|  |  |  |
| --- | --- | --- |
| лого для документов 2022 | ***Федеральное агентство по рыболовству***  ***Федеральное государственное бюджетное образовательное***  ***учреждение высшего образования***  ***«Астраханский государственный технический университет»***  **Система менеджмента качества в области образования, воспитания, науки и инноваций сертифицирована**  **ООО «ДКС РУС» по международному стандарту ISO 9001:2015** | |
| Институт информационных технологий и коммуникаций  Направление подготовки: 09.03.01 Информатика и вычислительная техника  Профиль: Автоматизированные системы обработки информации и управления  Кафедра «Автоматизированные системы обработки информации и управления» | | |
| **КУРСОВОЙ ПРОЕКТ**  **Информационная система «Междугородные пассажирские перевозки»**  по дисциплине «СУБД PostgreSQL» | | |
| Допущен к защите:  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024г.  Руководитель:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Оценка, полученная на защите: «\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_»  Дата защиты: \_\_\_\_\_\_\_\_\_\_\_\_\_ | | Проект выполнен  обучающимся группы ДИНРБ-31 Кузургалиевым Р.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Руководитель  ст. преп. Куркурин Н.Д.  ст. преп. Мамлеева А.Р. |
| Члены комиссии:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Куркурин Н.Д.\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Мамлеева А.Р.\_\_\_\_ | |  |

**Астрахань – 2024**

**ФЕДЕРАЛЬНОЕ АГЕНТСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ ПО РЫБОЛОВСТВУ**

**АСТРАХАНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

|  |  |
| --- | --- |
| **УТВЕРЖДАЮ** | Кафедра «Автоматизированные системы  обработки информации и управления» |
| Заведующий кафедрой  д.т.н., профессор  Т.В.Хоменко\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

**ЗАДАНИЕ**

**на выполнение курсового проекта**

Обучающийся ***Кузургалиев Радмир Алексеевич***

Группа ***ДИНРБ-31***

Дисциплина ***СУБД PostgreSQL***

Тема ***Информационная система «Междугородные пассажирские перевозки»***

Дата получения задания «\_12\_»\_\_\_февраля\_\_\_\_\_\_2024г.  
Срок представления обучающимся КП на кафедру «\_28\_\_»\_\_\_мая \_\_\_ \_\_\_\_\_\_\_2024г.

Руководитель ***ст. преподаватель***\_\_\_\_\_\_\_\_\_ ***Куркурин Н.Д.*** «\_12\_»\_\_\_февраля\_\_\_\_\_\_2024г.

*должность, степень, звание, подпись* *ФИО*

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ***Кузургалиев Р.А.*** «\_12\_»\_\_\_февраля\_\_\_\_\_2024г.

*подпись ФИО*

**Задачи**

|  |  |
| --- | --- |
| 1 | ***Разработка базы данных в СУБД PostrgeSQL.*** |
|  | ***- приведение базы данных к третьей нормальной форме*** |
|  | ***- обеспечение целостности данных и ссылочной целостности*** |
|  | ***- обеспечение безопасности базы данных***  ***- план обслуживания и резервного копирования базы данных*** |
| 2 | ***Разработка программного обеспечения, позволяющего:*** |
|  | ***- регистрировать нового пользователя в системе;*** |
|  | ***- осуществлять запись в журнал учета;*** |
|  | ***- осуществлять поиск информации в базе данных по различным критериям;*** |
|  | ***- осуществлять изменение и удаление информации базы данных;*** |
| 3 | ***Тестирование программного обеспечения.*** |

**Список рекомендуемой литературы**

|  |
| --- |
| 1. Саймон Ригс. Администрирование PostgreSQL 9. Книга рецептов. М.:ДМК Пресс, 2015. |
| 1. Моргунов Е.П. Основы языка SQL: учеб. пособие.- СПб.:БХВ-Петербург, 2018-336 с.:ил. 2. PostgreSQL. Документация [Электронный документ]. http://postgresql.ru.net/docs.html |

|  |  |
| --- | --- |
| **УТВЕРЖДАЮ** | К заданию на курсовой проектпо дисциплине  «СУБД PostgreSQL» |
| Заведующий кафедрой  д.т.н., профессор  Т.В.Хоменко\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

**КАЛЕНДАРНЫЙ ГРАФИК**

выполнения курсового проекта

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Разделы, темы и их содержание, графический материал | Дата сдачи | Объем, % |
| 1 | Утверждение темы курсового проекта | 12.02.2024 | 1 |
| 2 | Изучение предметной области.  Описание технологии обработки информации в системе.  Обзор аналогичных систем.  Разработка диаграммы вариантов использования. Определение цели и назначения системы. | 04.03.2024 | 10 |
|  | Инфологическое проектирование базы данных.  Приведение базы данных к третьей нормальной форме. | 25.03.2024 | 20 |
| 4 | Даталогическое проектирование базы данных в PostgreSQL.  Обеспечение целостности данных. | 15.04.2024 | 30 |
| 5 | Создание пользовательских функций. | 29.04.2024 | 40 |
| 6 | Разработка программного обеспечения.  Тестирование. | 23.05.2024 | 50 |
| 7 | Оформление пояснительной записки и презентации. ***Предоставление пояснительной записки на проверку***. | 27.05.2024-  28.05.2024 | 55 |
| 8 | ***Предзащита: презентация курсового проекта, демонстрация базы данных и программного продукта.*** | 29.05.2024-  31.05.2024 | 59 |
| 9 | ***Защита курсового проекта (по графику)*** | 03.06.2024-08.06.2024 | 60-100 |

С графиком ознакомлен «\_12\_»\_\_\_февраля\_\_\_\_\_\_2024г.

**Кузургалиев Р.А.** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, обучающийся группы ДИНРБ-31

*(фамилия, инициалы, подпись*)

График курсового проектирования выполнен без отклонений / с незначительными отклонениями / со значительными отклонениями

*нужное подчеркнуть*

Руководитель курсового проекта\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ст. преподаватель **Куркурин Н.Д**

**СОДЕРЖАНИЕ**

[Введение 7](#_Toc157267395)

[1 Технический проект 8](#_Toc157267396)

[1.1 Описание предметной области 8](#_Toc157267397)

[1.2 Макеты входных данных 11](#_Toc157267403)

[1.3 Макеты выходных данных 12](#_Toc157267404)

[1.4 Постановка задачи 12](#_Toc157267405)

[1.5 Цель создания автоматизированной (информационной) системы. 14](#_Toc157267406)

[1.6 Назначение автоматизированной (информационной) системы. 15](#_Toc157267407)

[1.7 Обзор аналогов. 15](#_Toc157267408)

[1.7.1 Аналог №1. Biletion Office. 16](#_Toc157267409)

[1.7.2. Аналог №2. 1С: Управление транспортом. 17](#_Toc157267410)

[1.7.3. Аналог №3. Busfor Manager. 17](#_Toc157267411)

[1.7.4. Сравнительная таблица аналогов. 18](#_Toc157267412)

[1.7.5. Достоинства и недостатки аналогов. 18](#_Toc157267413)

[1.8. Технология обработки информации 19](#_Toc157267414)

[1.9. Требования к техническому и программному обеспечению 20](#_Toc157267415)

[2 Рабочий проект 22](#_Toc157267416)

[2.1. Обоснование выбора решения. 22](#_Toc157267417)

[2.2. Дизайн основных интерфейсов. 25](#_Toc157267418)

[Заключение 27](#_Toc157267419)

[Список использованных источников 28](#_Toc157267420)

[Приложение. Функциональные требования к систем**е** 29](#_Toc157267421)

# ВВЕДЕНИЕ

Междугородные пассажирские перевозки являются важной составляющей транспортной системы страны. Они обеспечивают связь между различными городами и регионами, позволяют людям перемещаться на большие расстояния и осуществлять командировки, туристические поездки и другие виды путешествий. В современном мире междугородные пассажирские перевозки осуществляются различными видами транспорта, такими как автобусы, поезда, самолеты и другие. Однако, несмотря на широкое использование междугородных перевозок, существующие автоматизированные информационные системы учёта пассажирооборота имеют свои недостатки:

1. **Недостаточная интеграция:**

* Системы учета пассажирооборота разных видов транспорта могут быть недостаточно интегрированы между собой. Это может создавать сложности в отслеживании перемещения пассажиров между различными видами транспорта.

1. **Отсутствие единого стандарта:**

* Отсутствие единого стандарта для систем учета может затруднять сбор и обмен данными между различными перевозчиками. Это может усложнять анализ и оптимизацию пассажирских потоков.

1. **Ограниченная доступность информации:**

* Некоторые системы могут быть ограничены в предоставлении информации о расписаниях, билетах и доступности мест. Это может затруднять пассажирам получение необходимой информации для принятия решений.

1. **Безопасность данных:**

* Защита личных данных пассажиров и обеспечение безопасности системы от кибератак могут быть вызовами для автоматизированных информационных систем.

1. **Неэффективное управление пассажирскими потоками:**

* Недостаточное управление пассажирскими потоками может привести к перегруженности транспортных средств, задержкам и неудовлетворенности пассажиров.

Решение этих проблем может потребовать разработки более интегрированных, стандартизированных и безопасных информационных систем, способных эффективно управлять пассажирооборотом в междугородных перевозках.

# 1 ТЕХНИЧЕСКИЙ ПРОЕКТ

## Описание предметной области

Общественный транспорт рассматривается профессиональными специалистами как особо важная логистическая, относительно самостоятельная и само совершенствующаяся система, являющаяся в настоящее время составной частью региональной, национальной и междугородной логистики, развивающаяся в соответствии с основными этапами эволюции логистики, цепей поставок, также требующая изменений и логистических инноваций.

Развитие общественного междугородного транспорта является актуальной задачей современности, поскольку он непосредственно или косвенно влияет на эффективность экономики страны и реализацию социально-экономических функций. Система междугородного транспорта играет важнейшую роль в жизни регионов страны, в решении спектров вопросов, связанных с проблемами обеспечения комфортабельного, безопасного, качественного транспортного обслуживания, не защищенных слоев населения страны, не имеющих собственных транспортных средств. Успешный рост и развитие регионов невозможен без совершенствования инфраструктуры и коммуникаций городского пассажирского транспорта.

Говоря об общественном транспорте, как логистической системе в инфраструктуре регионов, необходимо разобраться с некоторыми терминами. Развитие современных того или иного региона сопровождается увеличением населения в сельской местности и городах, переселением в новые районы, что влечет за собой удаление жилых районов от мест работы. Вследствие этого возникает необходимость общественного междугородного транспорта в жизни страны. В научной литературе встречается несколько определений междугородного пассажирского транспорта:

* вид транспорта, выполняющий регулярные перевозки пассажиров по установленным и фиксированным маршрутам, рейсам, известным населению;
* многофункциональная транспортная система, которая объединяет различные виды транспорта и осуществляет движение по территории города и ближайшей пригородной зоне.

Анализируя вышеприведенные определения, можно сказать, что это стабильная система постоянных, регулярных пассажирских перевозок в регионе и между регионами, согласно с экономическими и социальными концепциями, проектами и планами развития страны.

Оценивая перспективу тех или иных направлений дальнейшего развития общественного транспорта необходимо соблюдать баланс между интересами потребителей транспортных услуг, нуждающихся в определенном уровне их качества, и транспортными предприятиями, которые заинтересованы в снижении собственных издержек. Эффективная организация управления системой городского пассажирского транспорта должна быть ориентирована на снижение влияния негативных факторов функционирования системы, учет интересов предприятий, при повышении уровня качества предоставляемых транспортных услуг.

Главное требование, которое необходимо соблюдать при [перевозке пассажиров](https://is-rent.ru/), — безопасность. Это относится как к людям в салоне автобуса, так и к водителю, пешеходам, а также к гражданам, находящимся в других транспортных средствах. Поэтому на работу по перевозке пассажиров принимаются только квалифицированный персонал. Важные качества для водителя: внимательность и стрессоустойчивость. Нужно чтобы и сам транспорт соответствовал всем требованиям к пассажирским автобусам, установленным действующим законодательством РФ.

Требования к пассажирским перевозкам автобусом в 2021 году установлены следующими нормативными актами:

* [Федеральным законом №220-ФЗ от 13.07.15](http://base.garant.ru/71129200/).
* [Уставом автотранспорта и городского наземного электротранспорта](http://base.garant.ru/12157005/).
* Постановлениями, приказами, инструкциями и письмами Ространснадзора.

Правила перевозки пассажиров обязательны к исполнению всеми юридическими лицами, осуществляющими деятельность в данной сфере. К водителю также предъявляются определенные требования.

Кроме перечисленных нормативных актов каждый регион РФ имеет право принимать в дополнение к ним свои законодательные документы. В уже существующие нормативы могут вноситься корректировки и дополнения, которые являются следствием проведенного анализа статистики всех дорожных происшествий с участием автобусов. Учитываются также пожелания водителей пассажирского транспорта. Чтобы исключить возможность допущения нарушений по причине неосведомленности, рекомендуется внимательно следить за всеми вносимыми изменениями.

Требования к пассажирским перевозкам автобусом:

* наличие пакета документов, необходимых для перевозки пассажиров;
* соблюдение водителем всех его профессиональных обязанностей;
* соответствие пассажирского автобуса всем техническим характеристикам, установленным действующими нормативами.

Соблюдение установленных нормативов обеспечивает безопасность не только жизни и здоровью граждан, но и их имуществу. Нарушение правил перевозки пассажиров влечет за собой административную ответственность, согласно АПК РФ.

Согласно требованиям законодательства, у водителя в салоне автобуса должен находиться пакет документов, содержащий:

* лицензию юридического лица (компании–перевозчика), предоставляющую право на осуществление деятельности по перевозке пассажиров;
* удостоверение водителя категории D;
* полис неограниченного страхования ОСАГО;
* путевой лист;
* справка, свидетельствующая о прохождении водителем медкомиссии;
* диагностическая карта технического осмотра автобуса;
* свидетельство о регистрации транспортного средства с присвоением ему государственных номеров;
* технический паспорт пассажирского автобуса.

В случае нарушения правил дорожного движения при отсутствии одного или нескольких документов из данного перечня к водителю или компании–перевозчику могут быть применены штрафные санкции.

Все вышеизложенные требования необходимо учитывать при создании программного продукта отчёта по практика.

Междугородние перевозки пассажиров могут осуществляться легковыми транспортными средствами, крупно и малогабаритными автобусами. согласно правилам проведения перевозки, принятым на федерально уровне, для перевозки людей может применяться транспортное средство одной из категорий:

* М1 или легковое такси – предполагаемая загруженность до 6 человек.
* М2 или микроавтобусы – количество пассажиров свыше восьми человек, общая грузоподъемность транспорта до 5 т.
* М3 или автобусы – транспортные средства, обустроенные для комфортной транспортировки более 10 пассажиров. Общая грузоподъемность транспорта свыше 5 т.

### Автобусы для междугородних перевозок классифицируются по ряду признаков в отдельные категории:

### По вместимости салона

* Микроавтобусы с малой вместимостью от 10 до 15 человек.
* Малогабаритные автобусики типа КАВЗ вместимостью до 25 пассажиров.
* Среднегабаритные автобусы, способные разместить до 35 человек.
* Большие автобусы типа ЛИАЗ вместимостью до 45 человек.
* Особо большие автобусы типа Икарус, вместимостью более 45 пассажиров.

### По количеству отсеков в салоне для размещения пассажиров

* Классический автобус с единым салоном.
* Сочлененный, имеющий 2 и более отделения салонов.
* Транспорт с полуприцепом.

### По количеству этажей

* Одноэтажные стандартные модели.
* Полутораэтажные.
* Двухэтажный (экскурсионный) транспорт для перевозки пассажиров.

### По предназначению

* Пассажирские автобусы общего пользования. Автобусы стандартной комплектации, применяемые для регулярных маршрутов или частной поездки по заказу клиента.
* Автобусы специального назначения. Транспортное средство, рассчитанное на транспортировку пассажиров с ограниченными возможностями, принадлежащее киностудии или государственной спецслужбе.

## Макеты входных данных

Характер, организация и предварительная подготовка входных данных:

* Сущность данных: информация о пассажирах и билетах на междугородные перевозки, направление, билет на которое хочет купить пассажир.
* Источники данных: регистрационные формы пассажиров.
* Сбор и обработка данных: данные собираются через регистрационные формы пассажиров, обрабатываются для проверки на достоверность и корректность.
* Предварительная подготовка: данные готовятся к вводу в систему путем структурирования и форматирования в соответствии с требованиями базы данных.

Формат, описание и способ кодирования входных данных:

* Формат данных: текстовый, числовой, дата/время.

Описание данных:

* Пассажиры: ФИО, дата рождения, пол, контактная информация(телефон, электронная почта), паспортные данные (серия, номер паспорта).

Способ кодирования: UTF-8.

Пример макета входных данных (данные о пассажирах) в формате MS Excel представлен на таблице 1.1.

Таблица 1.1. – Пассажиры:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Фамилия | Имя | Отчество | Дата рождения | Пол | Контактный телефон | Электронная почта | Серия паспорта | Номер паспорта |
| Иванова | Аня | Львовна | 1.1.2000 | Ж | +7989899889 | mail@mail.ru | 1111 | 1111111 |
| Борисов | Ян | Маркович | 2.2.2003 | М | +7989898889 | mail@mail.ru | 2222 | 2222222 |

## Макеты выходных данных

Характер и организация выходных данных:

* Сущность данных: информация о билетах, пассажирах, расписании рейсов, статистика перевозок.
* Формирование данных: данные будут формироваться на основе запросов к базе данных, анализа статистики и результатов операций с данными.

Формат и описание выходных данных:

* Формат данных: текстовый, числовой, дата/время.

Описание данных:

* Информация о билетах: номер билета, дата отправления, место назначения, тип билета, цена, статус (продан/не продан).
* Информация о пассажирах: ФИО, паспортные данные (серия, номер паспорта).
* Расписание рейсов: номер рейса, дата отправления, время отправления, ФИО водителя, направление, регистрационный номер автобуса, организация, осуществляющая перевозку.

Пример макета выходных данных(билета) в формате MS Excel представлен на таблице 1.2.

Таблица 1.2.- Билеты:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Номер рейса | Направление | Дата отправления | Время отправления | ФИО Пассажира | Серия и номер паспорта | Номер автобуса | Перевозчик |
| 1111 | Москва - Астрахань | 19.12.2023 | 13:43 | Иванов Иван Иванович | 1111 111111 | К232КУ199 | ИП Иванов И.И. |

## Постановка задачи

В рамках отчёта по практике требуется создать информационную систему, учитывающую междугородные пассажирские перевозки.  По результатам анализа предметной области было определено, что необходимо разработать базу данных, которая будет содержать информацию о маршрутах междугородных пассажирских перевозок, расписании рейсов, билетах, пассажирах и транспортных средствах. Это включает в себя создание сущностей (таблиц) для хранения данных о маршрутах, рейсах, билетах, пассажирах, автобусах/поездах и других средствах передвижения.

Также были определены требования к автоматизируемым функциям:

* Реализовать возможность поиска и бронирования билетов на конкретные маршруты .Для этого необходимо создать функционал, позволяющий пользователям искать доступные рейсы, выбирать удобное время отправления и прибытия, а также бронировать билеты.
* Реализовать возможность просмотра информации по рейсу: статус рейса, название перевозчика, цена билета и т.д. Это значительно упростит работу пользователя и его взаимодействие с системой. Для этого необходимо создать функционал, позволяющий пользователям искать доступные по номеру, выбирать удобное время отправления и прибытия, а также бронировать билеты.
* Реализовать функцию обработки данных системы и представления их в понятном для пользователя виде(в виде документа MS Excel, MS Word). Это позволит автоматизировать процесс выдачи/аннуляции билетов пассажиров на рейс.
* Обеспечить возможность формирования отчетов о загруженности рейсов, доходности маршрутов и других статистических данных. Это позволит анализировать эффективность перевозок, выявлять популярные маршруты и принимать управленческие решения на основе полученной информации.

В процессе разработки базы данных будет использоваться математическая модель для определения оптимальных маршрутов и графиков движения транспортных средств с учетом времени в пути, расстояний и потенциальной загруженности. Это поможет оптимизировать расписание рейсов и обеспечить эффективное использование транспортных средств.

Необходимо разработать алгоритмы для хранения и обработки информации о маршрутах, расписании рейсов, билетах, пассажирах и транспортных средствах. Например, это может включать алгоритмы поиска доступных рейсов, проверки наличия свободных мест, расчета стоимости билетов и другие операции.

Также необходимо обеспечить автоматическое обновление расписания и информации о доступных билетах. Это требует разработки механизмов автоматической синхронизации данных внутри системы (информации в базе данных).

Также следует предусмотреть возможность взаимодействия платежными системами для онлайн-оплаты билетов, а также интеграцию с системами учета и отчетности. Это позволит обеспечить полноценную работу сервиса по продаже билетов и учету финансовых операций.

Основной проблемой в ходе реализации вышесказанных задач является архитектуры программного продукта. В ходе анализа предметной области были рассмотрены:

* Desktop-приложение на языке C#;
* Web-приложение(сайт).

Однако сразу выяснились основные минусы проектирования Desktop-приложения:

* Ограниченная доступность: Десктоп-приложения могут быть установлены только на определенные операционные системы (например, Windows, macOS, Linux), что ограничивает их использование для пользователей, которые предпочитают другие платформы.
* Требуется установка: Для использования десктоп-приложений пользователи должны сначала установить их на свои компьютеры. Это может быть неудобно для пользователей, которые не хотят заниматься установкой программного обеспечения или имеют ограниченные права на установку на своих компьютерах.
* Обновления: Десктоп-приложения требуют регулярных обновлений для исправления ошибок и добавления новых функций. Пользователям приходится следить за новыми версиями и устанавливать обновления вручную.
* Зависимость от аппаратного обеспечения: Десктоп-приложения могут требовать определенных характеристик аппаратного обеспечения, таких как процессор, оперативная память или графический процессор. Это может ограничить использование приложения на старых или слабых компьютерах.
* Ограниченная мобильность: Десктоп-приложения не могут быть легко использованы на мобильных устройствах, таких как смартфоны или планшеты. Это может быть неудобно для пользователей, которые хотят иметь доступ к приложению в любое время и в любом месте.

В отличие от Desktop-приложения, у Web-приложения нет выше описанных недостатков. Для корректной работы достаточно иметь браузер и интернет-соединение.

## Цель создания автоматизированной (информационной) системы.

Целью отчёта по практике является создание Web-приложения для управления информацией о междугородных пассажирских перевозках.

Достижение данной цели сопровождается следующими задачами:

- изучить необходимый теоретический материал;

- провести анализ предметной области, составить примерный план разработки;

- проектирование базы данных;

- с помощью среды Visual Code, разработать программный продукт;

- провести тестирование полученной системы.

В ходе работы будет проведен анализ существующих баз данных в этой области, выявлены основные требования к хранению и обработке информации, а также разработана структура базы данных, учитывающая специфику данного вида транспорта.

Данная работа предполагает изучение основных принципов проектирования баз данных, использование соответствующих инструментов и технологий, а также практическую проверку разработанной модели на конкретных примерах данных о междугородных пассажирских перевозках.

## Назначение автоматизированной (информационной) системы.

Назначение программного продукта (Web-приложения):

* Обеспечить автоматизированную работу автовокзала: программа будет предоставлять возможность автоматизированного учета расписания рейсов, продажи билетов, контроля за запасами, а также возможность просмотра и анализа статистической информации о работе автовокзала.
* Облегчить работу сотрудников автовокзала: система предоставит автоматизированные инструменты для бухгалтеров, кассиров, контролеров и водителей, что позволит им эффективнее выполнять свои обязанности, упростит процесс учета и отчетности, а также повысит общую эффективность работы.
* Создать возможность пассажирам осуществлять покупку билетов: веб-приложение позволит пассажирам самостоятельно выбирать и приобретать билеты на нужные им маршруты и рейсы, осуществлять оплату онлайн, а также получать электронные билеты, что упростит процесс покупки и сэкономит время как для пассажиров, так и для сотрудников автовокзала.

## Обзор аналогов.

Аналоги – это продукты или услуги, которые предлагаются различными компаниями, но имеют сходные характеристики или функциональность. В мире технологий и бизнеса аналоги часто сравниваются между собой, чтобы определить их преимущества и недостатки.

При выборе аналогов важно учитывать не только цену, но и качество, функциональность, надежность и удобство использования. Пользователи часто сравнивают аналоги, чтобы определить, какой из них лучше соответствует их потребностям и ожиданиям.

Обзор аналогов помогает потенциальным покупателям принять обоснованное решение при выборе между несколькими продуктами или услугами. Он может включать в себя сравнительный анализ основных характеристик, отзывы пользователей, экспертные мнения и другую информацию, которая поможет сделать правильный выбор.

В зависимости от конкретной сферы деятельности, аналоги могут быть представлены в виде различных товаров, программного обеспечения, сервисов или технологий. При этом важно учитывать индивидуальные потребности и требования, чтобы выбрать наиболее подходящий вариант.

## Аналог №1. Biletion Office.

"Biletion Office" — программа для заказных и регулярных пассажирских перевозок междугороднего и международного сообщения. Программа Biletion Office предоставляет руководителю полный арсенал отчетов и историй заявок, чтобы вы имели полную картину о работе вашей компании в любое время и в любом месте. Кроме того, с использованием программы сводятся к минимуму случаи воровства и мошенничества как среди сотрудников компании, так и среди пассажиров. Использование программы возможно как в варианте перевозок пассажиров по межгороду, так и для учета заказных перевозок.

**Основные характеристики программы:**

* Хранение контактных данных и истории поездок клиентов.
* Быстрый алгоритм записи клиента на рейс.
* Возможность многопользовательской работы с программой.
* Формирование отчетов в реальном времени с показателями деятельности.
* Минимизация краж со стороны сотрудников, полный контроль для руководителя.
* Возможность отправки SMS сообщений пассажирам.
* Наличие мобильной версии для водителей и экспедиторов.
* Интеграция с Oktell IP АТС.

Интерфейс программы представлен на рисунке 1.1.

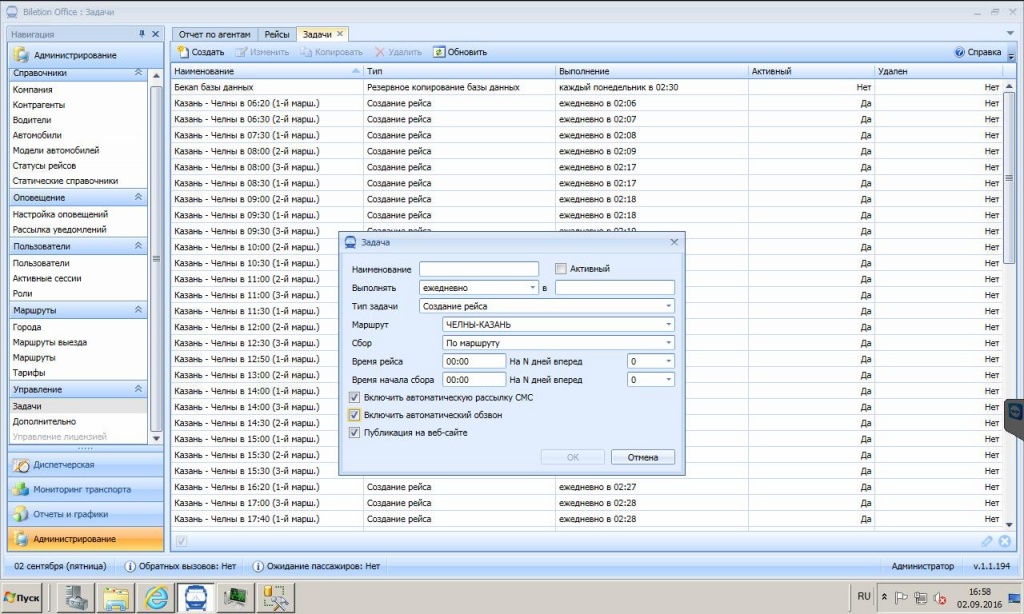


Рисунок 1.1. – Интерфейс программы Biletion Office

## Аналог №2. 1С: Управление транспортом.

Программный продукт «1С:Управление автотранспортом Проф» — совместное решение фирмы «1С» и компании «1С-Рарус», предназначенное для автоматизации управленческого и оперативного учета работы автопарка:

* в автотранспортных предприятиях;
* в транспортных подразделениях торговых и производственных компаний;
* в прочих компаниях с различной отраслевой спецификой.

Программа широко известна в среде российских предприятий, ее успешно используют ПАО «Автоваз», ФГУП «Почта России», ПАО «РусГидро», УК «Татнефть-ТрансСервис», ООО «Компания Востсибуголь», ООО «ГазпромТрансгаз Югорск» и многие другие компании, обладающие автопарком более 100 транспортных единиц.

Программа включена в Единый реестр российских программ.

Интерфейс программы представлен на рисунке 1.2.

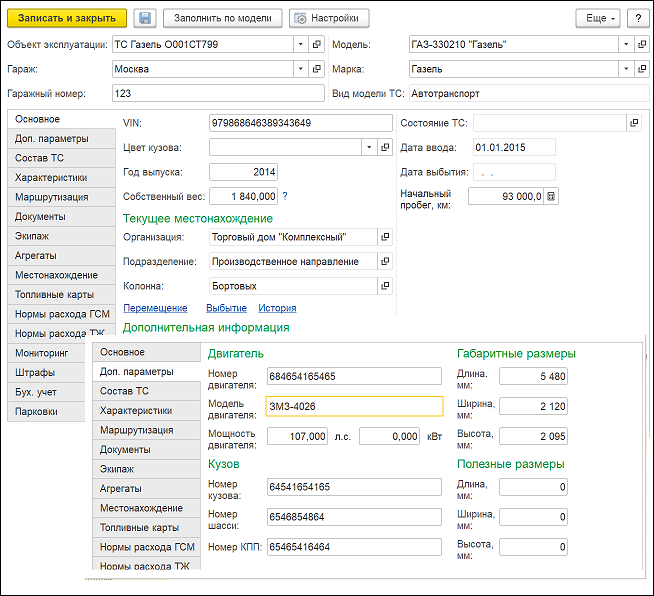


Рисунок 1.2. – Интерфейс программы 1С: Управление транспортом.

## Аналог №3. Busfor Manager.

Busfor Manager (Busfor.ru) - сайт, специально разработанный для автобусных перевозок. Он включает в себя учет билетов, маршрутов, продажи и отчетности. Пользователи отмечают удобство использования интерфейса и гибкость настроек программы под конкретные потребности компании.

Интерфейс сайта представлен на рисунке 1.3.

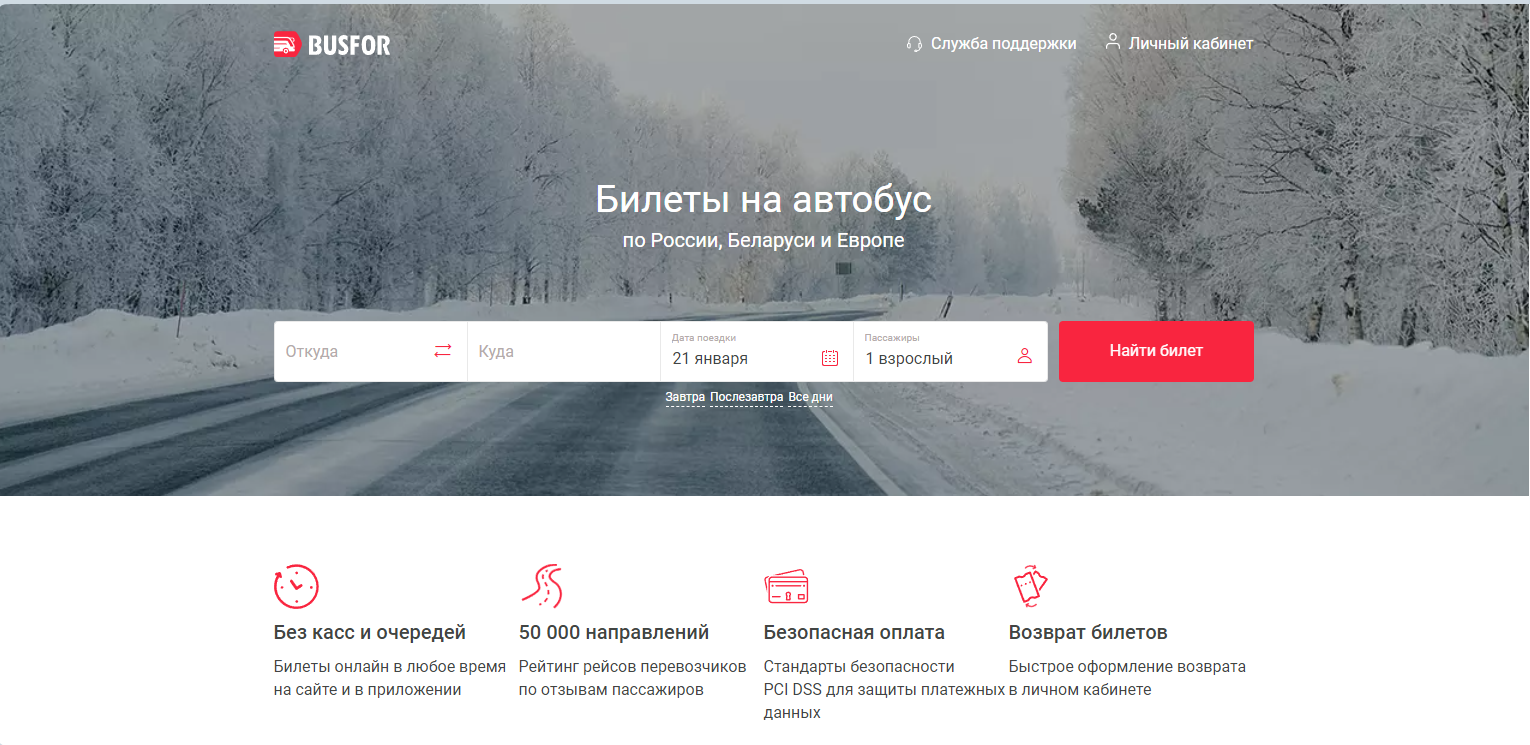


Рисунок 1.3. – Интерфейс сайта Busfor.ru.

## Сравнительная таблица аналогов.

Как было выяснено ранее наш программный продукт должен быть представлен в виде Web-приложения. Согласно требованиям к системе должно существовать две версии сайта: для клиента(пассажира) и сотрудника. Принципиальной является возможность осуществлять многокритериальный поиск и отображение статистики (для сотрудников). Полезным дополнением будет возможность бесплатного получения ПО.

Ниже в таблице 1.3. представлена сравнительная таблица аналогов разрабатываемой системы.

Таблица 1.3. - Сравнительная таблица аналогов.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Функция** | **Biletion Office** | **1C: Управление транспортом** | **Busfor Manager** | **Разрабатываемая система** |
| 1 | Web-приложение | **-** | **-** | **+** | **+** |
| 2 | Версия для сотрудников | **+** | **+** | **-** | **+** |
| 3 | Бесплатное ПО | **-** | **-** | **+** | **+** |
| 4 | Версия для клиентов | **-** | **-** | **+** | **+** |
| 5 | Многокритериальный поиск | **+** | + | + | **+** |
| 6 | Графическое отображение статистики | **+** | + | **-** | **+** |

## Достоинства и недостатки аналогов.

К достоинствам аналогов можно отнести высокую надёжность и репутацию. Все три аналога уже давно зарекомендовали себя на рынке пассажирских перевозок.

Однако ни один из исследованных аналогов не имеет два пользовательских интерфейса: для сотрудников и для клиента. Biletion Office и 1C: Управление транспортом являются Desktop-приложениями: это означает, что их придётся периодически обслуживать и обновлять с выходом новых версий. Да к тому же обе программы не являются бесплатно распространяемым программным обеспечением. Разрабатываемая система лишена всех вышеперечисленных недостатков.

## Технология обработки информации

По результатам анализа предметной области было определено, что в системе существует несколько ролей пользователей:

* клиент (или пассажир);
* сотрудник автовокзала (имеет доступ как к общедоступной, так и секретной информации);

На рисунках А.1., А.2. в Приложении представлены диаграммы вариантов использования, которые содержат основной состав и порядок основных событий, генерируемых пользователями при работе с системой в зависимости от его роли.

Все пользователи вне зависимости от роли имеют доступ к информации о маршрутах, автобусах, расписании, могут проводить сравнительный анализ с помощью применения многокритериального поиска.

Пользователю с ролью Клиент доступно онлайн-бронирование билетов на сайте. Также у него есть возможность отмены бронирования до начала рейса.

Пользователю с ролью Сотрудник доступно редактирование информации о рейсах и перевозчиков, расписания, добавление и удаление техники из базы данных системы. Также пользователю с данной ролью доступна возможность мониторинга(контроля) рейса в пути: отслеживание местоположения автобуса в пути.

Чтобы начать работу с сайтом, клиенту необходимо заполнить форму, указав свои контактные и персональные данные. После регистрации пользователю доступен весь клиентский функционал сайта: у клиента есть возможность забронировать любой доступный, не забронированный другим пользователем билет, выкупить, а также отменить его бронирование до начала регистрации на рейс. В Приложении на рисунке А.3. представлена диаграмма активности пользователя с ролью Клиент.

Начало работы сотрудника с сайтом отличается от клиента: у него уже есть логин и пароль с правами сотрудника, после введения данных (логина и пароля) пользователю доступен весь функционал сотрудника. Он несколько отличается от функционала клиента. Ему также доступен просмотр информации о будущих рейсах, автобусах и т.д. Но у сотрудника есть возможность редактирования этой информации: отмена(удаление) рейса, изменения расписания, маршрутов, автобусов. В Приложении на рисунке А.4. представлена диаграмма активности пользователя с ролью Сотрудник.

По результатам анализа предметной области и требований к системе было определено, что необходимо спроектировать базу данных. Спроектированная база данных должна содержать:

* Информацию о пассажирах
* Информацию о рейсах
* Информацию о типах технике
* Информацию о сотрудниках
* Информацию о билетах
* Информацию о местах в автобусе
* Информацию о маршрутах перевозок
* Информацию о городах
* Информацию о комбинации место-рейс
* Информацию о перевозчиках
* Информацию о типах ВУ
* Информацию о должностях
* Информацию об автобусах

Соответствующая диаграмма классов представлена на рисунке А.5. в Приложении.

На рисунке А.6. в Приложении представлена диаграмма состояний сущности Место. Рассмотрим основные состояния этой сущности. Допустим, что в систему пользователем Сотрудник был добавлен рейс, при этом в систему автоматически добавляются свободные места, которые доступны для бронирования Клиентам. После бронирования клиенту предоставляется возможность выкупа билета, а после завершения рейса Сотрудник удаляет его и после этого удаляются места из системы.

## Требования к техническому и программному обеспечению

Требование к программному обеспечению:

* ОС – Windows 10.
* Браузер Google Chrome

Требования к техническому обеспечению:

* IBM-совместимый компьютер с МП Intel Pentium III или AMD K6 с тактовой частотой 500 МГц и выше;
* не менее 512 МБ ОЗУ;
* соединение скоростью не менее 128 кбит/с;
* НЖМД со свободным со свободным объемом не менее 500 МБ
* 101-клавишная клавиатура;
* манипулятор «мышь».

# РАБОЧИЙ ПРОЕКТ

## Обоснование выбора решения.

Перед началом написания программного продукта необходимо выбрать среду разработки. Идеальным выбором для программиста является Visual Studio Code. Данная система отличается удобным интерфейсов, возможностью использования встроенной командной строки.

Значительно упростит работу по созданию Web-приложения использование фреймворков. Фреймворк - это набор библиотек, инструментов и структур, предназначенных для упрощения разработки программного обеспечения. Он предоставляет разработчикам готовые компоненты, шаблоны и архитектурные принципы, которые позволяют создавать приложения более эффективно и структурированно.

Основные преимущества использования фреймворков включают:

* Ускорение разработки: Фреймворк предоставляет готовые компоненты и инструменты, которые позволяют разработчикам создавать приложения быстрее, поскольку им не нужно писать все с нуля.
* Структурирование кода: Фреймворк обычно предлагает определенную архитектуру приложения, что помогает разработчикам организовать свой код и следовать bewst practices.
* Улучшение безопасности: Многие фреймворки включают в себя встроенные механизмы безопасности, такие как защита от SQL-инъекций, CSRF и XSS атак.
* Поддержка множества технологий: Фреймворк может предоставлять интеграцию с различными технологиями, такими как базы данных, шаблонизаторы, аутентификация и многое другое.
* Сообщество и документация: Популярные фреймворки имеют большие сообщества разработчиков и обширную документацию, что облегчает поддержку и развитие проектов.

В рамках создания программного продукта будет использоваться фреймворк Laravel. Laravel - это один из самых популярных фреймворков для разработки веб-приложений на языке PHP. Вот несколько преимуществ Laravel перед другими фреймворками:

* Удобство использования: Laravel предлагает чистый и интуитивно понятный синтаксис, что делает его легким для изучения и использования.
* Мощный ORM (Object-Relational Mapping): Laravel поставляется с встроенным ORM под названием Eloquent, который позволяет разработчикам работать с базой данных, используя объектно-ориентированный подход, что упрощает взаимодействие с базой данных
* Маршрутизация и контроллеры: Laravel предоставляет удобный механизм для определения маршрутов и контроллеров, что делает процесс создания веб-приложений более простым и эффективным.
* Встроенная поддержка аутентификации и авторизации: Laravel предоставляет готовые инструменты для реализации системы аутентификации и авторизации пользователей, что упрощает создание защищенных веб-приложений.
* Активное сообщество: Laravel имеет огромное активное сообщество разработчиков, которые постоянно обновляют и улучшают фреймворк, а также предоставляют обширную документацию и поддержку.

Хотя Laravel является одним из самых популярных и мощных фреймворков для разработки веб-приложений на PHP, у него также есть некоторые недостатки, которые следует учитывать:

* Производительность: В некоторых случаях Laravel может быть несколько медленнее по сравнению с другими фреймворками, особенно при работе с большим объемом данных. Однако это различие может быть незаметным для большинства приложений.
* Ресурсы: Использование Laravel может потребовать больше ресурсов сервера по сравнению с более легковесными фреймворками, что может повлиять на требования к хостингу и масштабируемость приложения.
* Обновления и совместимость: Поскольку Laravel постоянно обновляется, иногда могут возникать проблемы совместимости с более старыми версиями приложений или сторонними библиотеками.

Несмотря на эти недостатки, Laravel остается одним из самых популярных и мощных фреймворков для разработки веб-приложений на PHP, и его преимущества перевешивают недостатки.

В рамках работы над данными необходимо определиться с тем, какую СУБД стоит использовать в рамках реализации программного продукта.

PostgreSQL - это мощная и надежная объектно-реляционная система управления базами данных (СУБД), которая имеет множество преимуществ. Вот некоторые из них:

* Расширенная поддержка стандартов: PostgreSQL стремится к полной совместимости с ANSI SQL и другими стандартами, что делает его более предсказуемым и удобным для разработчиков, работающих с различными СУБД.
* Мощные возможности: PostgreSQL обладает широким спектром возможностей, включая поддержку JSON, XML, массивов, геоданных, полнотекстового поиска, триггеров, хранимых процедур и многое другое. Это позволяет разработчикам создавать сложные и масштабируемые приложения.
* Надежность и целостность данных: PostgreSQL обеспечивает высокую степень надежности и целостности данных благодаря механизмам транзакций, контролю целостности, репликации данных и механизмам восстановления после сбоев.
* Масштабируемость: PostgreSQL обладает возможностями для горизонтального и вертикального масштабирования, что позволяет эффективно управлять растущими объемами данных и нагрузкой на систему.
* Поддержка различных операционных систем: PostgreSQL поддерживается на различных операционных системах, включая Linux, Windows, macOS, FreeBSD и другие.

Эти преимущества делают PostgreSQL привлекательным выбором для разработчиков и организаций, стремящихся к созданию надежных и масштабируемых приложений с высокой степенью гибкости и функциональности.

Хотя PostgreSQL имеет множество преимуществ, у него также есть некоторые недостатки, которые стоит учитывать при выборе этой СУБД. Вот некоторые из них:

* Сложность настройки: Несмотря на то, что PostgreSQL обладает мощными возможностями, его настройка и управление могут быть сложными для новичков. Настройка параметров, оптимизация производительности и обслуживание базы данных могут потребовать специальных знаний и опыта.
* Относительно медленная производительность в некоторых случаях: В сравнении с некоторыми другими СУБД, такими как MySQL или MongoDB, PostgreSQL может показывать относительно медленную производительность в определенных сценариях работы с большими объемами данных или сложными запросами.
* Ограничения в ресурсах: В зависимости от конфигурации и характеристик сервера, PostgreSQL может иметь ограничения по использованию ресурсов, таких как память, процессорное время и дисковое пространство.
* Ограниченная поддержка NoSQL: В отличие от некоторых NoSQL баз данных, PostgreSQL не является идеальным выбором для работы с некоторыми типами данных, такими как ключ-значение или документы, хотя имеет поддержку JSON и XML.

Учитывая вышеописанные преимущества и недостатки, следует отметить, что в рамках реализации программного продукта функционала и возможностей СУБД PostgreSQL будет более чем достаточно.

## Дизайн основных интерфейсов.

В ходе проектирования программного продукта необходимо разработать несколько интерфейсов: форма авторизации, где пользователи будут вводить логин и пароль для начала работы с системой и пользовательский интерфейс. Пример формы авторизации представлен на рисунке 2.1. Она открывается при первой запуске сайта пользователем. Для продолжения работы необходимо правильно заполнить поля E-mail и Password и выполнить вход в систему.

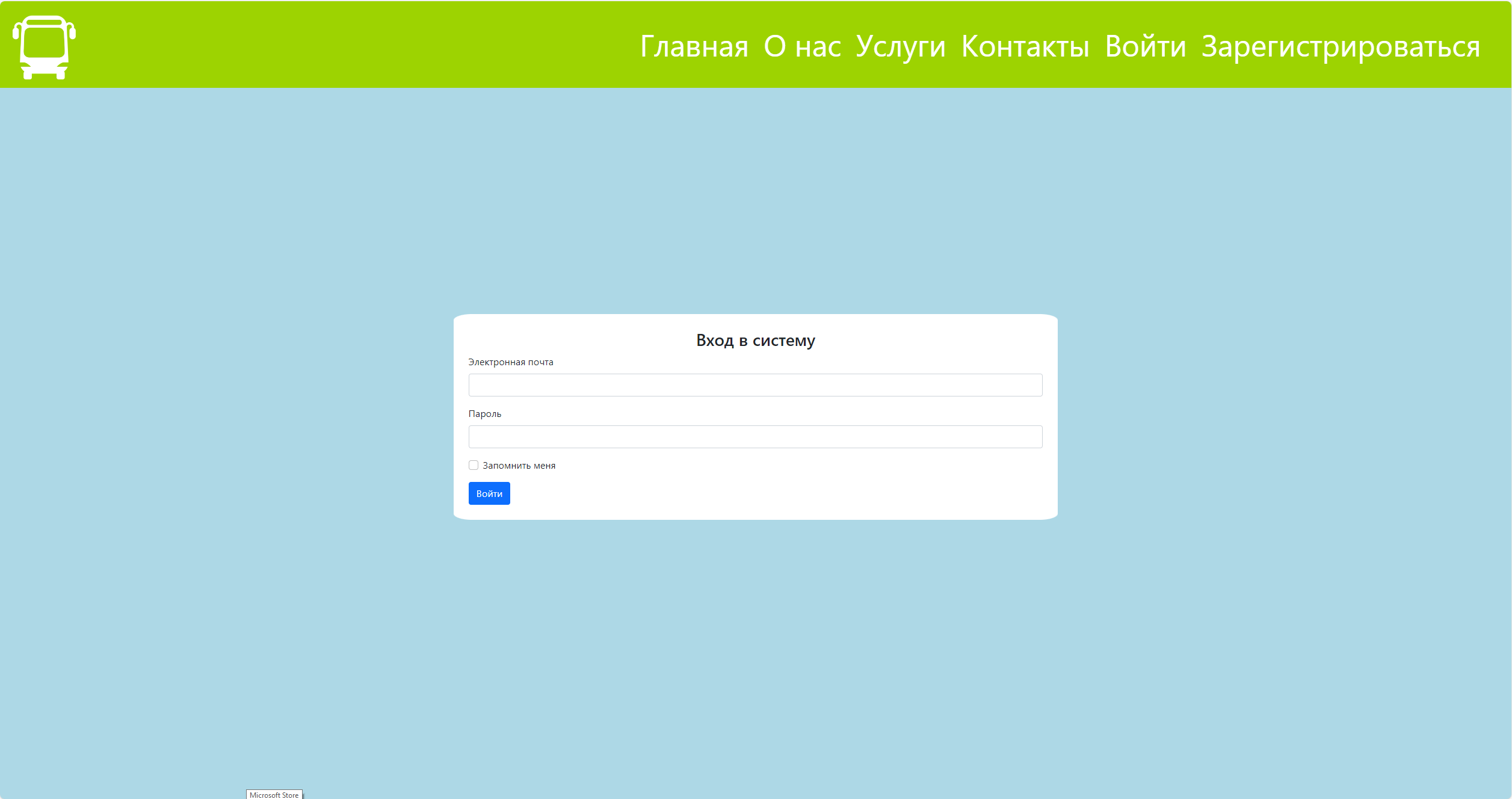


Рисунок 2.1. – Форма авторизации.

После успешной авторизации на экране пользователя появляется главная страница. Благодаря «шапке» сайта пользователю доступна быстрая навигация между разделами. На основной части страницы тезисно написана информация о деятельности автовокзала. Также есть возможность быстрого доступа к расписанию и покупке билетов. Проект главной страницы представлен на рисунке 2.2.

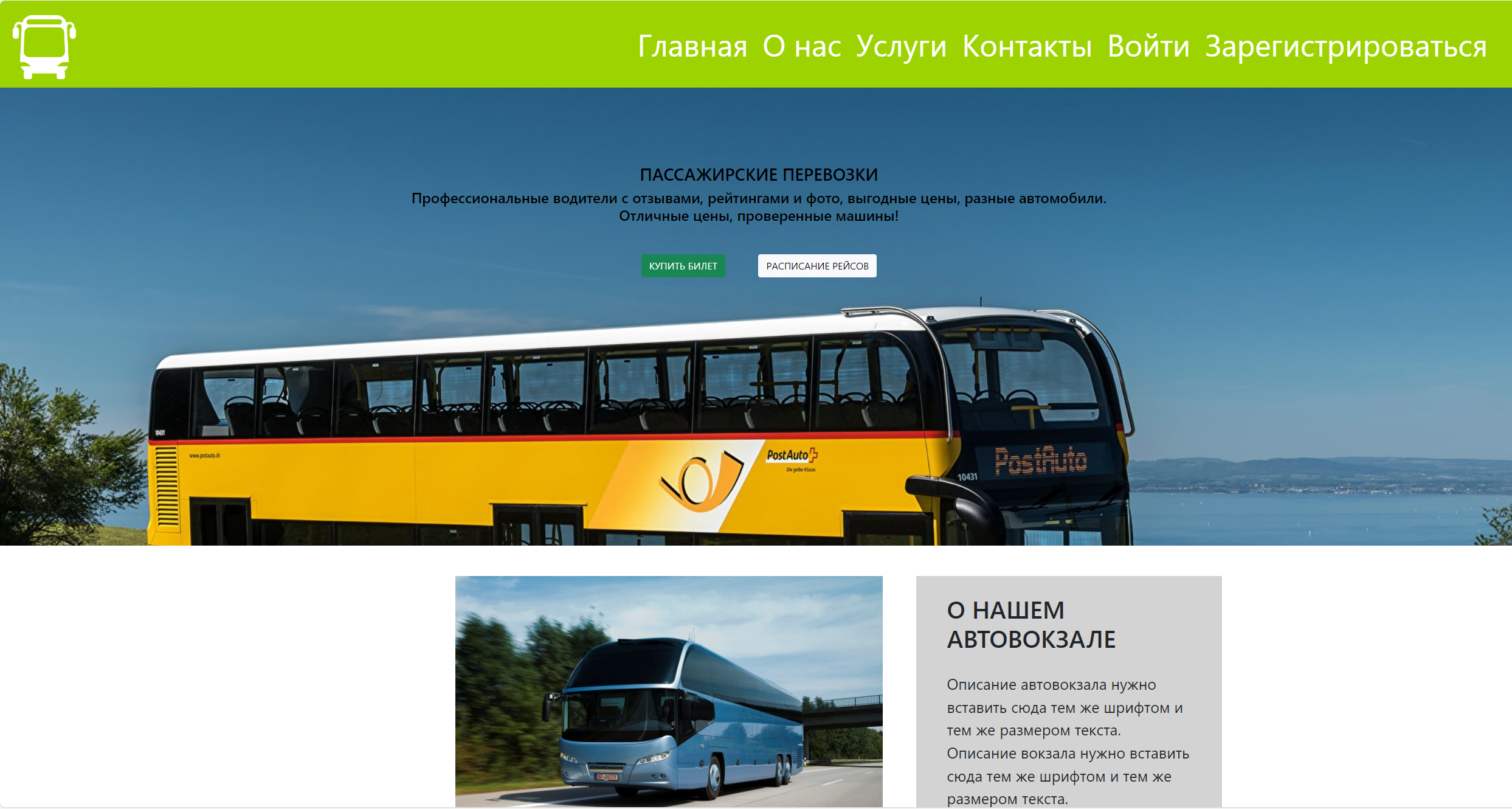


Рисунок 2.2. - Главная страница

Если продолжить рассмотрение главной страницы, то есть возможность просмотреть список сотрудников автовокзала с описанием их должности и деятельности. Данный интерфейс представлен на рисунке 2.3.



Рисунок 2.3. – Сотрудники автовокзала.

Спроектированные интерфейсы интуитивно понятны как опытному, так и начинающему пользователю.

## Установка системы.

1. Скачайте проект:

* Можно клонировать репозиторий с помощью Git: git clone https://github.com/RAdKuzur/ASTU.git
* Или скачать архив с проектом из диска и распаковать его.

1. Установите зависимости:

* Перейдите в директорию проекта.
* Запустите composer install для установки всех необходимых зависимостей.

1. Установите настройки окружения:

* Создайте файл .env на основе .env.example.
* Укажите параметры подключения к базе данных, почтовый сервер и другие настройки.

1. Сгенерируйте ключ приложения:Запустите php artisan key:generate для генерации уникального ключа приложения.
2. Создайте структуру базы данных:
3. Запустите миграции: php artisan migrate, чтобы создать необходимые таблицы в базе данных.
4. Запустите локальный встроенный сервер Laravel: php artisan serve. После этого сайт будет доступен по адресу <http://localhost:8000> или <http://localhost:8000/m>ain.
5. Запустите проект: Перейдите в браузере по указанному адресу и убедитесь, что сайт работает корректно.

## Описание физической архитектуры системы.

Для работы системы необходимо наличие компьютера с присутствующей на ней базой данных автовокзала. В качестве СУБД используется PostgreSQL 15. Доступ к базам данных с клиентских компьютеров, на которых установлена исполняемая программа, осуществляется благодаря фреймворку Laravel.

## Даталогическая модель данных.

Физическая модель данных базы данных автовокзала в СУБД PostgreSQL включает в себя таблицы 'buses', 'carriers', 'cities', 'comments', 'customers', model\_buses', 'routes', 'runs', 'seat\_runs', 'seats', 'tickets', 'titles' для обеспечения эффективного хранения и управления данными.

Рассмотрим каждую таблицу детальнее. Справочник «buses» предназначен для хранения информации о тестированиях студентов (таблица 2.1).

Таблица 2.1 - Справочник «buses»

| **Параметр** | **Тип** | **Размер** | **Примечание** |
| --- | --- | --- | --- |
| id | serial, bigint, not null |  | Номер записи. Уникальное значение. Первичный ключ. |
| model\_id | integer, not null |  | Внешний ключ. |
| number | text, not null |  | Регистрационный номер автобуса |
| status | character varying, not null | 255 | Степень готовности автобуса к рейсу. |
| seats | integer, not null |  | Вместимость автобуса. |

Таблица 2.2 - Справочник «carriers»

| **Параметр** | **Тип** | **Размер** | **Примечание** |
| --- | --- | --- | --- |
| id | serial, bigint, not null |  | Номер записи. Уникальное значение. Первичный ключ. |
| name | character varying, not null | 255 | Название организации перевозчика |

Таблица 2.3 - Справочник «cities»

| **Параметр** | **Тип** | **Размер** | **Примечание** |
| --- | --- | --- | --- |
| id | serial, bigint, not null |  | Номер записи. Уникальное значение. Первичный ключ. |
| name | character varying, not null | 255 | Название города |

Таблица 2.4 - Справочник «comments»

| **Параметр** | **Тип** | **Размер** | **Примечание** |
| --- | --- | --- | --- |
| id | serial, bigint, not null |  | Номер записи. Уникальное значение. Первичный ключ. |
| comment | text, not null |  | Текст комментария. |
| customer\_id | integer, not null |  | Внешний ключ |

Таблица 2.5 - Справочник «customers»

| **Параметр** | **Тип** | **Размер** | **Примечание** |
| --- | --- | --- | --- |
| id | serial, bigint, not null |  | Номер записи. Уникальное значение. Первичный ключ. |
| surname | character varying, not null | 255 | Фамилия пользователя. |
| name | character varying, not null | 255 | Имя пользователя. |
| passport\_series | integer, not null |  | Серия паспорта. |
| passport\_number | integer, not null |  | Номер паспорта. |
| email | character varying, not null | 255 | Электронная почта пользователя. |
| password | character varying, not null | 255 | Пароль к учетной записи. |
| role | integer, not null |  | Роль пользователя. |

Таблица 2.6 - Справочник «model\_buses»

| **Параметр** | **Тип** | **Размер** | **Примечание** |
| --- | --- | --- | --- |
| id | serial, bigint, not null |  | Номер записи. Уникальное значение. Первичный ключ. |
| brand | character varying, not null | 255 | Марка автобуса. |
| model | character varying, not null | 255 | Модель автобуса. |

Таблица 2.7 - Справочник «routes»

| **Параметр** | **Тип** | **Размер** | **Примечание** |
| --- | --- | --- | --- |
| id | serial, bigint, not null |  | Номер записи. Уникальное значение. Первичный ключ. |
| arrival\_city\_id | integer, not null |  | Внешний ключ. |
| departure\_city\_id | integer, not null |  | Внешний ключ. |

Таблица 2.8 - Справочник «runs»

| **Параметр** | **Тип** | **Размер** | **Примечание** |
| --- | --- | --- | --- |
| id | serial, bigint, not null |  | Номер записи. Уникальное значение. Первичный ключ. |
| bus\_id | integer, not null |  | Внешний ключ. |
| route\_id | integer, not null |  | Внешний ключ. |
| departure\_time | timestamp without time zone, not null | 0 | Время отправления. |
| arrival\_time | timestamp without time zone, not null | 0 | Время прибытия. |
| status | character varying, not null | 255 | Статус рейса. |
| carrier\_id | integer, not null |  | Внешний ключ. |

Таблица 2.9 - Справочник «seat\_runs»

| **Параметр** | **Тип** | **Размер** | **Примечание** |
| --- | --- | --- | --- |
| id | serial, bigint, not null |  | Номер записи. Уникальное значение. Первичный ключ. |
| seat\_id | integer, not null |  | Внешний ключ. |
| run\_id | integer, not null |  | Внешний ключ. |
| customer\_id | integer, not null |  | Внешний ключ. |
| flag | integer, not null |  | Статус места в рейсе. |
| price | integer, not null |  | Цена билета. |

Таблица 2.10 - Справочник «seats»

| **Параметр** | **Тип** | **Размер** | **Примечание** |
| --- | --- | --- | --- |
| id | serial, bigint, not null |  | Номер записи. Уникальное значение. Первичный ключ. |
| bus\_id | integer, not null |  | Внешний ключ. |
| number | integer, not null |  | Номер места в автобусе. |

Таблица 2.11 - Справочник «tickets»

| **Параметр** | **Тип** | **Размер** | **Примечание** |
| --- | --- | --- | --- |
| id | serial, bigint, not null |  | Номер записи. Уникальное значение. Первичный ключ. |
| seat\_run\_id | integer, not null |  | Внешний ключ. |
| customer\_id | integer, not null |  | Внешний ключ. |
| carrier\_id | integer, not null |  | Внешний ключ. |
| code | character varying, not null | 255 | Код билета. |

## Определение целостности сущностей.

Каждая запись сущности должна обладать уникальным идентификатором и содержать данные. В базе данных PostgreSQL первичные ключи играют важную роль для обеспечения уникальности и целостности данных. Вот несколько оснований для использования первичных ключей в базе данных PostgreSQL:

1. **Уникальность записей**: Первичный ключ гарантирует уникальность значений в столбце (или группе столбцов) таблицы, что исключает дублирование данных и помогает избежать ошибок вставки повторяющихся записей.
2. **Идентификация записей**: Первичный ключ позволяет однозначно идентифицировать каждую запись в таблице, обеспечивая простой и эффективный способ ссылаться на конкретные данные внутри базы данных.
3. **Оптимизация поиска и соединения**: Использование первичного ключа в качестве ключа для соединения таблиц ускоряет процессы поиска и объединения данных, так как PostgreSQL использует индексы первичного ключа для быстрого доступа к записям.
4. **Создание внешних ключей**: Первичный ключ часто используется для определения внешних ключей, которые обеспечивают связи между таблицами и поддерживают целостность данных путем обеспечения соблюдения правил ссылочной целостности.
5. **Автоматическое управление поведением**: PostgreSQL может автоматически создавать индексы для первичных ключей, что обеспечивает оптимизацию производительности при выполнении запросов к данным.

Таким образом, использование первичных ключей в базе данных PostgreSQL помогает обеспечить уникальность данных, облегчает доступ к информации, оптимизирует работу с данными и поддерживает целостность и организацию структуры базы данных.

В таблице 2.12 представлены уникальные идентификаторы, используемые в физической модели данных.

Таблица 2.12 - Уникальные идентификаторы, используемые в физической модели данных.

| **Название** | **Название таблицы** | **Название столбца** |
| --- | --- | --- |
| PK\_buses | buses | Id |
| PK\_carriers | carriers | Id |
| PK\_cities | cities | Id |
| PK\_comments | comments | Id |
| PK\_customers | customers | Id |
| PK\_model\_buses | model\_buses | Id |
| PK\_routes | routes | Id |
| PK\_runs | runs | Id |
| PK\_seat\_runs | seat\_runs | Id |
| PK\_seats | seats | Id |
| PK\_tickets | tickets | Id |

## Определение целостности атрибутов.

Ограничение UNIQUE в реляционных базах данных используется для гарантирования уникальности значений в определенном столбце или наборе столбцов. Как только определен столбец как UNIQUE, база данных не позволит вставить дублирующиеся значения в этот столбец, обеспечивая тем самым целостность данных и уникальность записей. Это особенно полезно для ключевых столбцов, таких как идентификаторы или имена пользователей, где необходимо исключить повторяющиеся значения.

Ограничение CHECK, с другой стороны, позволяет определить условие, которое должно быть выполнено для каждой записи в таблице. Если запись не соответствует условию CHECK, операция вставки или обновления будет отклонена.

Таким образом, UNIQUE и CHECK - это два важных ограничения, которые помогают обеспечить целостность и связность данных в базах данных, предотвращая вставку нежелательных или некорректных значений. Их использование способствует поддержанию качественной информации и предотвращает ошибки в данных.

В таблице 2.13 представлены ограничения, используемые в физической модели данных.

Таблица 2.13 - Ограничения, используемые в физической модели данных.

| **Название поля** | **Название таблицы** | **Тип ограничения** | **Обоснование** |
| --- | --- | --- | --- |
| number | buses | UNIQUE | Невозможно существование двух автомобилей с одинаковыми регистрационными номерами. |
| name | carriers | UNIQUE | Невозможно существование двух организаций с одинаковыми названиями. |
| name | cities | UNIQUE | Невозможно существование двух городов с одинаковыми названиями. |
| price | seat\_runs | CHECK(price > 0) | Цена всегда неотрицательное число |

## Определение целостности атрибутов.

В таблице 2.14 представлено описание внешних ключей физической модели данных.

Таблица 2.14 – Описание внешних ключей

| **Название** | **Название главной таблицы** | **Название столбца в главной таблице** | **Название подчиненной таблицы** | **Название столбца в подчинённой таблице** |
| --- | --- | --- | --- | --- |
| FK\_buses\_model\_id | model\_buses | id | buses | model\_id |
| FK\_comments\_customer\_id | customers | id | comments | customer\_id |
| FK\_routes\_arrival\_city\_id | cities | id | routes | arrival\_city\_id |
| FK\_routes\_departure\_city\_id | cities | id | routes | departure\_city\_id |
| FK\_runs\_carrier\_id | carrier | id | runs | carrier\_id |
| FK\_runs\_bus\_id | buses | id | runs | bus\_id |
| FK\_runs\_route\_id | routes | id | runs | route\_id |
| FK\_seat\_runs\_seat\_id | seats | id | seat\_runs | seat\_id |
| FK\_seat\_runs\_run\_id | runs | id | seat\_runs | run\_id |
| FK\_seat\_runs\_customer\_id | customers | id | seat\_runs | customer\_id |
| FK\_seats\_bus\_id | buses | id | seats | bus\_id |
| FK\_tickets\_seat\_run\_id | seat\_runs | id | tickets | seat\_run\_id |
| FK\_tickets\_  customer\_id | customers | id | tickets | customer\_id |
| FK\_tickets\_carrier\_id | carriers | id | tickets | carrier\_id |

## Основные пользовательские функции, триггерные функции, триггеры.

В рамках предметной области и структуры реализации программного продукта было принято решение не создавать пользовательские функции в рамках СУБД PostgreSQL, а написать аналогичные по функционалу функции с помощью конструктора запросов Laravel. Это решение связано с тем, что функции на PHP могут быть переиспользованы в других проектах, не привязываясь к конкретной СУБД, а также при разработке функций прямо на уровне базы данных, вы можете столкнуться с ограничениями в управлении версиями и тестировании. Основные пользовательские функции и их назначение представлены в табл. 2.15.

Таблица 2.15 – Основные пользовательские функции клиентского приложения.

| **Название** | **Тип запроса** | **Параметры** | **Назначение** |
| --- | --- | --- | --- |
| auto\_run() | - | - | Функция автоматического изменения статуса рейса |
| login\_show() | GET | - | Вывод формы авторизации на экран |
| login\_post() | POST | Request $request | Аутентификация пользователя на сервере |
| profile() | GET | - | Вывод информации о пользователе на экран |
| run() | GET | - | Вывод интерфейса редактирования рейсов на экран |
| run\_post() | POST | Request $request | Добавление рейса в систему |
| register\_post() | POST | Request $request | Создание пользователя на сервере |
| booking\_id\_post() | POST | Request $request, $id | Бронирование билета на сервере |

## Использование массивов.

## Использование массивов не предусмотрено в данной предметной области.

## Использование JSON-объектов.

Использование JSON-объектов в базе данных не предусмотрено в данной предметной области. Однако JSON-объекты используются в рамках API(набор правил и соглашений, который определяет, как различные компоненты программного обеспечения могут взаимодействовать друг с другом. API является прослойкой между разными приложениями, позволяющей им обмениваться данными и функциональностью). Таким образом JSON-объекты используются для взаимодействия клиента и сервера. Ниже приведён пример JSON-объекта, благодаря которому осуществляется аутентификация пользователя в системе:

{

"email": "mail@mail.com",

"password": "1"

}

## 2.12. Созданные индексы базы данных.

В процессе создания базы данных в СУБД PostgreSQL были автоматически созданы индексы по первичным ключам всех таблиц.

## 2.13 Обеспечение безопасности базы данных

Для обеспечения безопасности системы приняты следующие меры:

* Все ключевые части программы содержат обработчики исключений, которые информируют пользователя о возникших ошибках, обеспечивая более понятное взаимодействие.
* Все пользователи системы обладают правами только на выполнение соответствующих хранимых процедур, что гарантирует ограниченный доступ и повышает уровень безопасности. Принятые меры безопасности распространяются как на клиентскую, так и на серверную части системы, обеспечивая надежную защиту данных и операций.

**2.14 План обслуживания и резервного копирования базы данных**

## План обслуживания базы данных PostgreSQL:

## Создание еженедельного бэкапа:

## Каждую неделю запускать операцию создания бэкапа базы данных PostgreSQL.

## Для создания бэкапа использовать инструмент управления базами данных PostgreSQL, например, pg\_dump.

## Сохранять бэкапы в безопасном месте с ограниченным доступом для обеспечения конфиденциальности.

## Перенос базы данных с одного сервера на другой:

## При необходимости переноса базы данных с одного сервера на другой, использовать созданные бэкапы.

## Перенос базы данных на другой сервер выполнять с помощью инструментов управления базами данных, таких как pg\_dump и pg\_restore.

## Убедиться, что все зависимости и конфигурации сохраняются и переносятся правильно.

## 2.15 Назначение основных классов системы

MVC (Model-View-Controller) включает в себя три основных компонента: Модель (Model), Представление (View) и Контроллер (Controller). Модель отвечает за работу с данными, Представление за отображение данных пользователю, а Контроллер обрабатывает запросы от пользователя, взаимодействует с моделью для получения данных и передает их в представление. Основные принципы MVC включают разделение ответственностей, модульность и удобство разработки. Использование паттерна MVC в Laravel помогает создавать структурированные, легко поддерживаемые и расширяемые веб-приложения.

В Laravel взаимодействие с базой данных осуществляется через модели (Models). Модель представляет отдельную таблицу в базе данных и определяет структуру данных, а также содержит логику для работы с этими данными.

В табл. 2.16 представлены модули, реализующие работу с базой данных.

Таблица 2.16 – Модули, реализующие работу с базой данных

| **Название модуля** | **Описание** |
| --- | --- |
| SiteController | Контроллер. Обеспечивает общую логику веб-приложения. |
| Bus | Модель. Реализует работу с таблицей buses |
| Carrier | Модель. Реализует работу с таблицей carriers |
| City | Модель. Реализует работу с таблицей cities |
| Comment | Модель. Реализует работу с таблицей comments |
| Customer | Модель. Реализует работу с таблицей customers |
| Modelbus | Модель. Реализует работу с таблицей model\_buses |
| Route | Модель. Реализует работу с таблицей routes |
| Run | Модель. Реализует работу с таблицей runs |
| Seat | Модель. Реализует работу с таблицей seats |
| SeatRun | Модель. Реализует работу с таблицей seat\_runs |
| Ticket | Модель. Реализует работу с таблицей tickets |

Все классы находятся в пространстве имен App\Models и App\Http\Controllers.

# ЗАКЛЮЧЕНИЕ

В процессе выполнения отчёта по практике проектированию были решены следующие задачи:

• изучена предметная область;

• разработана база данных;

• разработано программное обеспечение.

Разработанный программный продукт позволяет сократить бумажную работу и облегчить работу с бронированием.

Интерфейсы разработаны в соответствии с требованиями эргономики рабочего места пользователя информационной системы и интуитивно понятны.

Дальнейшее развитие информационной системы:

• учет большего количества свойств бронирования;

• разработка дизайна форм.

В процессе выполнения отчёта по практике получены практические навыки работы в среде Visual Studio Code и PhpStorm, проектировании структуры базы данных и ее реализации в СУБД PostgreSQL, программировании на языках JavaScript и PHP.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Левитин А. В. Алгоритмы. Введение в разработку и анализ — 2006. - С. - 576 с
2. Белов С.В., Лаптев В.В., Морозов А.В., Толасова В.В., Мамлеева А.Р. Требования к оформлению студенческих работ. / АГТУ – Астрахань, 2019. 60 с.
3. Алгоритмы и структуры данных — Кормен - 2022
4. Брайан Сиверсон и Джоэл Мурах «Murach’s SQL Server 2019 for Developers»Издательство: Gazelle Book Services (2020)
5. Интернет-портал Консультант Плюс
6. Изучаем SQL. Алан Бьюли (2019) Издательство: Литрес
7. “Грокаем алгоритмы. Иллюстрированное пособие для программистов и любопытствующих” - Адитья Бхаргава(2022) Издательство: Литрес
8. “Чистый код. Создание, анализ и рефакторинг” Мартин Роберт С.Издательство: Питер (2022)
9. Тернстрем Т. Microsoft SQL Server 2008. Разработка баз данных. Учебный курс. 2012.
10. Жилинский А.А. Самоучитель Microsoft SQL Server 2008. СПб.:БХВ-Петербург. 2013

.

**ПРИЛОЖЕНИЕ**

**ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ К СИСТЕМЕ**

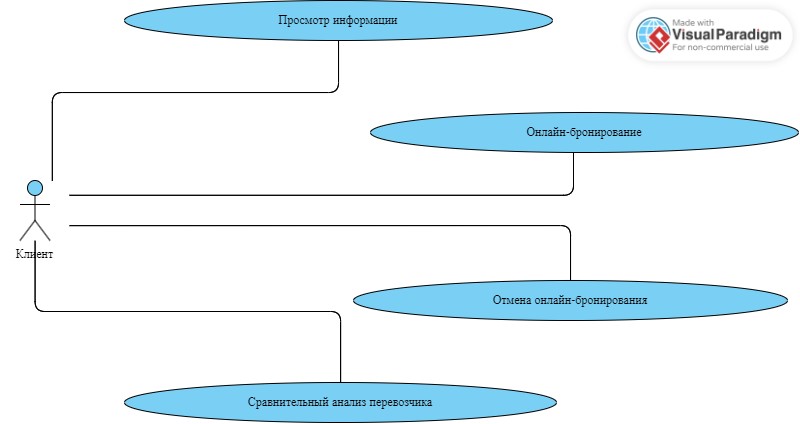


Рисунок А.1. – Диаграмма вариантов использования. Роль — Клиент.

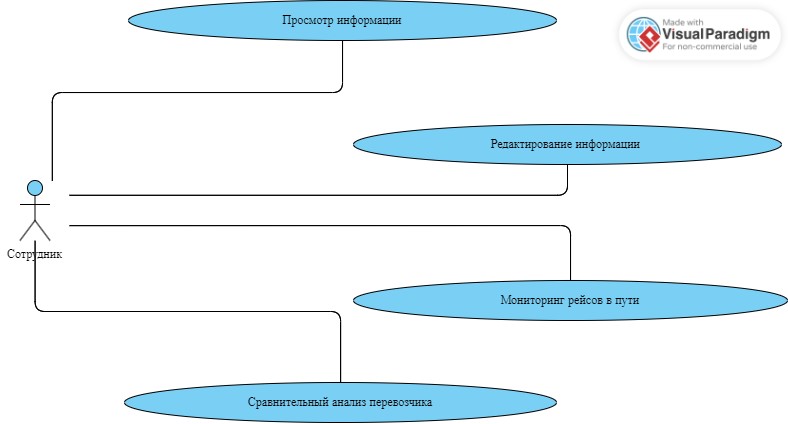


Рисунок А.2. – Диаграмма вариантов использования .Роль — Сотрудник.

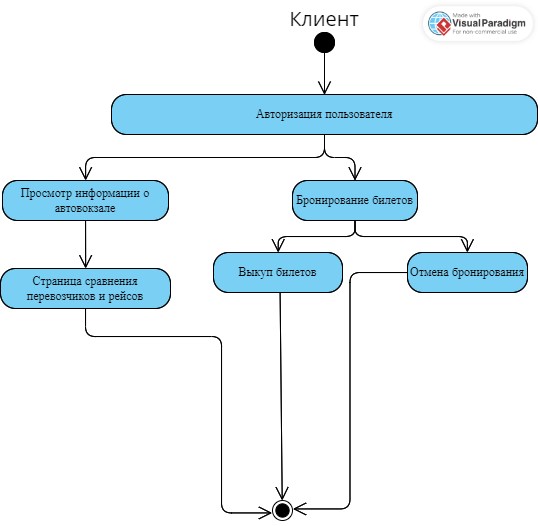


Рисунок А.3. – Диаграмма активности пользователя. Роль – Клиент

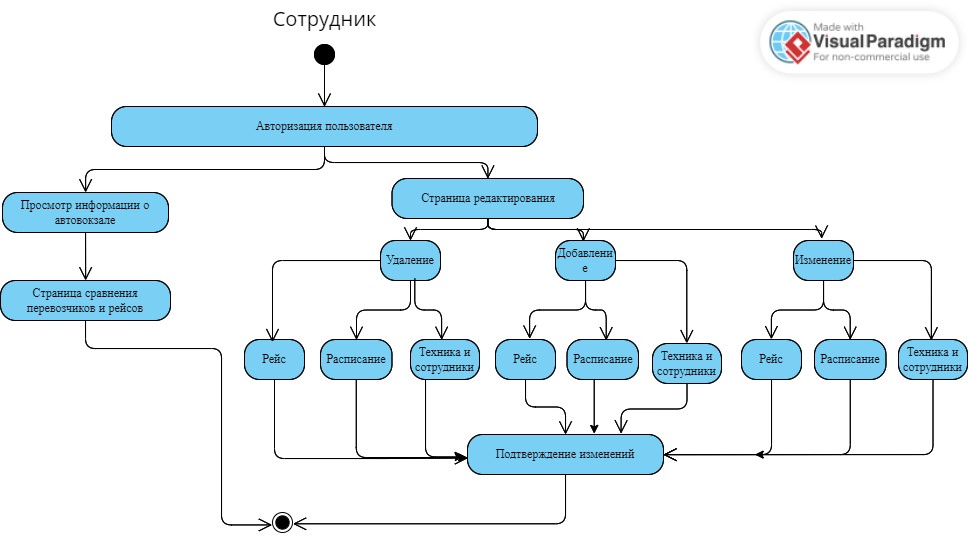


Рисунок А.4. –Диаграмма активности пользователя. Роль - Сотрудник

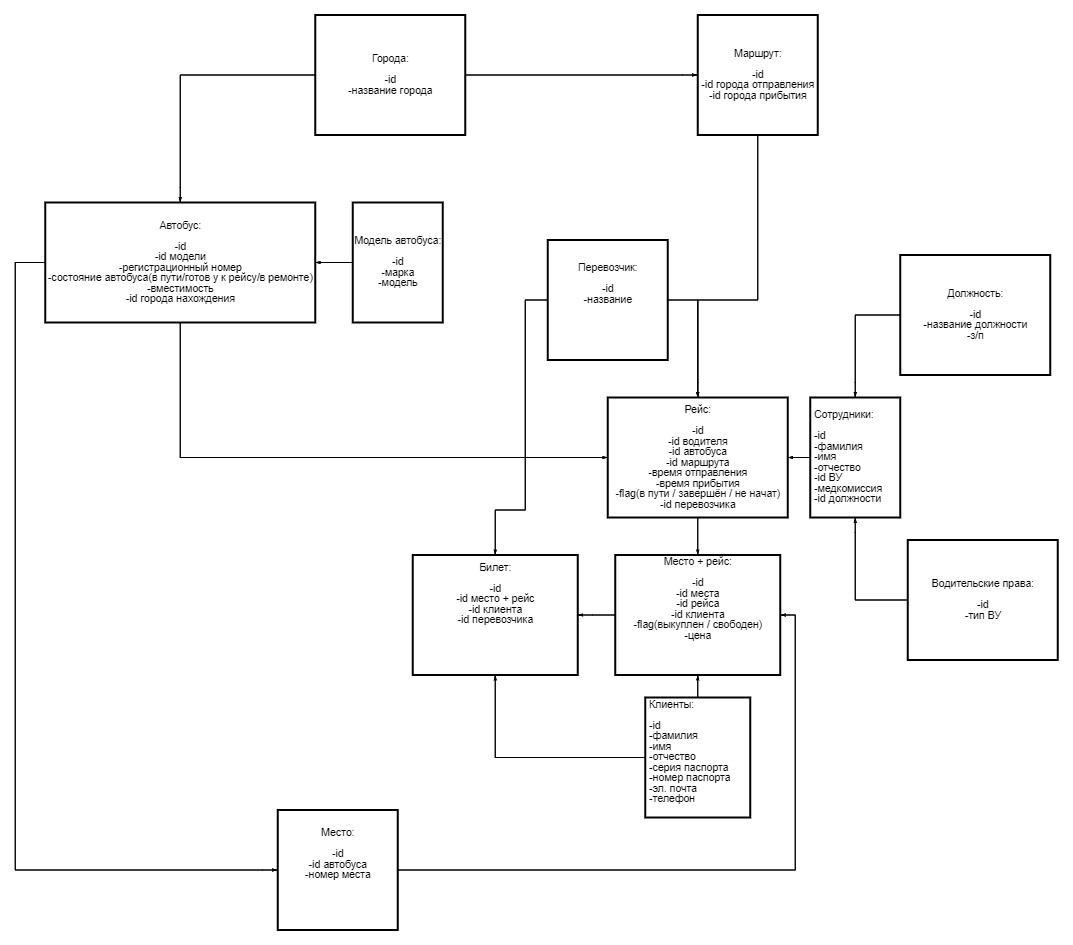


Рисунок А.5. – Диаграмма классов

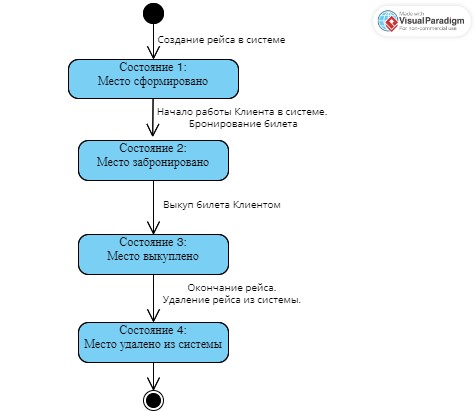


Рисунок А.6. – Диаграмма состояния сущности Место в системе.