый удобный вариант работы.

Результаты анализа существующих решений собраны в таблице 1. Для оценки соответствия рассмотренных систем выделенным критериям использовалась шкала от 0 до 3, где 0 – полностью не удовлетворяет, 3 – полностью удовлетворяет.

Таблица 1 - Результаты анализа

| **Система** | **CRUD-операции** | **Интерфейс** | **Бронирование** | **Вывод характеристик** | **Возможность выложить Объявление** | **Мониторинг ТС и аренд** | **Аутентификация и авторизация** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| «dacartur» | 0 | 3 | 3 | 3 | 0 | 0 | 0 |
| «Авито» | 1 | 3 | 2 | 3 | 3 | 0 | 1 |
| «1С» | 3 | 1 | 0 | 3 | 0 | 3 | 0 |

По результатам анализа установлено, что на данный момент не существует системы учета проката автомобилей, соответствующей одновременно всем необходимым свойствам, и которая при этом давала бы доступ для хранения данных не только одной компании. Следовательно, разработка такой системы является актуальной задачей.

* 1. Анализ требований

По сформулированным выше требуемым свойствам информационной системы должны быть определены функциональные и нефункциональные требования. Функциональные требования – это определения того, какие задачи, функции и операции должна выполнять информационная система.

Анализ нефункциональных требований – это определение свойств и ограничений системы, не относящихся к ее функциям.

Функциональные требования к системе:

* аутентификация и авторизация;
* просмотр, добавление, редактирование и удаление данных об автомобилях (для пользователя, авторизованного, как «администратор»);
* добавление данных об аренде после бронирования автомобиля клиентом;
* просмотр данных об аренде (для пользователя, авторизованного, как «администратор»);
* назначение штрафов клиентам (для пользователя, авторизованного, как «администратор»);
* добавление данных о техническом состоянии автомобиля после технического осмотра (для пользователя, авторизованного, как «механик»).

Нефункциональные требования к системе:

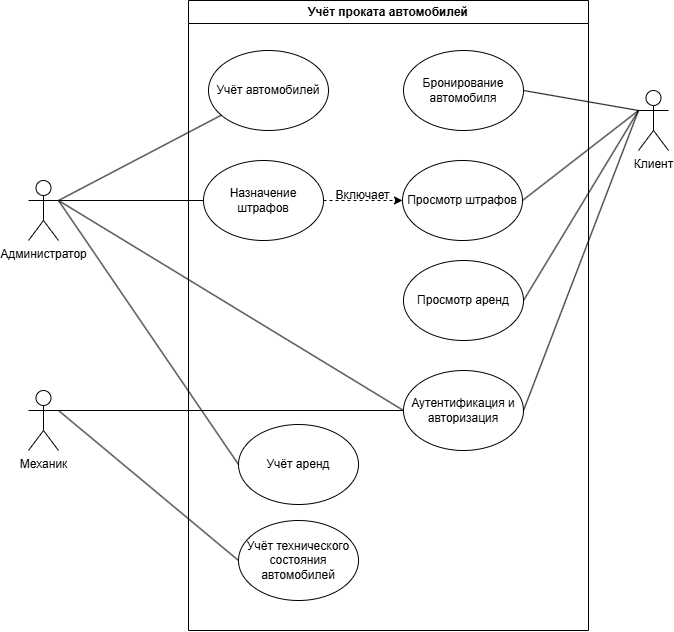
1. Кросс-браузерная совместимость – полноценная работа во всех современных веб-браузерах
2. Оптимизированная структура данных – база должна соответствовать требованиям 3НФ (третьей нормальной формы)
3. Целостность данных – поддержка каскадных операций при изменении и удалении записей, обеспечение физической целостности БД
4. Интуитивно понятный интерфейс – простота и удобство взаимодействия для пользователей
5. Отказоустойчивость – стабильная работа без сбоев в круглосуточном режиме
6. Высокая производительность – минимальное время отклика на пользовательские запросы
   * 1. Анализ процесса учёта проката автомобилей

В системе могут быть три типа пользователей (акторов): клиент, механик и администратор. Все действия по учёту проката выполняет администратор, механик заносит данные о техническом состоянии, а клиент имеет возможность бронировать авто, а также просматривать информацию о своих арендах и смотреть информацию о штрафах. Все акторы должны иметь возможность авторизоваться в системе.

Процесс учёта проката автомобилей администратором можно разделить на следующие подпроцессы:

* учёт аренд;
* учёт автомобилей;
* учёт технического состояния автомобиля;
* учёт штрафов.

Диаграмма прецедентов процесса учета проката автомобилей представлена на рисунке 4.



***Рисунок 4 - процесс учёта проката автомобилей***

Все прецеденты, кроме просмотра штрафов и аренд клиентом, выделенные на рисунке 4, являются обобщёнными, поэтому для них требуется декомпозиция. Для просмотра штрафов клиентом описание приведено ниже:

Название: просмотр штрафов.

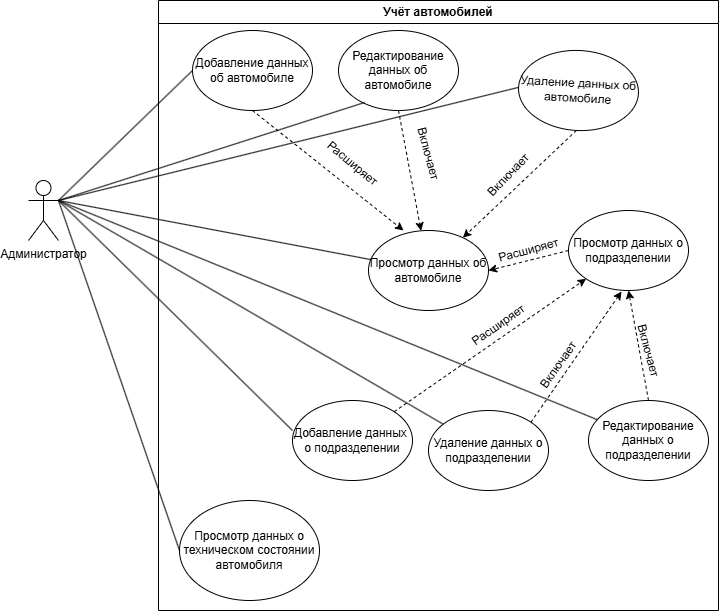
Описание: клиент, находясь на главной, переходит посредством меню на страницу просмотра аренд, на которой также будут отображены штрафы клиента (при наличии).

Название: просмотр аренд.

Описание: клиент, находясь на главной, переходит посредством меню на страницу просмотра аренд, на которой будут отображены аренды клиента (при наличии).

* + 1. Анализ процесса учёта автомобилей

Диаграмма прецедентов процесса учета автомобилей представлена на рисунке 5.



***Рисунок 5 - Процесс учёта автомобилей***

Дальнейшая декомпозиция прецендентов не требуется. Далее приведены описания прецендентов:

Название: просмотр данных об автомобиле.

Описание: администратор, находясь в панели администратора, видит карточки всех автомобилей. Если нажать на карточку, или на кнопку «Подробнее», расположенную на карточке, то откроется страница просмотра данных об автомобиле. Данные должны загружаться из базы данных.

Название: Добавление данных об автомобиле.

Описание: администратор, находясь в панели администратора, нажимает на кнопку «Добавить автомобиль». Открывается страница добавления автомобиля с формами, в которых необходимо заполнить все данные об автомобиле. После нажатия кнопки «Добавить» на странице добавления автомобиля, администратор должен быть возвращён в админ-панель, где с карточками уже имеющихся автомобилей должна отображаться карточка с данными о новом автомобиле. Данные также добавляются в базу данных.

Название: Редактирование данных об автомобиле.

Описание: администратор, находясь в панели администратора, нажимает на кнопку «Редактировать автомобиль», расположенную на карточке автомобиля. Открывается страница редактирования автомобиля с формами, в которых необходимо обновить необходимые данные об автомобиле. После нажатия кнопки «Сохранить» на странице редактирования автомобиля, администратор должен быть возвращён в админ-панель, где карточка с данным автомобилем имеет отредактированные данные. Данные также обновляются в базе данных

Название: Удаление данных об автомобиле.

Описание: администратор, находясь в панели администратора, нажимает на кнопку «Удалить автомобиль», расположенную на карточке автомобиля. После нажатия этой кнопки карточка с данным автомобилем удаляется. В базе данных также происходит каскадное удаление.

Название: Просмотр данных о техническом состоянии автомобиля.

Описание: администратор, находясь в панели администратора, видит карточки всех автомобилей. Если нажать на карточку, или на кнопку «Подробнее», расположенную на карточке, то откроется страница просмотра данных об автомобиле, на которой также должна быть выведена информация о техническом состоянии автомобиля после последнего технического осмотра. Данные должны загружаться из базы данных.

Название: просмотр данных о подразделении.

Описание: администратор, находясь в панели администратора, нажимает на кнопку в меню «Подразделения», после чего попадает на страницу просмотра подразделений, где он видит данные о подразделениях в виде таблицы.

Название: Добавление данных об автомобиле.

Описание: администратор, находясь в панели администратора, нажимает на кнопку «Добавить автомобиль». Открывается страница добавления автомобиля с формами, в которых необходимо заполнить все данные об автомобиле. После нажатия кнопки «Добавить» на странице добавления автомобиля, администратор должен быть возвращён в админ-панель, где с карточками уже имеющихся автомобилей должна отображаться карточка с данными о новом автомобиле.

Название: Добавление данных об подразделении.

Описание: администратор, находясь на странице просмотра подразделений, нажимает на кнопку «Добавить подразделение». Открывается страница добавления подразделения с формами, в которых необходимо заполнить все данные об подразделении. После нажатия кнопки «Добавить» на странице добавления подразделения, администратор должен быть возвращён на страницу добавления подразделений. Теперь у него есть возможность добавить автомобиль в это подразделение.

Название: Редактирование данных о подразделении.

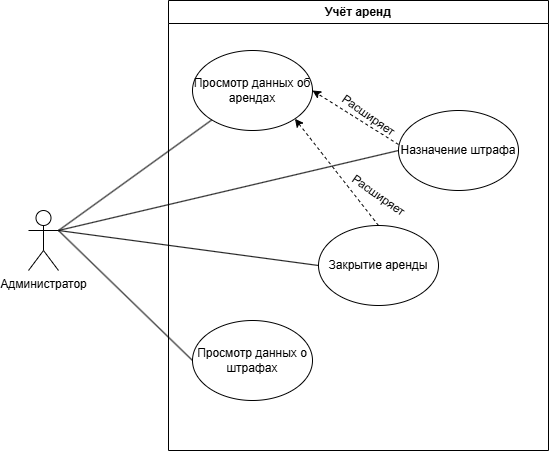
Описание: администратор, находясь на странице просмотра подразделений, нажимает на кнопку «Редактировать данные о подразделении», расположенную на строке с подразделением. Открывается страница редактирования подразделения с формами, в которых необходимо обновить необходимые данные о подразделении. После нажатия кнопки «Сохранить», администратор должен быть возвращён на станицу просмотра подразделений. Данные также обновляются в базе данных.

Название: Удаление данных о подразделении.

Описание: администратор, находясь на странице просмотра подразделений, нажимает на кнопку «Удалить подразделение», расположенную на строке с подразделением. После нажатия этой кнопки в базе данных также происходит каскадное удаление.

* + 1. Анализ процесса учёта аренд

Диаграмма прецедентов процесса учета аренд представлена на рисунке 6.



***Рисунок 6 - процесс учёта аренд***

Дальнейшая декомпозиция прецендентов не требуется. Далее приведены описания прецендентов:

Название: просмотр данных об арендах.

Описание: администратор, находясь в панели администратора, нажимает на кнопку «Аренды». Открывается страница учёта аренд, на которой выведены текущие аренды (которые ещё не истекли), а также законченные аренды, закрытие которых требует подтверждения или назначения штрафа (в виде таблицы). При нажатии «Найти аренды» выводятся данные всех аренд в зависимости от выбранного автомобиля или паспортных данных. При нажатии кнопки «Загрузить все аренды» выводятся все аренды.

Название: назначение штрафа.

Описание: администратор, находясь на странице учёта аренд, на которой выведены законченные аренды, закрытие которых требует подтверждения или назначения штрафа. При нажатии на кнопку «Назначить штраф» открывается страница назначения штрафа, на которой вводятся все необходимые данные. После нажатия кнопки «Назначить штраф», администратор должен быть возвращён на страницу учёта аренд.

Название: закрытие аренды

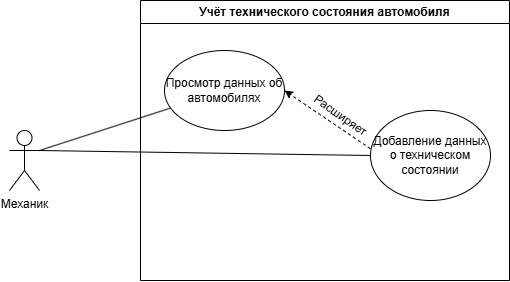
Описание: администратор, находясь на странице учёта аренд, на которой выведены законченные аренды, закрытие которых требует подтверждения или назначения штрафа. При нажатии на кнопку «Закрыть аренду» данная аренда считается завершённой и больше не отображается.

Название: просмотр данных о штрафах.

Описание: администратор, находясь на странице учёта аренд, нажимает на кнопку «Штрафы». Открывается страница штрафов, на которой выводятся все штрафы. Также могут быть найдены штрафы в зависимости от выбранного id аренды.

* + 1. Анализ процесса учёта технического состояния автомобиля

Диаграмма прецедентов процесса учета технического состояния автомобиля представлена на рисунке 7.



***Рисунок 7 - процесс учёта технического состояния автомобиля***

Название: просмотр данных об автомобиле.

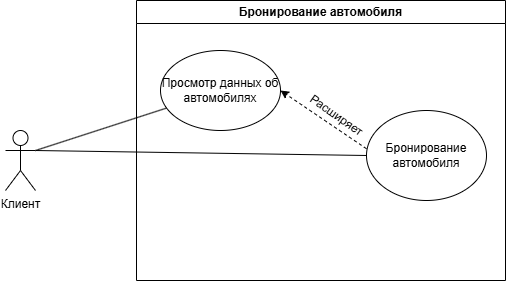
Описание: механик, находясь на «сервисной» странице, видит карточки всех автомобилей. Если нажать на карточку, или на кнопку «Подробнее», расположенную на карточке, то откроется страница просмотра данных об автомобиле. Данные должны загружаться из базы данных.

Название: добавление данных о техническом состоянии

Описание: механик, находясь на «сервисной» странице, переходит по нажатию на кнопку «Добавить данные тех. осмотра» на карточке автомобиля или на странице просмотра данных об автомобиле, переходит на страницу добавления данных о техническом осмотре автомобиля, где он заполняет все необходимые данные по техническому осмотру. После заполнения всех данных и нажатия на кнопку «Добавить» механик должен быть возвращён на «сервисную» страницу.

* + 1. Анализ процесса бронирование автомобиля

Диаграмма прецедентов процесса бронирования автомобиля представлена на рисунке 8.



***Рисунок 8 - процесс бронирования автомобиля***

Название: просмотр данных об автомобиле.

Описание: клиент, находясь на главной странице, видит карточки всех автомобилей. Если нажать на карточку, или на кнопку «Подробнее», расположенную на карточке, то откроется страница просмотра данных об автомобиле. Данные должны загружаться из базы данных.

Название: бронирование автомобиля.

Описание: клиент, находясь на главной странице, видит карточки всех автомобилей. Если нажать на карточку, или на кнопку «Подробнее», расположенную на карточке, то откроется страница просмотра данных об автомобиле. Данные должны загружаться из базы данных. Для клиента на этой странице должна быть кнопка «Забронировать», после нажатия на которую открывается страница бронирования автомобиля. На ней клиент должен заполнить форму с необходимыми для аренды (договора аренды) персональные данные. После нажатия кнопки «Забронировать клиент должен быть возвращён на главную страницу.

* + 1. Анализ процесса аутентификации и авторизации

Диаграмма прецедентов процесса аутентификации и авторизации автомобиля представлена на рисунке 9.



***Рисунок 9 - процесс аутентификации и авторизации***

Название: Регистрация в системе.

Описание: неавторизованный пользователь, находясь на главной странице или на странице входа в систему, нажимает на кнопку «Зарегистрироваться», после чего попадает на страницу регистрации, где после заполнения формы регистрации и нажатия кнопки «Зарегистрироваться» пользователь должен быть направлен на страницу входа в систему, где ему должно быть предложено авторизоваться.

Название: Вход в систему.

Описание: неавторизованный пользователь, находясь на главной странице или на странице регистрации в системе, нажимает на кнопку «Войти», после чего попадает на страницу входа в систему, где после заполнения формы входа и нажатия кнопки «Войти» пользователь должен быть направлен на главную страницу.

Название: аутентификация.

Описание: после ввода пользователем данных в форму входа, система должна установить подлинность «личности» пользователя.

Название: авторизация.

Описание: после ввода пользователем данных в форму входа и установления «личности» пользователя, система должна определить права, которыми обладает пользователь для разграничения доступа к некоторым функциям.

* 1. Результаты анализа

В результате анализа объекта автоматизации, были выполнены следующие задачи:

1. Рассмотрены основные понятия, документы и проблемы.
2. Проведено сравнение наиболее популярных и подходящих под предметную область решений, таких как сайт компании «dacartur», «Авито» и «1С».
3. Разработаны требования к системе.
4. Определены сценарии работы пользователей, построены диаграммы прецедентов, приведено их подробное описание.

В результате анализа составлено техническое задание на разработку информационной системы (см. приложение А).

1. Проектирование информационной системы

При проектировании информационной системы требуется решить несколько ключевых задач. Прежде всего, нужно выполнить проектирование базы данных, так как на ней, в первую очередь, будет основываться вся система.

Также необходимо спроектировать архитектуру и модель поведения приложения, запросы и некоторые функции для обращения к БД, задать некоторые требования для входных данных.

* 1. Проектирование базы данных

В процессе проектирования базы данных необходимо определить операции ввода данных и соответствующие требования к ним, а также выявить функциональные зависимости между атрибутами для последующей декомпозиции универсального отношения путем нормализации, обеспечив при этом соответствие базы данных как минимум третьей нормальной форме.

* + 1. Описание операций

Основными пользователями системы являются администраторы, решающие основные задачи, связанные с целостностью автопарка, а также механик, следящий за исправностью автомобилей и делающий отметку о необходимости их обслуживания. Операции, которые должны быть автоматизированы с помощью системы описаны ниже (таблица 2):

Таблица 2 - Операции автоматизируемого бизнес-процесса «Учёт проката автомобилей»

| **Название операции** | **Исполнитель** | **Периодичность выполнения** | **Входные данные (документы)** | **Выходные данные (документы)** |
| --- | --- | --- | --- | --- |
| Ввод информации об автомобиле | Администратор проката | При приобретении прокатом нового автомобиля | Паспорт транспортного средства, некоторые характеристик, не представленные в ПТС, изображение авто, подразделение, в котором располагается автомобиль | Запись с характеристиками автомобиля в БД |
| Ввод информации об арендаторе | Клиент | При бронировании | Паспорт, водительское удостоверение | Запись в БД с документами, удостоверяющими личность арендатора |
| Ввод информации о подразделении | Администратор проката | Когда появляется новое место хранения авто | Адрес, номер телефона подразделения | Запись в БД с данными о подразделении |
| Ввод информации об учетной записи клиента | Клиент | При регистрации клиента в ИС | Фамилия и имя клиента,  Эл. почта клиента, пароль | Запись с данными о клиенте |
| Ввод данных об аренде | Клиент | При бронировании автомобиля клиентом | Данные учетной записи клиента, номер телефона, желаемый автомобиль, время | Запись с данными о брони автомобиля |
| Ввод данных о штрафах | Администратор проката | После осмотра автомобиля на наличие повреждений или при получении штрафа за нарушение ПДД | Выписанный штраф | Запись с данными о штрафе |
| Ввод информации о техническом обслуживании автомобиля | Механик | При возврате клиентом автомобиля | Данные о техническом состоянии автомобиля | Запись с данными о степени износа автомобиля |
| Просмотр данных об автомобиле | Любой пользователь | При запросе на получение данных | Нет | Данные об автомобиле |
| Просмотр данных о подразделении | Администратор проката | При запросе на получение данных | Нет | Данные о подразделениях |
| Просмотр данных о техническом состоянии автомобилей | Механик и Администратор проката | При запросе на получение данных | Нет | Данные о техническом состоянии автомобиля |
| Просмотр данных о штрафах | Администратор проката | При запросе на получение данных | Нет | Данные о штрафах |
| Просмотр данных об арендах | Администратор проката | При запросе на получение данных | Нет | Данные об арендах |
| Изменение данных об автомобиле | Администратор проката | При запросе на изменение данных | VIN номер автомобиля | Обновлённые данные об автомобиле |
| Изменение данных о подразделениях | Администратор проката | При запросе на изменение данных | Адрес подразделения | Обновлённые данные о подразделении |
| Удаление данных об автомобиле | Администратор проката | При запросе на удаление данных | VIN номер автомобиля | Нет |
| Удаление данных о подразделении | Администратор проката | При запросе на удаление данных | Адрес подразделения | Нет |

* + 1. Описание атрибутов для проектирования БД

Для выполнения описанных выше операций необходимо, чтобы в БД хранилась информация, описанная в таблице 3:

Таблица 3 - атрибуты

| **Имя атрибута** | **Тип данных** | **Значение по умолчанию** | **Формат ввода (маска)** | **Ограничение на значение атрибута** | **Комментарий** |
| --- | --- | --- | --- | --- | --- |
| VIN-номер | Текст | Нет | 00000000000000000 | Атрибут является уникальным и обязательным к заполнению | 17-символьный код, используются как цифры, так и латинские буквы |
| Марка | Короткий текст | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии с ПТС |
| Модель | Короткий текст | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии с ПТС |
| Год выпуска | Числовой | Нет | Нет | Значение атрибута обязательно для заполнения, не может быть больше текущей даты, не может быть отрицательным | Вводится в соответствии с ПТС |
| Мощность двигателя (л.с.) | Числовой | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии с ПТС |
| Рабочий объём двигателя | Числовой | Нет | 0.0 | Значение атрибута обязательно для заполнения | Вводится в соответствии с ПТС |
| Тип двигателя | Короткий текст | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии с ПТС |
| Тип трансмиссии | Короткий текст | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии с характеристиками автомобиля |
| Количество мест | Числовой | Нет | Нет | Значение атрибута обязательно для заполнения, не может быть отрицатнльным | Вводится в соответствии с характеристиками автомобиля |
| Привод | Короткий текст | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии с характеристиками автомобиля |
| Стоимость за сутки (руб) | Числовой | Нет | Нет | Значение атрибута обязательно для заполнения, не может быть отрицательным | Вводится в соответствии с документами |
| Регистрационный номер | Короткий текст | Нет | о000оо000 | Значение атрибута обязательно для заполнения | Вводится в соответствии с документами |
| Код учетной записи | Число | Нет | Нет | Атрибут является уникальным и обязательным к заполнению. Порядковый номер - автоинкрементация | Вводится автоматически, порядковый номер записи |
| Пароль | Текст | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии со вводом пользователя |
| Электронная почта | Текст | Нет | Нет | Значение атрибута уникально и обязательно для заполнения | Вводится в соответствии со вводом пользователя |
| Фамилия | Текст | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии со вводом пользователя |
| Имя | Текст | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии со вводом пользователя |
| Отчество | Текст | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии со вводом пользователя |
| Код аренды | Число | Нет | Нет | Атрибут является уникальным и обязательным к заполнению. Порядковый номер - автоинкрементация | Вводится автоматически, порядковый номер записи |
| Закрыта ли аренда | Логический | Нет | 0 или 1 | Значение атрибута обязательно для заполнения | Показывает, можно ли закрыть аренду без претензий к клиенту и штрафов |
| Номер телефона | Короткий текст | Нет | +7000000000 или 80000000000 | Значение атрибута обязательно для заполнения | Вводится в соответствии со вводом пользователя, 10-значный российский номер телефона |
| Дата начала аренды | Дата | Нет | 00.00.0000 | Значение атрибута обязательно для заполнения | Вводится в соответствии со вводом пользователя, короткий формат даты |
| Дата конца аренды | Дата | Нет | 00.00.0000 | Значение атрибута обязательно для заполнения | Вводится в соответствии со вводом пользователя, короткий формат даты |
| Общая стоимость аренды | Число | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится автоматически на основе стоимости автомобиля за сутки и количества дней в аренде |
| Код обслуживания | Число | Нет | Нет | Атрибут является уникальным и обязательным к заполнению. Порядковый номер - автоинкрементация | Вводится автоматически, порядковый номер записи |
| Дата проведения обслуживания | Дата | Нет | 00.00.0000 | Значение атрибута обязательно для заполнения | Вводится в соответствии со вводом механика |
| Тип обслуживания | Короткий текст | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии со вводом механика |
| Описание состояния автомобиля | Текст | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии со вводом механика |
| Стоимость обслуживания | Число | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии со вводом механика |
| Статус обслуживания | Короткий текст | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится в соответствии со вводом механика |
| Серия и номер паспорта арендатора | Текст | Нет | 0000 000000 | Значение атрибута обязательно для заполнения | Вводится в соответствии с документом об аренде |
| Серия и номер водительского удостоверения арендатора | Текст | Нет | 0000 000000 | Значение атрибута обязательно для заполнения | Вводится в соответствии с документом об аренде |
| Код штрафа | Число | Нет | Нет | Атрибут является уникальным и обязательным к заполнению. Порядковый номер - автоинкрементация | Вводится автоматически после добавления новой записи |
| Формулировка штрафа | Текст | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится на основе того, за что был выдан штраф |
| Сумма штрафа | Число | Нет | Нет | Значение атрибута обязательно для заполнения | Вводится после подсчёта ущерба или по приходе штрафа за нарушение ПДД |
| Код изображения | Число | Нет | Нет | Атрибут является уникальным и обязательным к заполнению | Код изображения автомобиля |
| Название изображения | Текст | Нет | Нет | Значение атрибута обязательно для заполнения | Название файла изображения с расширением |
| Данные изображения | Массив байтов | Нет | Нет | Значение атрибута обязательно для заполнения | Изображение хранится в базе данных в виде массива байтов и должно быть конвертировано обратно в изображение при получении из базы данных |
| Тип контента | Текст | Нет | Нет | Значение атрибута обязательно для заполнения | Тип файла изображения |
| Код подразделения | Число | Нет | Нет | Атрибут является уникальным и обязательным к заполнению | Код подразделения, в котором хранятся автомобили |
| Адрес подразделения | Текст | Нет | Нет | Значение атрибута обязательно для заполнения | - |
| Номер телефона подразделения | Короткий текст | Нет | +7000000000 или 80000000000 | Значение атрибута обязательно для заполнения | Номер телефона для обращения за получением более подробной информации |

* + 1. Функциональные зависимости

При установлении функциональных зависимостей при проектировании БД необходимо учесть следующие связи между атрибутами:

* 1ФЗ: По VIN-номеру можно однозначно определить такие характеристики автомобиля как: марка, модель, год выпуска, мощность двигателя, рабочий объём двигателя, тип двигателя, тип трансмиссии, количество мест, привод, стоимость за сутки, регистрационный номер.
* 2ФЗ: По коду учетной записи можно однозначно определить пароль, электронную почту, фамилию, имя и отчество пользователя.
* 3ФЗ: По коду аренды можно однозначно определить номер телефона пользователя, желаемый автомобиль по VIN-номеру, а также дату начала и конца аренды, серию и номер паспорта и водительского удостоверения арендатора.
* 4ФЗ: По коду изображения можно однозначно определить данные изображения, его название и тип контента.
* 5ФЗ: По коду обслуживания можно однозначно определить информацию по дате проведения обслуживания, типе обслуживания, описании обслуживания, стоимости обслуживания и статусе обслуживания.
* 6ФЗ: По коду штрафа можно однозначно определить информацию по формулировке штрафа, сумме штрафа.
* 7ФЗ: По коду подразделения можно однозначно определить адрес подразделения.
  + 1. Универсальное отношение

Отношение находится в первой нормальной форме (1НФ) тогда и только тогда, когда в любом допустимом значении отношения каждый его кортеж содержит только одно значение для каждого из атрибутов.

Универсальное отношение включает следующие атрибуты:

1. VIN-номер.
2. Марка.
3. Модель.
4. Год выпуска.
5. Мощность двигателя (л.с.).
6. Рабочий объём.
7. Двигателя.
8. Тип двигателя.
9. Тип трансмиссии.
10. Количество мест.
11. Привод.
12. Стоимость за сутки (руб).
13. Регистрационный номер.
14. Суммарная выручка.
15. Код учетной записи.
16. Пароль.
17. Электронная почта.
18. Фамилия.
19. Имя.
20. Код аренды.
21. Номер телефона.
22. Дата начала аренды.
23. Дата конца аренды.
24. Закрыта ли аренда.
25. Общая стоимость аренды.
26. Код изображения.
27. Название изображения.
28. Данные.
29. Тип изображения.
30. Код обслуживания.
31. Дата проведения обслуживания.
32. Тип обслуживания.
33. Описание состояния автомобиля.
34. Стоимость обслуживания.
35. Статус обслуживания.
36. Серия и номер паспорта арендатора.
37. Серия и номер водительского удостоверения арендатора.
38. Код штрафа.
39. Формулировка штрафа.
40. Сумма штрафа.
41. Код подразделения.
42. Адрес подразделения.
43. Номер телефона подразделения.

Сформирован первичный ключ:

Код учетной записи.

Данный ключ является уникальным для каждой строки, позволяет однозначно идентифицировать запись.

Для того чтобы обеспечить выполнение требования атомарности, в каждой строке таблицы необходимо вводить все значения указанных атрибутов, не группируя данные. Однако могут возникнуть аномалии:

Ввода (При появлении нового автомобиля в автопарке, должны быть сразу заполнены и данные о техническом состоянии, хотя осмотр мог быть ещё не проведён, также должна сразу появиться информация о аренде, хотя автомобиль ещё не был в аренде).

Редактирования (Если необходимо отредактировать электронную почту у одного пользователя, то придется редактировать в каждой записи, в которой фигурирует данный пользователь)

Удаления (Если мы хотим удалить запись о брони автомобиля, то удалиться также и запись с учетной записи пользователя, которую, возможно, не хотелось бы терять).

* + 1. Приведение к 2НФ

Отношение находится во 2НФ, если оно находится в 1НФ и каждый не ключевой атрибут неприводимо зависит от Первичного Ключа (ПК).

Имеются следующие частичные зависимости:

VIN-номер определяет:

* Марка.
* Модель.
* Год выпуска.
* Мощность двигателя (л.с.).
* Рабочий объём двигателя.
* Тип двигателя.
* Тип трансмиссии.
* Количество мест.
* Привод.
* Стоимость за сутки (руб).
* Регистрационный номер.
* Код изображения.
* Название изображения.
* Данные.
* Тип изображения.
* Код обслуживания.
* Дата проведения обслуживания.
* Тип обслуживания.
* Описание состояния автомобиля.
* Стоимость обслуживания.
* Статус обслуживания.
* Код подразделения.
* Адрес подразделения.
* Номер телефона подразделения.

Указанные атрибуты не зависят от кода аренды.

Код учетной записи определяет:

* Пароль.
* Электронная почта.
* Фамилия.
* Имя.

Указанные атрибуты также не зависят от кода аренды.

Необходимо выделить в отдельные таблицы:

Сведения об автомобиле, где VIN-номер становится первичным ключом.

Сведения об учетной записи, где код учетной записи становится первичным ключом.

Таким образом, имеем следующие отношения (таблицы):

Автомобили в наличии (VIN-номер, Марка, Модель, Год выпуска, Мощность двигателя (л.с.), Рабочий объём двигателя, Тип двигателя, Тип трансмиссии, Количество мест, Привод, Стоимость за сутки (руб), Регистрационный номер, Код обслуживания, Дата проведения обслуживания, Тип обслуживания, Описание состояния автомобиля, Стоимость обслуживания, Статус обслуживания, Код изображения, Название изображения, Данные, Тип изображения, Код подразделения, Адрес подразделения, Номер телефона подразделения.)

Учётные записи (Код учетной записи, Пароль, Электронная почта, Фамилия, Имя, Отчество)

Аренды (Код аренды, Код учетной записи, Номер телефона, VIN-номер, Дата начала аренды, Дата конца аренды, Серия и номер паспорта арендатора, Серия и номер водительского удостоверения арендатора, Код штрафа, Формулировка штрафа, Сумма штрафа, Общая стоимость аренды, Закрыта ли аренда.)

В приведённых отношениях отсутствуют частичные функциональные зависимости. Таким образом, отношения находятся во 2НФ (пунктиром отмечены внешние ключи).

Однако при работе с данными могут возникнуть аномалии:

1. Ввода (при вводе данных автомобиле по-прежнему необходимо заполнять сразу и данные о тех. обслуживании, хотя оно могло и не проводиться и информации о нём может не быть).
2. Редактирования (При редактировании, например, стоимость за сутки у конкретного автомобиля, то нужно редактировать все записи, где есть данный автомобиль, а записей с конкретным автомобилем может быть много из-за того, что тех. обслуживаний может быть несколько, тогда также нарушается свойство ключа об уникальности).
3. Удаления (при удалении, например, записи с одним из технических обслуживаний, также удалится и информация о характеристиках автомобиля, в чём могло не быть необходимости)
   * 1. Приведение к 3НФ

Отношение находится в 3НФ, когда находится во 2НФ и каждый неключевой атрибут нетранзитивно зависит от первичного ключа. То есть какой-то атрибут может зависеть от неключевого атрибута.

В описанном выше отношении «Учётные записи» отсутствуют транзитивные зависимости, что означает, что данное отношение уже находится в 3НФ.

В отношении «Автомобили в наличии» такие атрибуты как Дата проведения обслуживания, Тип обслуживания, Описание состояния автомобиля, Стоимость обслуживания, Статус обслуживания зависят от неключевого атрибута Код обслуживания. Выполним декомпозицию и выделим ещё одно отношение «Техническое обслуживание» с атрибутами:

(Код обслуживания, VIN-номер, Дата проведения обслуживания, Тип обслуживания, Описание состояния автомобиля, Стоимость обслуживания, Статус обслуживания)

В данном отношении код обслуживания становится первичным ключом, а VIN-номер остаётся внешним ключом для связи с таблицей «Автомобили в наличии», которая также получена после декомпозиции:

(VIN-номер, Марка, Модель, Год выпуска, Мощность двигателя (л.с.), Рабочий объём двигателя, Тип двигателя, Тип трансмиссии, Количество мест, Привод, Стоимость за сутки (руб), Регистрационный номер, Код подразделения)

Из этого отношения также пропали такие атрибуты, как Название изображения, данные изображения, тип изображения. Данные атрибуты транзитивно зависят от неключевого атрибута код изображения. Атрибут, адрес подразделения также зависит от неключевого атрибута Код подразделения. Поэтому выделено ещё два новых отношения «Изображения» и «Подразделения»:

(Код изображения, VIN-номер, Название изображения, данные изображения, тип изображения).

(Код подразделения, Адрес подразделения, Номер телефона подразделения).

Также такие атрибуты, как Формулировка штрафа и Сумма штрафа зависят от неключевого атрибута Код штрафа. Выполним декомпозицию и получим отношение «Штрафы», оставив атрибут Код аренды для связи с таблицей «Аренды»:

(Код штрафа, Код аренды, Формулировка штрафа, Сумма штрафа)

Таблица «Аренды» после декомпозиции:

(Код аренды, Код учетной записи, Номер телефона, VIN-номер, Дата начала аренды, Дата конца аренды, Серия и номер паспорта арендатора, Серия и номер водительского удостоверения арендатора, Общая стоимость аренды, закрыта ли аренда).

В полученных отношениях отсутствуют зависимости между неключевыми атрибутами, следовательно, все отношения находятся в 3НФ.

В результате нормализации получена схема БД, где каждое отношение находится в 3НФ:

«Автомобили в наличии»:

(VIN-номер, Марка, Модель, Год выпуска, Мощность двигателя (л.с.), Рабочий объём двигателя, Тип двигателя, Тип трансмиссии, Количество мест, Привод, Стоимость за сутки (руб), Регистрационный номер, Код подразделения).

«Техническое обслуживание»:

(Код обслуживания, VIN-номер, Дата проведения обслуживания, Тип обслуживания, Описание состояния автомобиля, Стоимость обслуживания, Статус обслуживания).

«Учётные записи»:

(Код учетной записи, Пароль, Электронная почта, Фамилия, Имя, Отчество).

«Аренды»:

(Код аренды, Код учетной записи, Номер телефона, VIN-номер, Дата начала аренды, Дата конца аренды, Серия и номер паспорта арендатора, Серия и номер водительского удостоверения арендатора, Общая стоимость аренды, Закрыта ли аренда).

«Штрафы»:

(Код штрафа, Код аренды, Формулировка штрафа, Сумма штрафа).

«Изображения»:

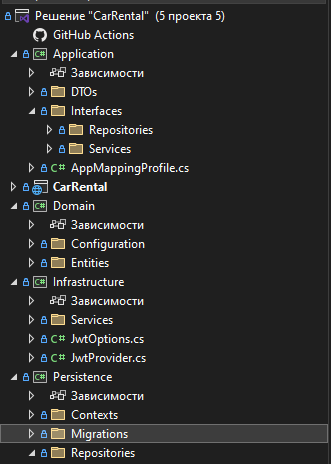
(Код изображения, VIN-номер, Название изображения, данные изображения, тип изображения).

«Подразделения»

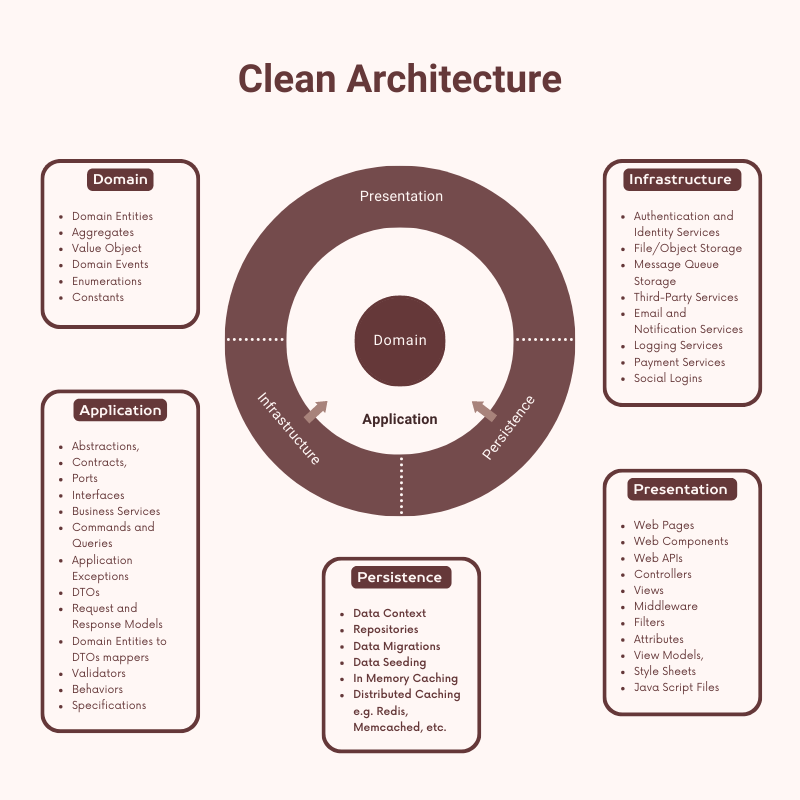
(Код подразделения, Адрес подразделения, Номер телефона подразделения).

* 1. Проектирование архитектуры приложения

При разработке различных компонентов информационной системы важно поддерживать его масштабируемость и тестируемость. Для этого необходимо задуматься над архитектурой приложения. Для данной курсовой работы используется ASP.NET Core. Среди разработчиков на данной платформе распространено понятие «чистой архитектуры». То, как данный вид архитектуры должен примерно выглядеть показано на рисунке 10.



***Рисунок 10 - Архитектура приложения***



***Рисунок 11 - схема «чистой архитектуры»***

Главная идея – разделение приложения на несколько слоёв (проектов). Зависимости проектов должны быть настроены таким образом, чтобы бизнес-логика не зависела от инфраструктуры. Далее будут описаны все слои:

Слой Domain – слой, содержащий все сущности, требуемые системе, и некоторую бизнес-логику. Данный слой не зависит от внешних слоёв.

Слой Application – содержит объекты для передачи данных (DTO), а также интерфейсы, которые должны использоваться внешними сервисами и репозиториями (репозиторий – это паттерн проектирования программного обеспечения, который выступает промежуточным звеном между бизнес-логикой приложения и хранилищем данных, в данном случае с использованием EntityFrameworkCore). Данный слой должен ссылаться только на Domain.

Слой Infrastructure – хранит внешние зависимости, такие как инструменты для кэширования, внешние API, файловые хранилища. Также может содержать реализацию различных сервисов (классы с бизнес-логикой). Данный слой зависит от слоёв Domain и Application.

Слой Persistence – это часть Infrastructure слоя, который отвечает за взаимодействие с базой данных. Данный слой хранит реализацию репозиториев, настройки контекста базы данных для EntityFrameworkCore, миграции и другие вещи, связанные с базой данных. Зависит от слоёв Application и Domain.

Слой Presentation – содержит веб-API, где определены все компоненты middleware, контроллеры и конечные точки (endpoints). Зависит от слоя Application.

Таким образом, придерживаясь данных правил можно получить гибкое, масштабируемое и легко тестируемое приложение.

* 1. Результаты проектирования

В результате проектирования информационной системы были получены следующие результаты:

* Спроектирована и нормализована до 3 нормальной формы база данных, содержащая 7 таблиц.
* Выполнено проектирование архитектуры приложения, которая должна обеспечить гибкость и масштабируемость информационной системы.

Полученные результаты будут использоваться при реализации системы.

1. Реализация системы

На этапе реализации требуется разработать серверную часть приложения, написать все необходимые сервисы и контроллеры, имплементировать интерфейсы. Также важно разработать клиентскую часть с графическим интерфейсом, определить связи между страницами и настроить динамические элементы страниц.

В результате этой главы должно быть получено веб-приложение, обладающее удобным графическим интерфейсом и стабильно и быстро работающим сервером. Серверная часть должна быть написана на ASP.NET Core C# с использованием EntityFrameworkCore для работы с базой данных, для клиентской части должен быть использован JavaScript с библиотекой React, а также HTML и CSS.

Исходные коды программы выложены на сервис для публикации и хранения IT проектов GitHub. Ссылка для серверной части: <https://github.com/auone1328/Car-Rental>. Ссылка для клиентской части: <https://github.com/auone1328/CarRental_UI>.

* 1. Описание классов серверной части

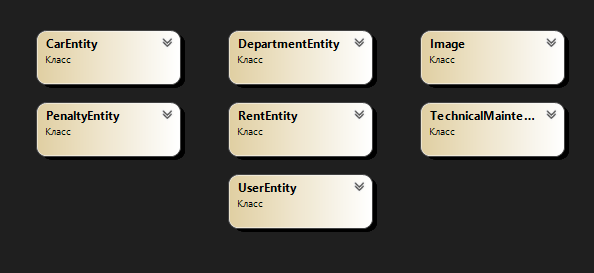
При реализации серверной части приложения были получены следующие классы для различных слоёв:

Классы слоя Application представлены на рисунке 12. В эти классы входят объекты передачи данных (data transfer objects), а также интерфейсы, которые должны быть имплементированы в репозиториях и сервисах из других слоёв:



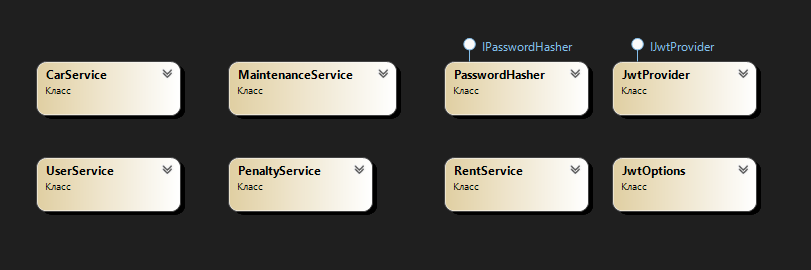
***Рисунок 12 - Диаграмма классов для слоя Application***

Классы слоя Domain представлены на рисунке 13. В эти классы входят объекты сущностей, содержащие те же поля с теми же типами данных, что должны быть в базе данных. Посредством EntityFrameworkCore, миграций и конфигураций данные объекты сущностей будут перенесены в базу данных:



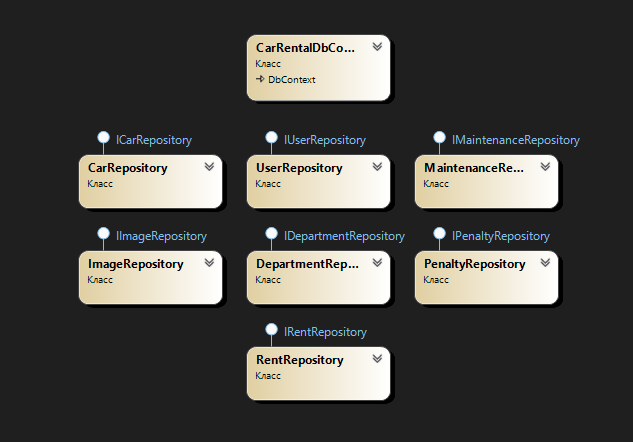
***Рисунок 13 - Диаграмма классов для слоя Domain***

Классы слоя Infrastructure представлены на рисунке 14. В эти классы входят реализация сервисов, которые используются контроллерами:



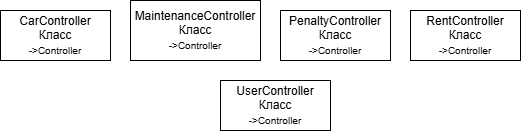
***Рисунок 14 - Диаграмма классов для слоя Infrastructure***

Классы слоя Persistence представлены на рисунке 15. В эти классы входят реализации репозиториев, а также контекст базы данных:



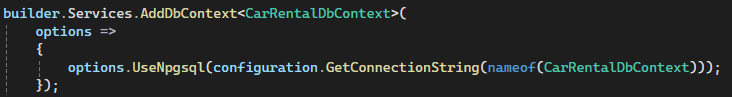
***Рисунок 15 - Диаграмма классов для слоя Persistence***

Классы слоя Presentation представлены на рисунке 15. В эти классы входят реализации репозиториев, а также контекст базы данных:



***Рисунок 16 - Диаграмма классов для слоя Presentation***

Также в этом слое есть класс Program, где описаны все компоненты middleware (фрагменты кода в конвейере приложения, используемые для обработки запросов и ответов). Также здесь регистрируются все сервисы в Dependency Injection Container, который помогает сделать компоненты программы менее связанными между собой (чтобы объекты не зависели от деталей других классов). Здесь же происходит регистрация и настройка контекста базы данных (рисунок 17). Используется провайдер «Npgsql» (так как в качестве СУБД используется PostgreSQL), строка подключения получена из файла «appsettings.json».



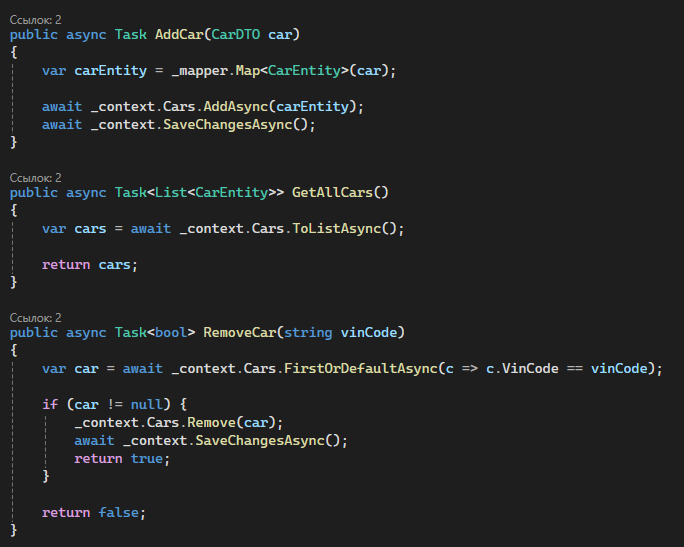
***Рисунок 17 - Добавление контекста базы данных и подключение к ней через строку подключения***

После добавления миграции, все сущности должны быть перенесены в базу данных PostgreSQL. ER-диаграмма полученной базы данных представлена на рисунке 18.

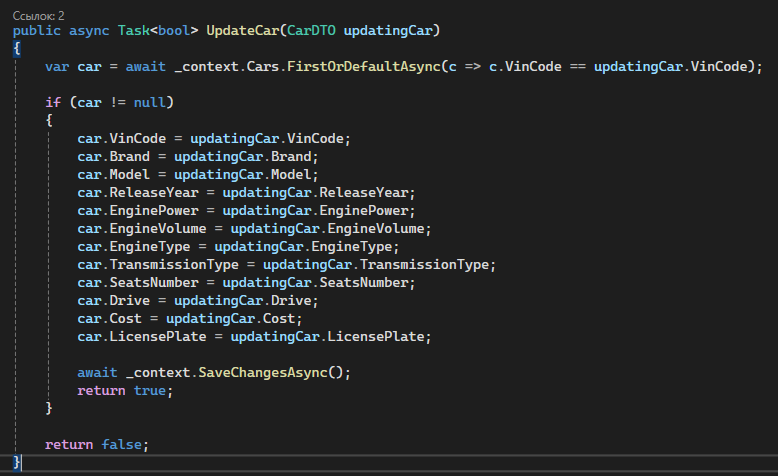


***Рисунок 18 - ER-диаграмма базы данных***

Примеры основных запросов к базе данных представлены на примере «CarRepository», остальные репозитории имеют аналогичный код запросов (рисунок 18 и рисунок 19).



***Рисунок 19 - запросы на добавление, просмотр и удаление к базе данных***



***Рисунок 20 - запрос на обновление к базе данных***

В коде некоторых функций можно заметить функцию Map(). Это функция из плагина «AutoMapper» – это средство сопоставления одного объекта на другой. Оно преобразует входной объект одного типа в выходной объект другого типа. «Маппиннг» настроен в классе AppMappingProfile.

Также важно описать процесс авторизации и аутентификации. В классе Program также регистрируется сервисы по авторизации и аутентификации и подключаются соответствующие компоненты middleware (рисунок 20). Аутентификация и авторизация использует JWT-токен (refresh и access токены). При аутентификации система генерирует access и refresh токены. Access-токен сохраняется на клиенте в cookies, что является безопасным, так как к cookies у JavaScript нет доступа. Access-токен может хранить различную информацию о пользователе, в том числе информацию об его идентичности (объекты Claims.Identity), которая используется для авторизации как на сервере, так и на клиенте. Refresh токен сохраняется в базе данных и используется для аутентификации пользователя в случае истечения access токена, то есть в случае, если пользователь давно не пользовался системой, но не настолько давно, чтобы истёк и Refresh токен (средний срок жизни access-токена – 15 минут, refresh-токена – от 15 до 30 дней). Refresh-токен также сохраняется в cookies. Генерация токенов представлена на рисунке 21. Токены генерируются при входе пользователя в систему. Доступ пользователя к конечной точке определяется с помощью атрибута [Authorize(Roles = “role”)]



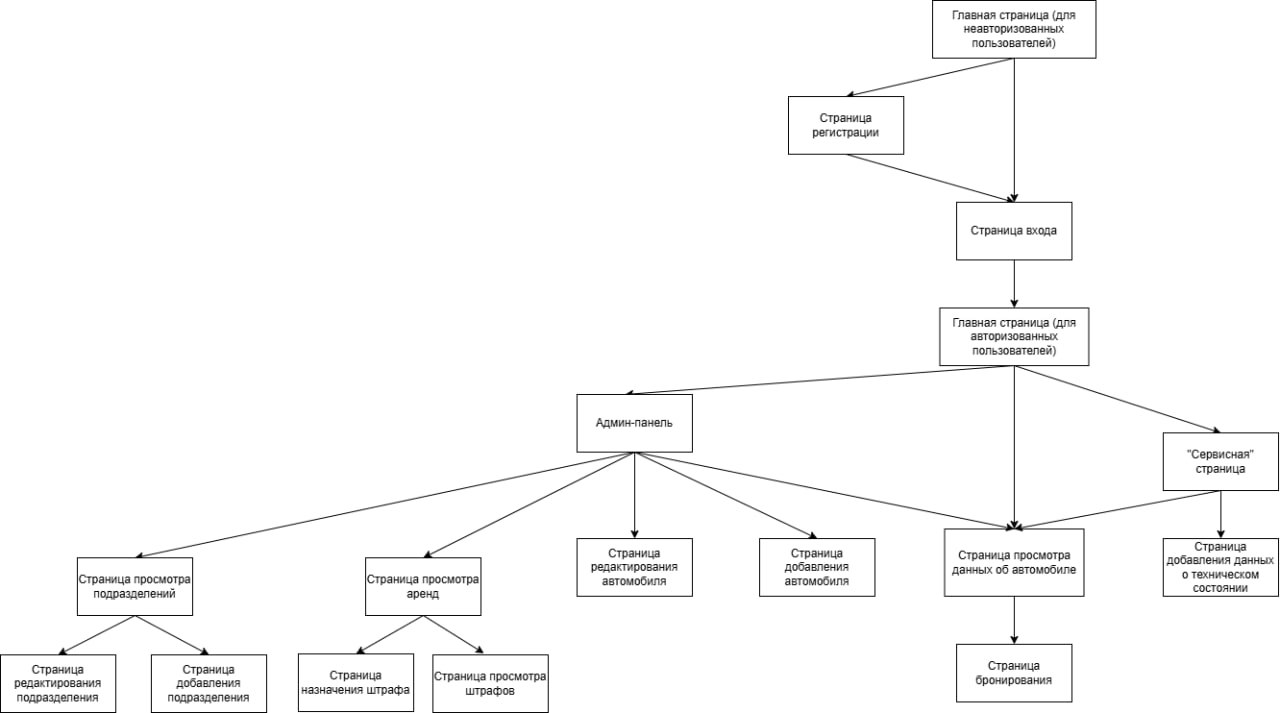
***Рисунок 21 - подключение аутентификации и авторизации***



***Рисунок 22 - Генерация JWT-токенов***

* 1. Схема графического интерфейса

Система имеет следующие страницы с графическим интерфейсом:

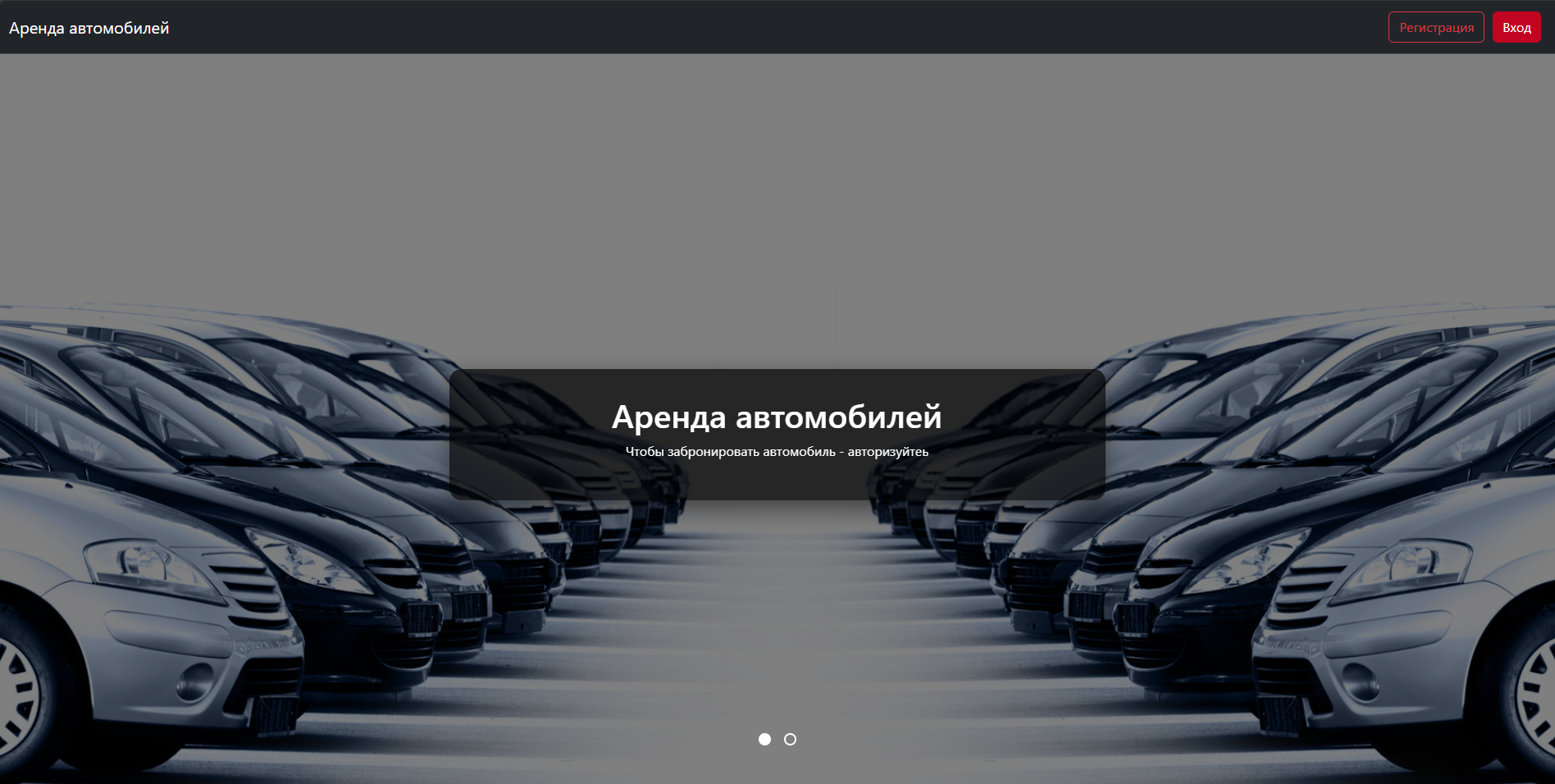


***Рисунок 23 - схема интерфейса информационной системы***

Графический интерфейс данной информационной системы построен на компонентах React, которые представляют собой стилизованную HTML разметку с некоторой динамикой, которую обеспечивает JavaScript. Благодаря динамике некоторые страницы сделаны при использовании одного компонента, например, страница редактирования автомобиля и страница добавления автомобиля. Страницы для этих операций очень похожи, поэтому можно использовать один компонент, но в зависимости от требуемой операции, некоторые части страницы динамически «перерисовываются». Также некоторые страницы могут динамически меняться в зависимости от роли пользователя (администратор, механик, клиент).

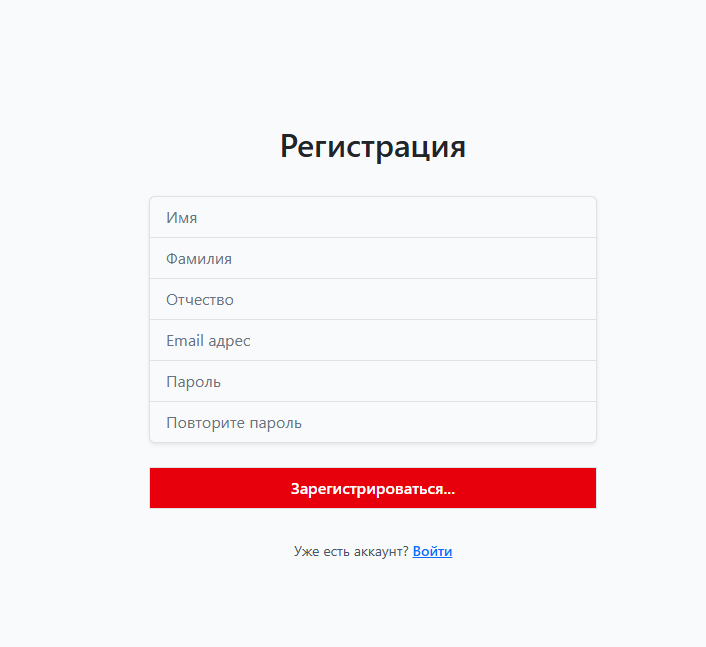
* 1. Характеристики поведения интерфейса

Когда пользователь в первый раз переходит на сайт, то он видит главную страницу (рисунок 23) с кнопками «регистрация» и «вход», которые ведут на соответствующие страницы.



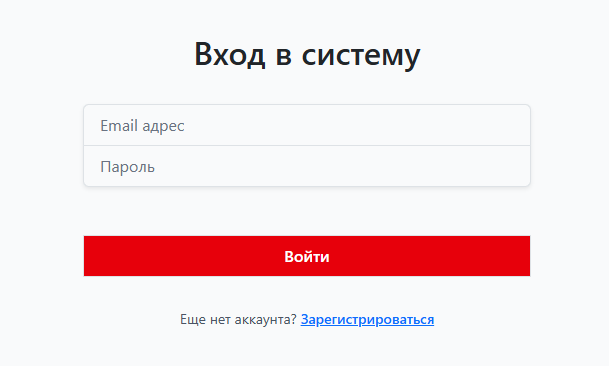
***Рисунок 24 - Главная страница***

При нажатии на кнопку «Регистрация» открывается, соответственно, форма регистрации (рисунок 24), в которой можно перейти на страницу входа, если аккаунт уже создан пользователем. Если аккаунт ещё не создан, то после нажатия на кнопку «Зарегистрироваться» в базу данных сохраняется новый пользователь с ролью «клиент» (если пользователь с такой электронной почтой ещё не зарегистрирован), после чего пользователь перенаправляется на страницу «Входа».



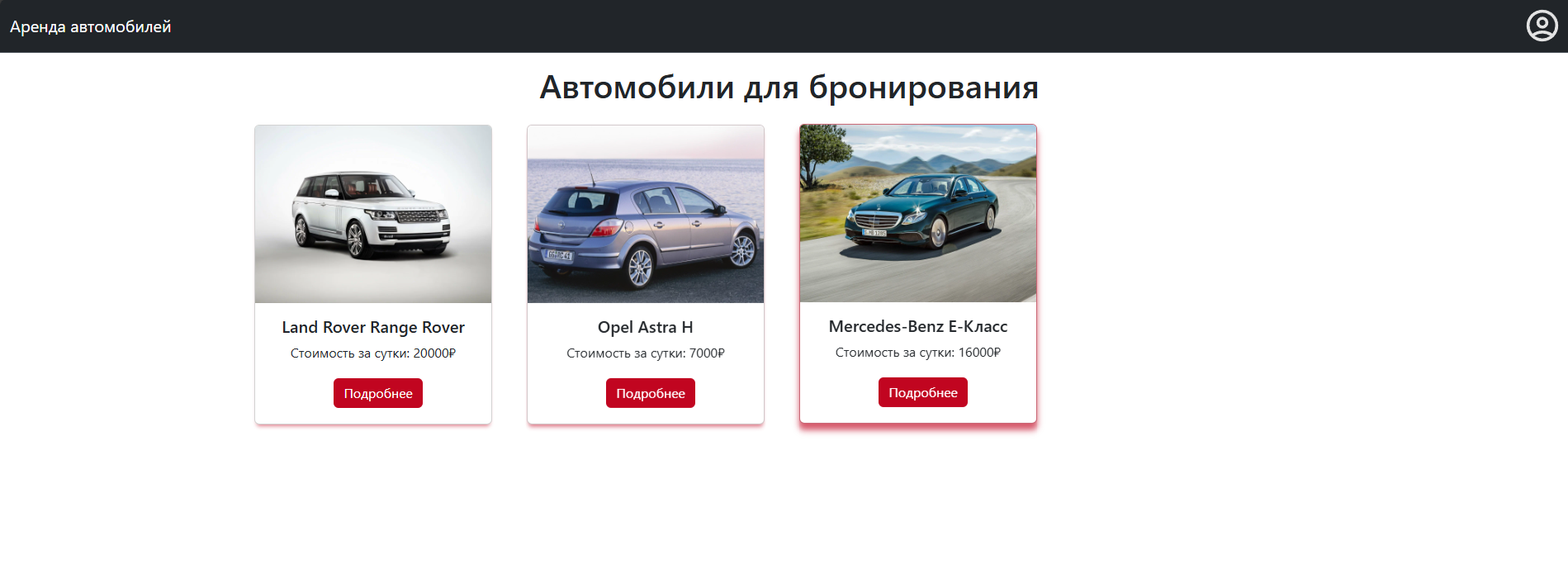
***Рисунок 25 - страница регистрации***

При нажатии на главной странице кнопки «Вход» или при нажатии на ссылку «Войти» на странице регистрации пользователь попадёт на страницу входа (рисунок 25). Оттуда также можно попасть на страницу регистрации по нажатию на ссылку «Зарегистрироваться». После заполнения формы входа и нажатия кнопки «Войти», при успешной аутентификации, пользователь будет направлен на главную страницу.

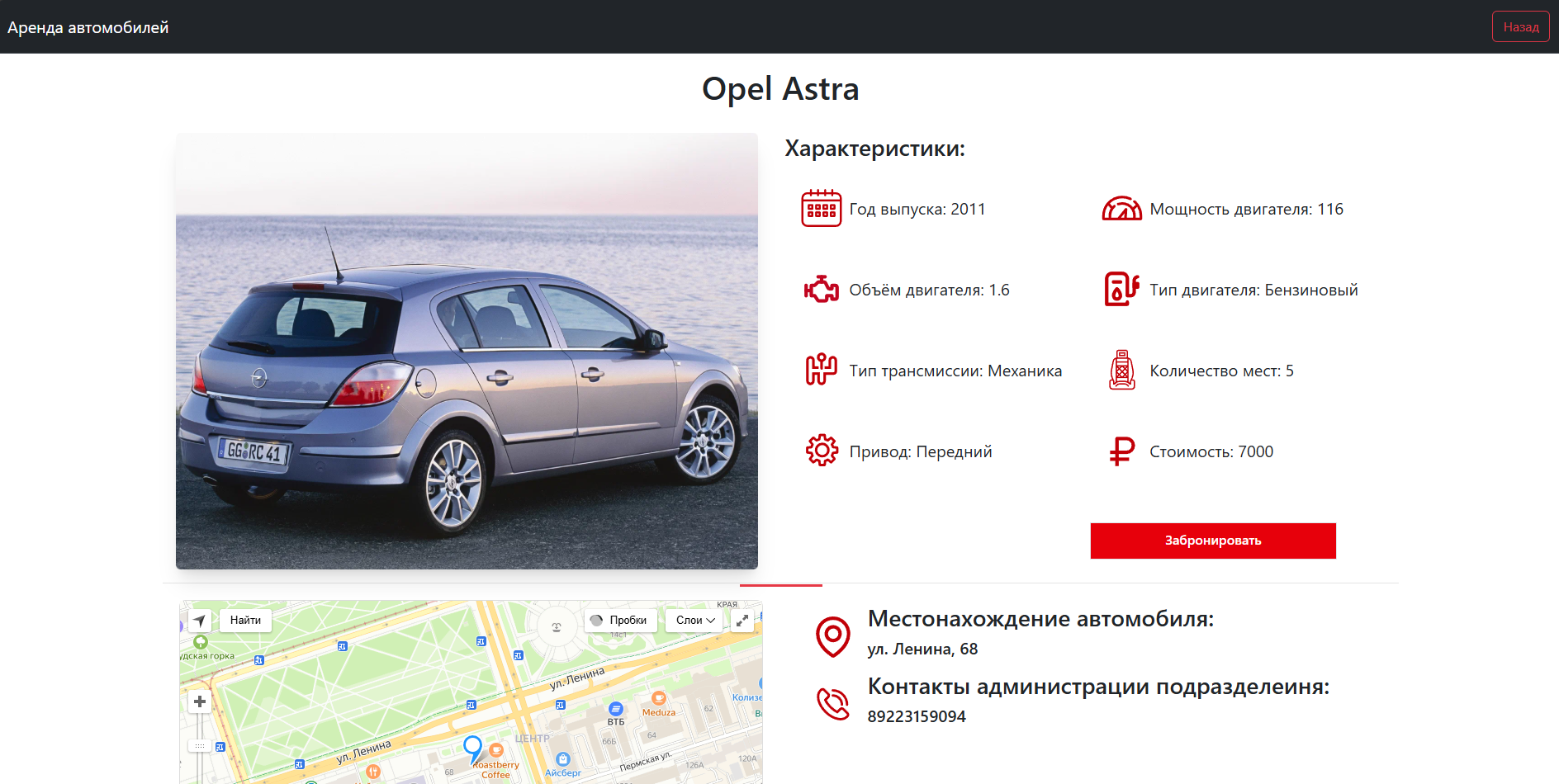


***Рисунок 26 - Страница входа***

После аутентификации всем пользователям открывается главная страница (рисунок 26), но теперь там вместо баннера – карточки с автомобилями, по которым можно нажать, после чего клиент попадет на страницу с информацией об автомобиле (рисунок 28).

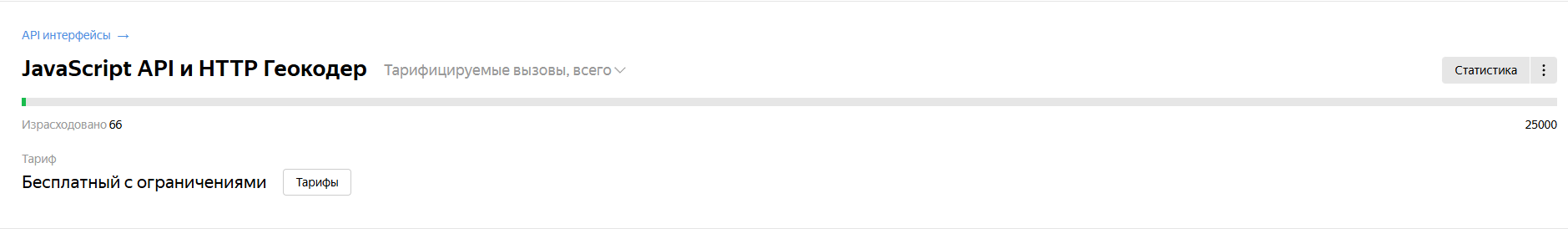


***Рисунок 27 - Главная страница после авторизации***

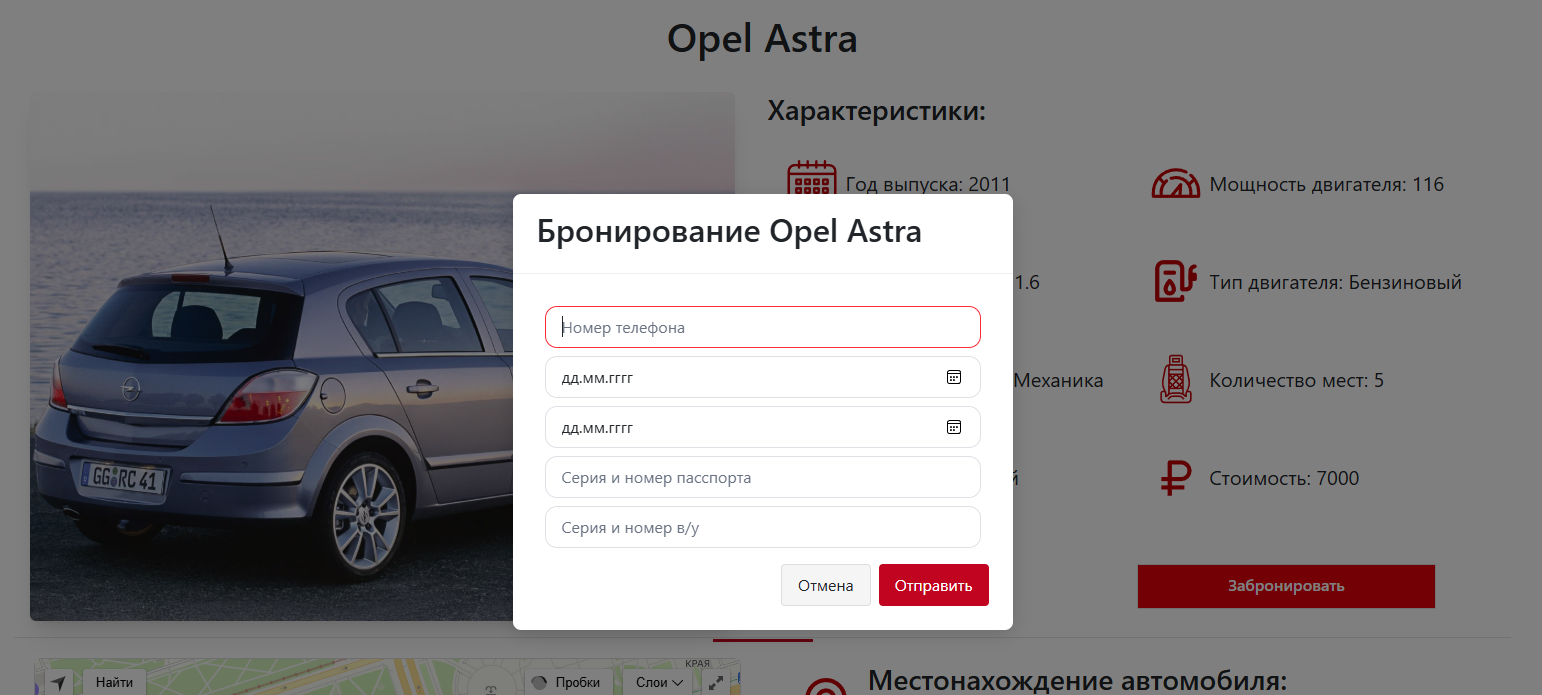


***Рисунок 28 - страница просмотра данных об автомобиле***

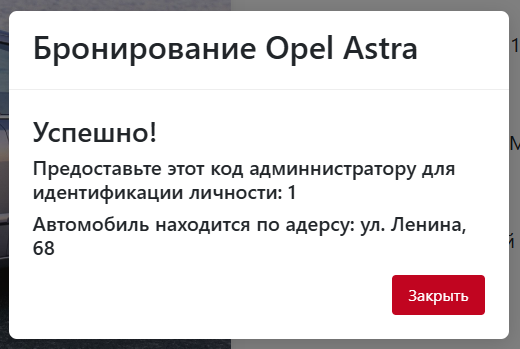
На странице просмотра данных об автомобиле отображается вся информация, которую должен был занести в БД администратор проката (описание интерфейса админ-панели будет описано далее), в том числе есть карта с местоположением подразделения, в котором сейчас находится данный автомобиль. Карта предоставлена Яндекс API (рисунок 29). По нажатию кнопки «Забронировать» будет открыто модальное окно с формой, после заполнения которой будет добавлена запись с арендой в базу данных, а пользователь получит код аренды и адрес, по которому находится автомобиль (рисунок 30 и рисунок 31 соответственно).



***Рисунок 29 - API для отображения Яндекс Карты***

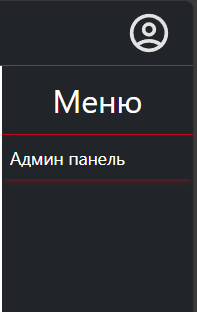


***Рисунок 30 - Окно бронирования***



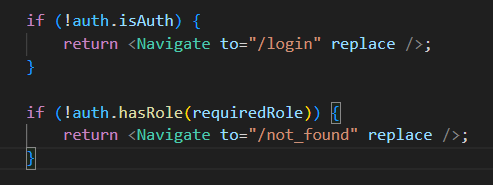
***Рисунок 31 - Окно бронирования после ввода данных***

Рассмотрим админ-панель. Если пользователь с ролью администратор нажмёт на кнопку профиля в правом верхнем углу приложения, то он откроет меню (рисунок 32).



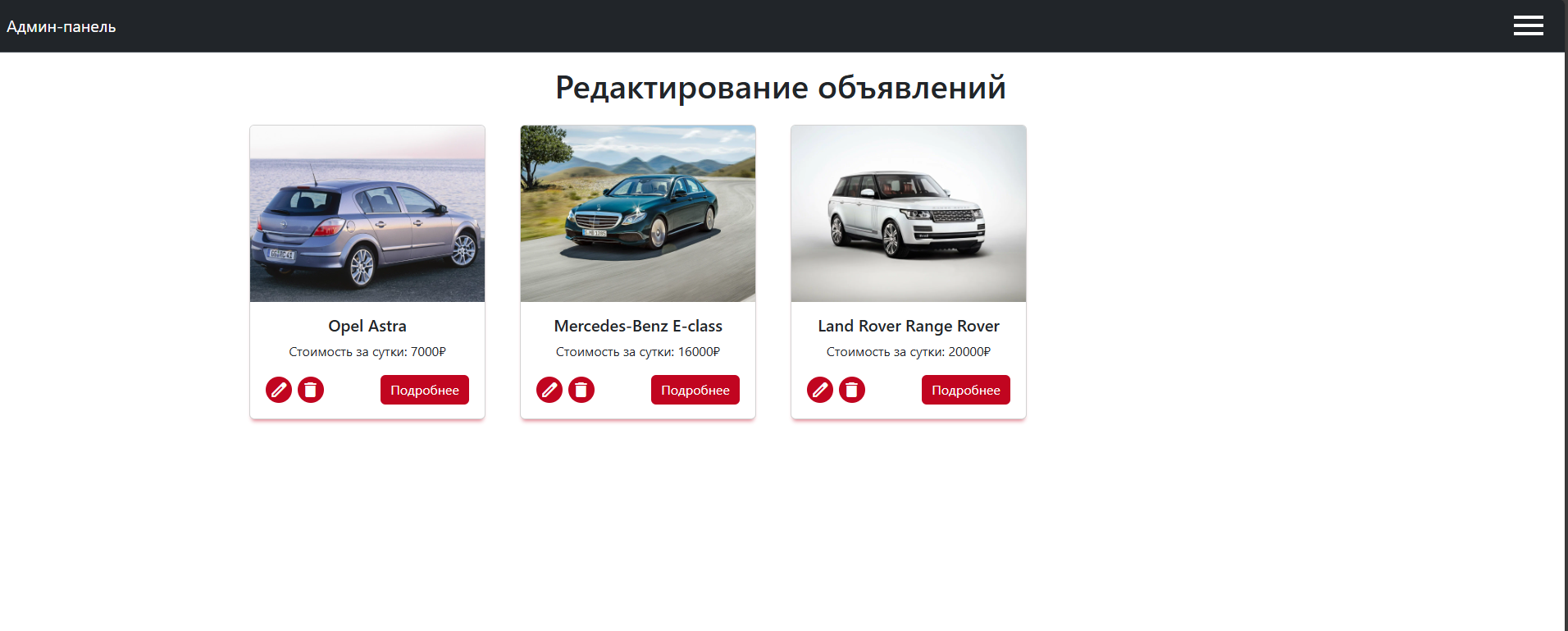
***Рисунок 32 - Меню администратора***

Кнопка «Админ панель» в меню отображается только у пользователя с ролью «Администратор», при нажатии на неё пользователь направляется на защищенную страницу. Для обеспечения приватности маршрута, по которому переходит пользователь при нажатии на кнопку, компонент, к которому ведёт конечная точка, оборачивается в React -компонент ProtectedRoute, часть кода которого изображена на ***рисунке 33***. Роль пользователя хранится в закодированном виде в payload JWT-токена. Функция hasRole() декодирует токен и получает роль пользователя, которая проверяется в ProtectedRoute. Если пользователь не имеет доступа к ресурсу, то он будет направлен на страницу с кодом 404 «Страница не найдена». Данное решение ограничивает доступ к приватным ресурсам обычным пользователям.



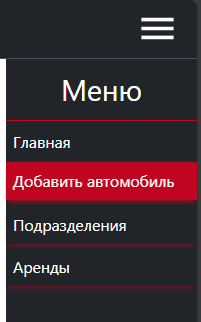
***Рисунок 33 - Защита приватных маршрутов***

В админ панели также сначала выведены карточки со всеми автомобилями, однако в отличие от того, что видят обычные пользователи, у администратора есть кнопки удаления и редактирования объявления (рисунок 34).

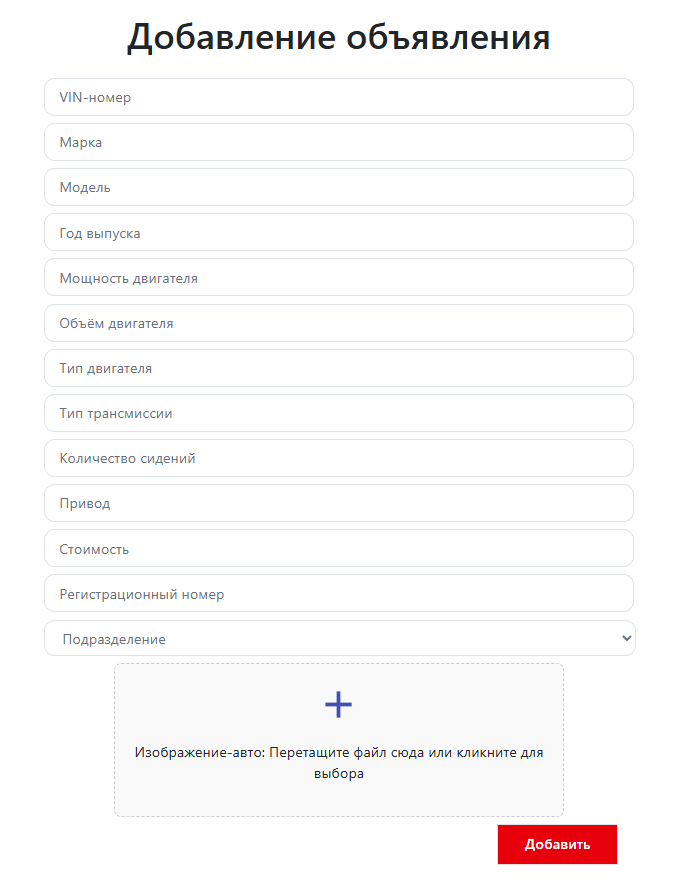


***Рисунок 34 - Админ-панель***

В меню админ панели находятся пункты, обозначенные на рисунке 35. После нажатия на кнопку «Добавить автомобиль» администратор будет направлен на страницу добавления автомобиля (рисунок 36). После заполнения данных и нажатия на кнопку «Добавить», данные об автомобиле и изображении добавятся в базу данных, а администратор будет направлен обратно в админ-панель, где также будет отображаться новое объявление.



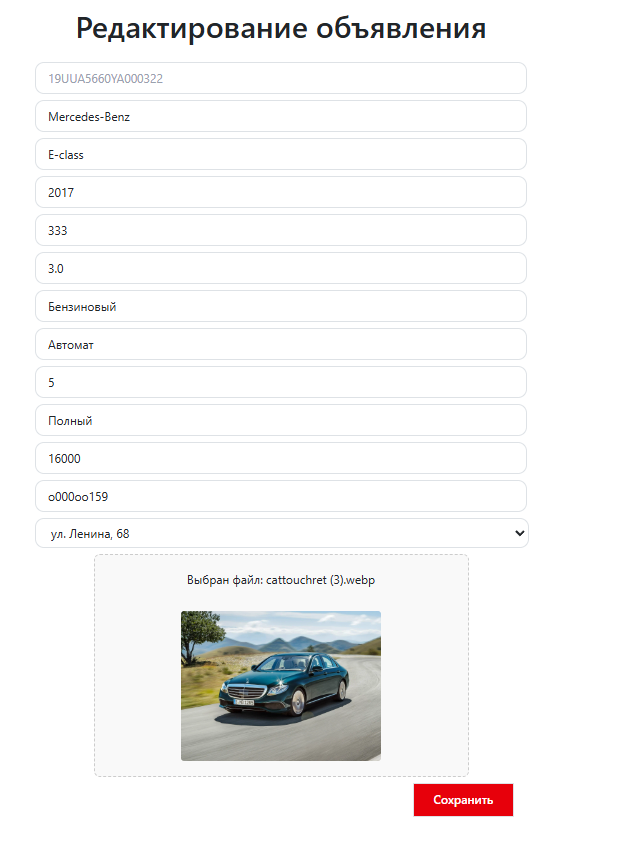
***Рисунок 35 - Меню админ-панели***



***Рисунок 36 - Страница добавления автомобиля***

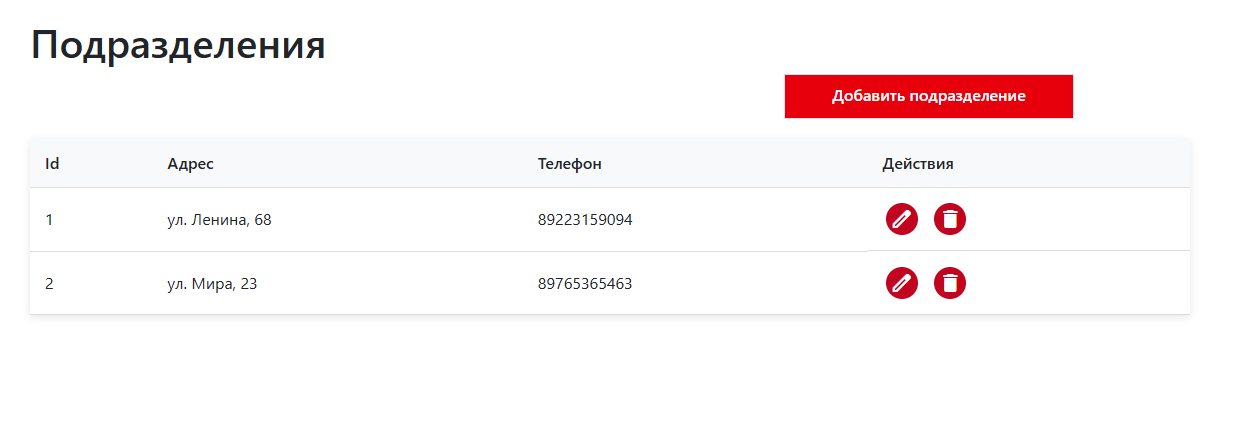
При нажатии на кнопку удаления из базы данных каскадно удаляется вся информация по автомобилю (в том числе связанные с ним изображения, аренды, обслуживания). Объявление также перестаёт отображаться у всех пользователей.

После рассмотрения добавления автомобиля, уместно рассмотреть редактирование автомобиля, так как данные действия используют один и тот же React-компонент с минимальными динамическими изменениями, такими как заполненные по умолчанию поля формы и изменённое событие нажатия на кнопку. После нажатии на кнопку редактирования открывается страница редактирования данных об автомобиле (рисунок 37). После изменения данных и нажатия на кнопку «Сохранить» данные сохраняются в базе данных, на клиенте также отображаются обновлённые данные.



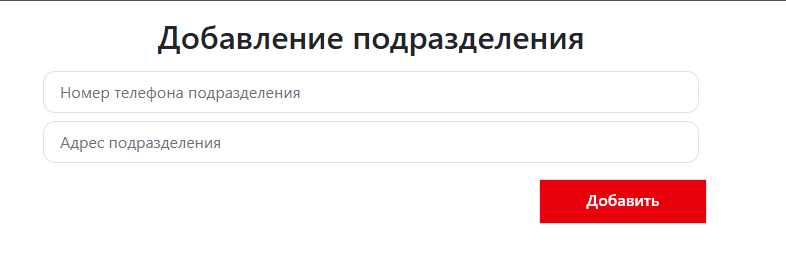
***Рисунок 37 - Страница редактирования данных об автомобиле***

Если вернуться в админ-панель и в меню нажать на кнопку «Подразделения», то администратор будет направлен на страницу просмотра данных о подразделениях, где он сможет редактировать или удалить определённую запись базы данных, а также добавить новую (рисунок 38).

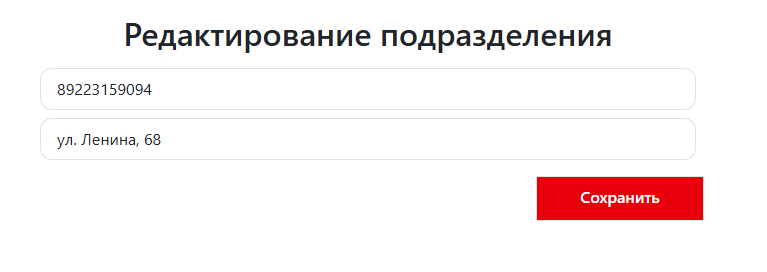


***Рисунок 38 - Страница просмотра подразделений***

После нажатия на кнопку «Добавить подразделение», пользователь попадёт на страницу добавления данных о подразделении. После заполнения формы и нажатия на кнопку «Добавить», администратор будет направлен обратно на страницу просмотра данных о подразделениях, а в базу данных добавятся новые данные о подразделении.

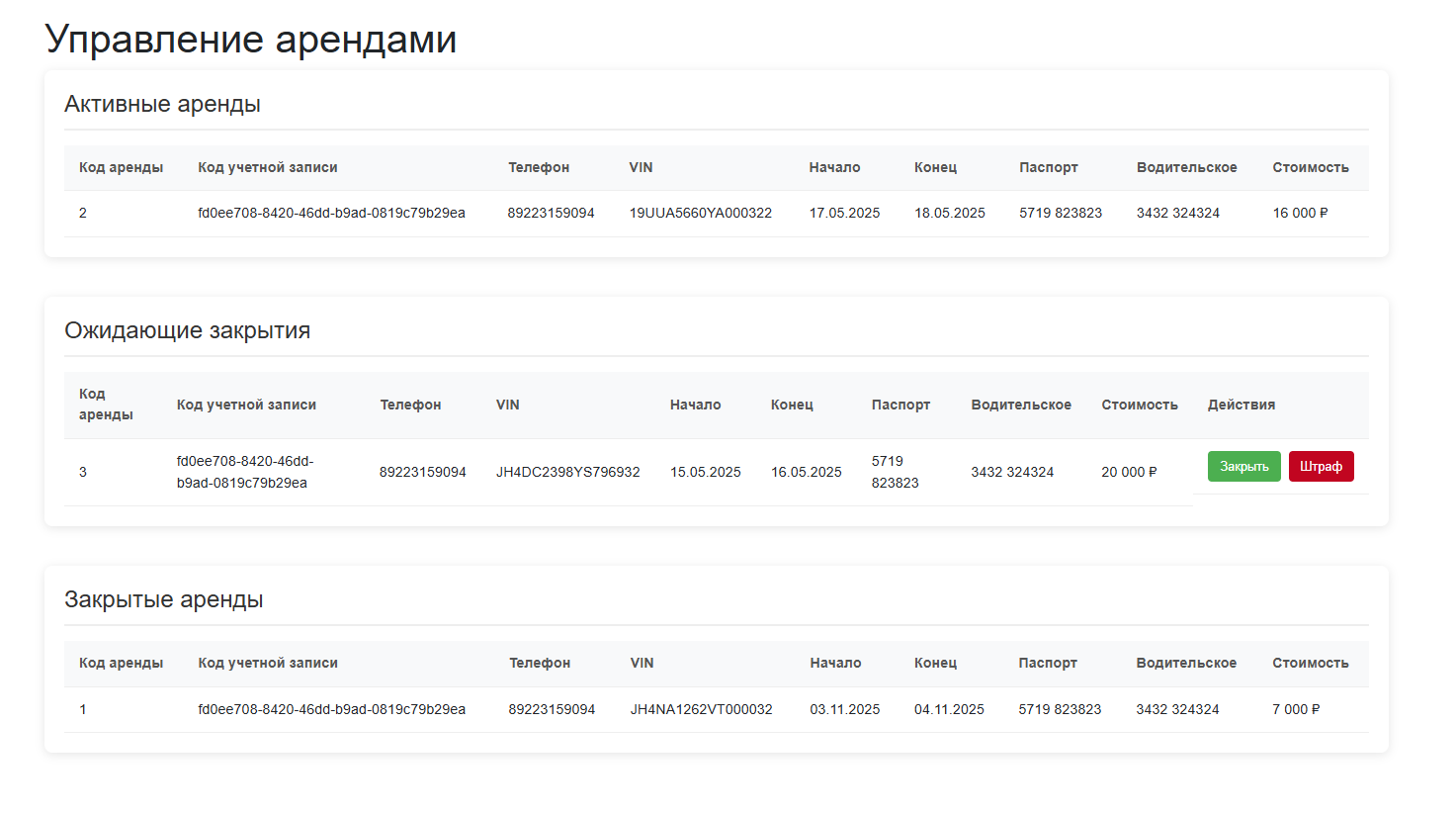
******

***Рисунок 39 - Страница добавления подразделения***

При выборе функции редактирования подразделения система перенаправляет администратора на специализированную форму, идентичную интерфейсу создания нового подразделения (рисунок 40). После внесения необходимых корректировок и подтверждения действий посредством кнопки сохранения, обновленная информация автоматически фиксируется в базе данных, после чего происходит возврат к основному списку подразделений. Функция удаления обеспечивает комплексное устранение всех связанных данных о подразделении как в базе данных (с соблюдением каскадных ограничений), так и в пользовательском интерфейсе, что немедленно отражается на страннице.

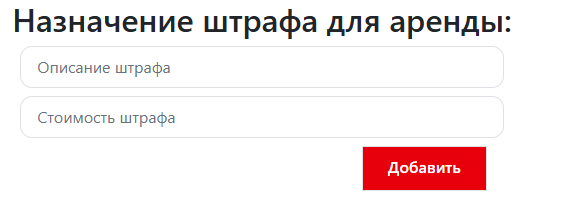
***Рисунок 40 - Страница редактирования данных о подразделении***

Если вернуться в админ-панель и в меню нажать на кнопку «Аренды», то администратор будет направлен на страницу просмотра данных об арендах, где он сможет просмотреть последние завершенные и незавершенные аренды, на которых есть две кнопки: закрыть аренду или назначить штраф.



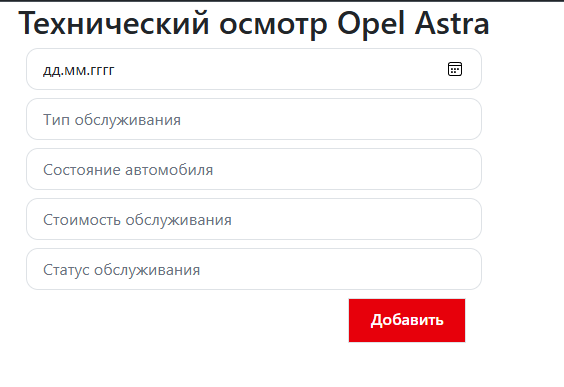
***Рисунок 41 - Страница просмотра аренд***

После нажатия на кнопку «Штраф» будет открыта форма добавления штрафа (рисунок 42).

******

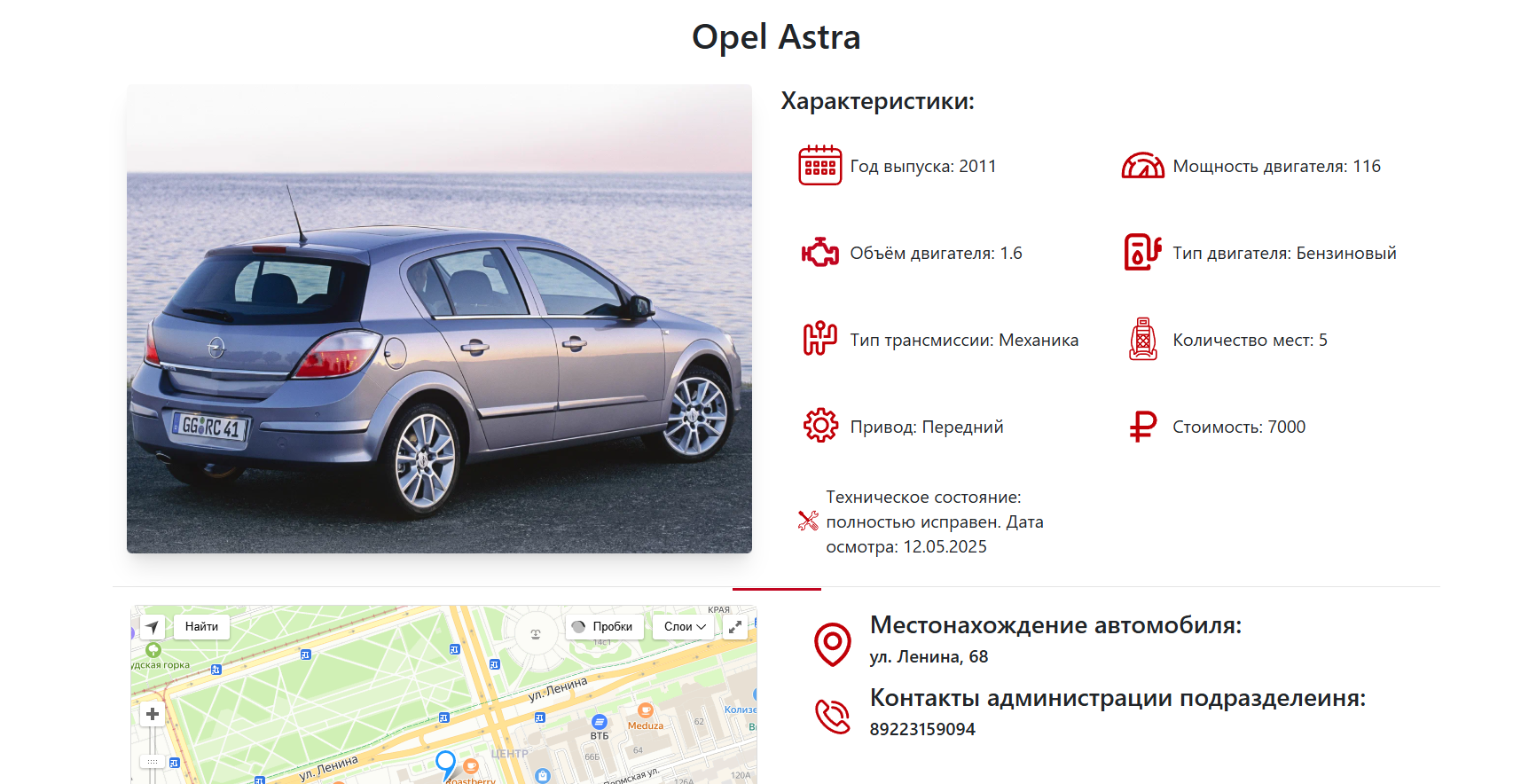
***Рисунок 42 - Форма добавления штрафа***

Что касается роли механика, для него в меню будет кнопка «Сервис», при переходе по которой он также попадёт на страницу с отображенными карточками автомобилей, однако, для него доступна кнопка «Добавить осмотр» по нажатию на которую он попадёт на страницу добавления технического осмотра (рисунок 43).

******

***Рисунок 43 - Страница добавления технического осмотра***

После заполнения формы, на странице информации об автомобиле администратора будет выведена информация о техническом состоянии автомобиля, у клиента же такой информации отображаться не будет (рисунок 44).

******

***Рисунок 44 - Страница просмотра автомобиля для администратора***

Последняя функция информационной системы – просмотр аренд клиентом. В случае если аренды и штрафы есть у клиента на аккаунте, тогда он увидит таблицу с данными по ним. Иначе, будет показано текстовое сообщение об их отсутствии (рисунок 45).

******

***Рисунок 45 - Страница просмотра аренд (для клиента)***

* 1. Результаты реализации системы

В процессе реализации системы были получены следующие результаты:

1. Создано веб-приложение, использующее 40 классов различного назначения в серверной части.
2. Создано 17 страниц интерфейса для просмотра, добавления и редактирования данных.
3. Проведено компонентное и системное тестирование приложения.

ЗАКЛЮЧЕНИЕ

В результате анализа предметной области выявлены проблемы, с которыми может столкнуться бизнес по прокату автомобилей, главной из которых является сбор, хранение и поиск важной информации. Проблемы возникают ввиду отсутствия удобного и быстрого инструмента для работы с данными.

Анализ показал, что существующие решения не могут предложить инструментов для полноценного учёта данных, чтобы при этом тот же ресурс мог использоваться и для сбора данных клиентов.

Результаты анализа требований оформлены в виде технического задания (Приложение A).

В ходе проектирования разработана физическая модель базы данных, которая была приведена к 3 нормальной форме. Также была разработана архитектура приложения, которая должна обеспечить гибкость, хорошую тестируемость и масштабируемость системы.

При реализации прототипа создано 17 страниц графического интерфейса, разработаны 7 классов для сущностей, 7 классов-репозиториев, 1 класс контекста базы данных, 9 классов-сервисов, 7 объектов передачи информации (DTO), 9 интерфейсов.

Все описанные сценарии реализованы и протестированы с помощью «Чёрного ящика».

Полученные результаты могут полезны для учебных целей, но также могут быть использованы при разработке реальной информационной системы для какого-либо бизнеса с подобными требованиями.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. ГОСТ 19.201–78. Единая система программной документации. Техническое задание. Требования к содержанию и оформлению : издание официальное : межгосударственный стандарт : утвержден и введен в действие Постановлением Государственного комитета СССР по стандартизации от 18 декабря 1978 г. № 3351 с 01.01.80. – М.: Стандартинформ, 2010.
2. Руководство по созданию программного обеспечения с чистой архитектурой. [Электронный ресурс] – Режим доступа: <https://www.ezzylearning.net/tutorial/a-guide-for-building-software-with-clean-architecture>.
3. Гражданский кодекс Российской Федерации. [Электронный ресурс] – режим доступа: <http://kodeks.systecs.ru/gk_rf/gk_glava34/>.
4. Гражданский кодекс. Аренда транспортных средств. [Электронный ресурс] – Режим доступа: <https://base.garant.ru/10164072/08b7d927fd863bb6a25ed3796d6629ee/>.
5. Интернет-сервис для размещения объявлений о товарах «Авито». [Электронный ресурс] – Режим доступа: <https://www.avito.ru/>.
6. DACARTUR – аренда автомобилей в Перми. [Электронный ресурс] – Режим доступа: <https://perm.dacartur.com/>.
7. Руководство по ASP.NET Core 8. [Электронный ресурс] – Режим доступа: <https://metanit.com/sharp/aspnet6/>.
8. Руководство по Entity Framework Core 9. [Электронный ресурс] – Режим доступа: <https://metanit.com/sharp/efcore/>.
9. Руководство по React. [Электронный ресурс] – Режим доступа: <https://metanit.com/web/react/>.