**Содержание**

[Введение 2](#_Toc149513739)

[1. Анализ требований к программного средству 3](#_Toc149513740)

[1.1 Аналитический обзор аналогов 4](#_Toc149513741)

[1.1.1 Программное средство «taxi.yandex.by» 4](#_Toc149513742)

[1.1.2 Программное средство «135.by» 5](#_Toc149513743)

[1.1.3 Программное средство «taximaxim.by» 6](#_Toc149513744)

[1.2 Вывод анализа требований 6](#_Toc149513745)

[2. Проектирование базы данных 7](#_Toc149513747)

[2.1 Описание проектирования базы данных 7](#_Toc149513748)

[2.1 Вывод по проектировании модели базы данных 8](#_Toc149513749)

[3. Разработка объектов базы данных 9](#_Toc149513750)

[3.1 Разработка таблиц базы данных 9](#_Toc149513751)

[3.2 Индексы 11](#_Toc149513752)

[3.3 Процедуры 11](#_Toc149513753)

[3.4 Триггеры 12](#_Toc149513754)

[3.5 Пользователи 13](#_Toc149513755)

[3.6 Вывод по разработке объектов базы данных 14](#_Toc149513756)

[4. Руководство пользователя 15](#_Toc149513757)

[4.2 Вывод об руководстве пользователя 21](#_Toc149513758)

# **Введение**

Современный мир ценит скорость, надежность и удобство в любых аспектах деятельности. Транспортное обслуживание, и в частности сфера такси, не является исключением. Успешное функционирование таксопарка во многом зависит от качества программного обеспечения, которое помогает автоматизировать процессы, связанные с обслуживанием клиентов, управлением автопарком, а также отчетностью и аналитикой.

Технология Windows Presentation Foundation (WPF) на языке программирования C# открывает широкие возможности для разработки современных, функциональных и визуально привлекательных приложений. WPF позволяет создавать графически сложные интерфейсы, реализовывать анимацию, трехмерную графику и многие другие функции, которые делают работу с программой комфортной и интуитивно понятной для конечного пользователя.

Задачей данного курсового проекта является разработка программного решения для управления таксопарком с использованием технологии WPF на C#. Основные аспекты проекта включают в себя проектирование интуитивного пользовательского интерфейса, разработку функционала для учета водителей и автомобилей, реализацию системы заказа такси, а также инструменты для аналитики и отчетности.

Цель проекта – создание надежного, многофункционального приложения, которое будет соответствовать потребностям современных таксопарков и предоставлять все необходимые инструменты для эффективного управления бизнесом в этой сфере.

В результате выполнения проекта будет представлено готовое приложение для таксопарка, учитывающее современные тренды в области дизайна и программирования, а также специфику бизнеса в сфере такси. Это позволит получить ценный опыт в разработке комплексных программных решений на базе WPF и C#.

1. **Анализ требований к программного средству**

Функции администратора таксопарка:

* Администрирование базы данных водителей: Администратор должен иметь возможность управлять информацией о водителях, что включает в себя добавление новых водителей, редактирование информации о существующих водителях и удаление устаревших данных. Это позволит поддерживать актуальную базу водителей таксопарка;
* администрирование базы данных автомобилей: Администратор должен иметь права на добавление, редактирование и удаление данных автомобилей. Это обеспечивает корректное отображение состояния автопарка;
* управление статистикой таксопарка: Администратору необходим инструментарий для просмотра и анализа статистической информации, что способствует более обоснованным управленческим решениям;
* редактирование стоимости за км поездки: Это ключевой аспект управления таксопарком, позволяющий гибко устанавливать тарифы в соответствии с экономической обстановкой.

Функции клиента/пользователя:

* Регистрация и авторизация: Пользователи должны иметь возможность создавать аккаунт и входить в систему, чтобы воспользоваться услугами такси и просмотреть историю своих заказов;
* заказ такси: Клиентам предоставляется функционал для заказа такси двумя способами: указав адрес или выбрав местоположение на интерактивной карте;
* просмотр истории заказов: Пользователи должны иметь доступ к истории своих заказов, чтобы контролировать свои поездки и расходы;
* редактирование своего профиля: Клиентам предоставляется возможность редактировать свои личные данные, в том числе контактную информацию, для удобства использования сервиса.

Таким образом, разрабатываемое программное средство должно предоставлять интуитивно понятный и функциональный интерфейс, обеспечивающий эффективное управление таксопарком для администраторов и удобное взаимодействие с сервисом для клиентов.

## **Аналитический обзор аналогов**

В наши дни множество подобных программ можно найти на сайтах с тематикой по данному профилю. Перед тем как приступить к работе требовалось провести анализ и прочитать соответствующие статьи по данной теме. В ходе поиска данной литературы мною были найдены прототипы компаний таксопарков.

1. **Программное средство «taxi.yandex.by»**

На рисунке 1.1 предствлен скриншот веб-приложения «taxi.yandex.by».

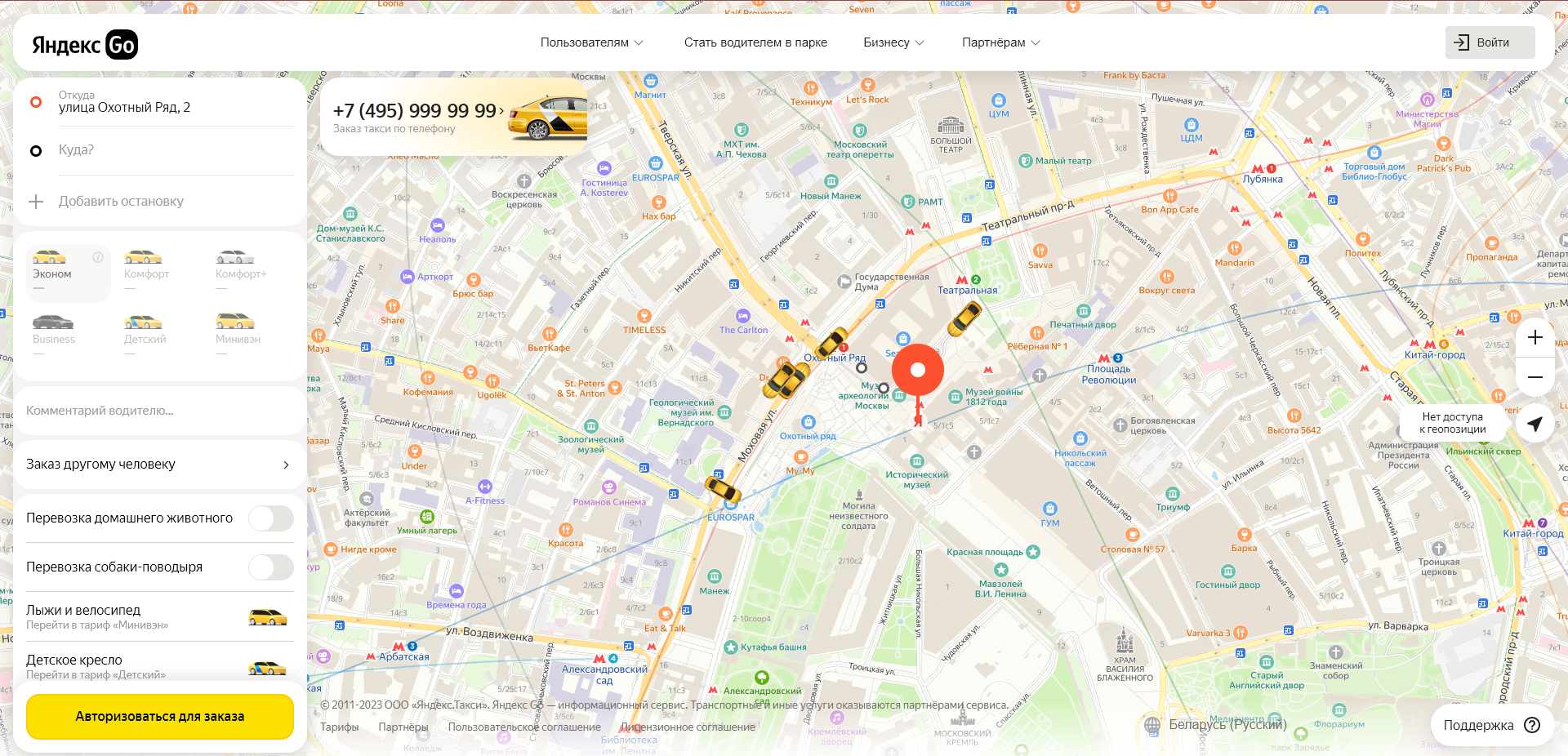


Рисунок 1.1 – Скриншот аналога веб-приложения «taxi.yandex.by»

На этом сайте можно заказать необходимую машину по введенным пользователем данным. У каждого пользователя есть доступ к личному кабинету, в котором он может просмотреть историю поездок и информацию аккаунта. [1]

Проанализировав «taxi.yandex.by», можно выделить некоторые плюсы и минусы программного средства.

Основные плюсы:

* возможность выбрать точку подачи с помощью карты;
* возможность выбрать предпочитаемый тариф поездки;
* возможность просмотреть историю поездок.

Основные минусы:

* много пустого пространства на панели меню;
* нет возможности выбрать предпочитаемую машину;
* слишком светлые тона.

1. **Программное средство «135.by»**

На рисунке 1.2 предствлен скриншот веб-приложения «135.by».

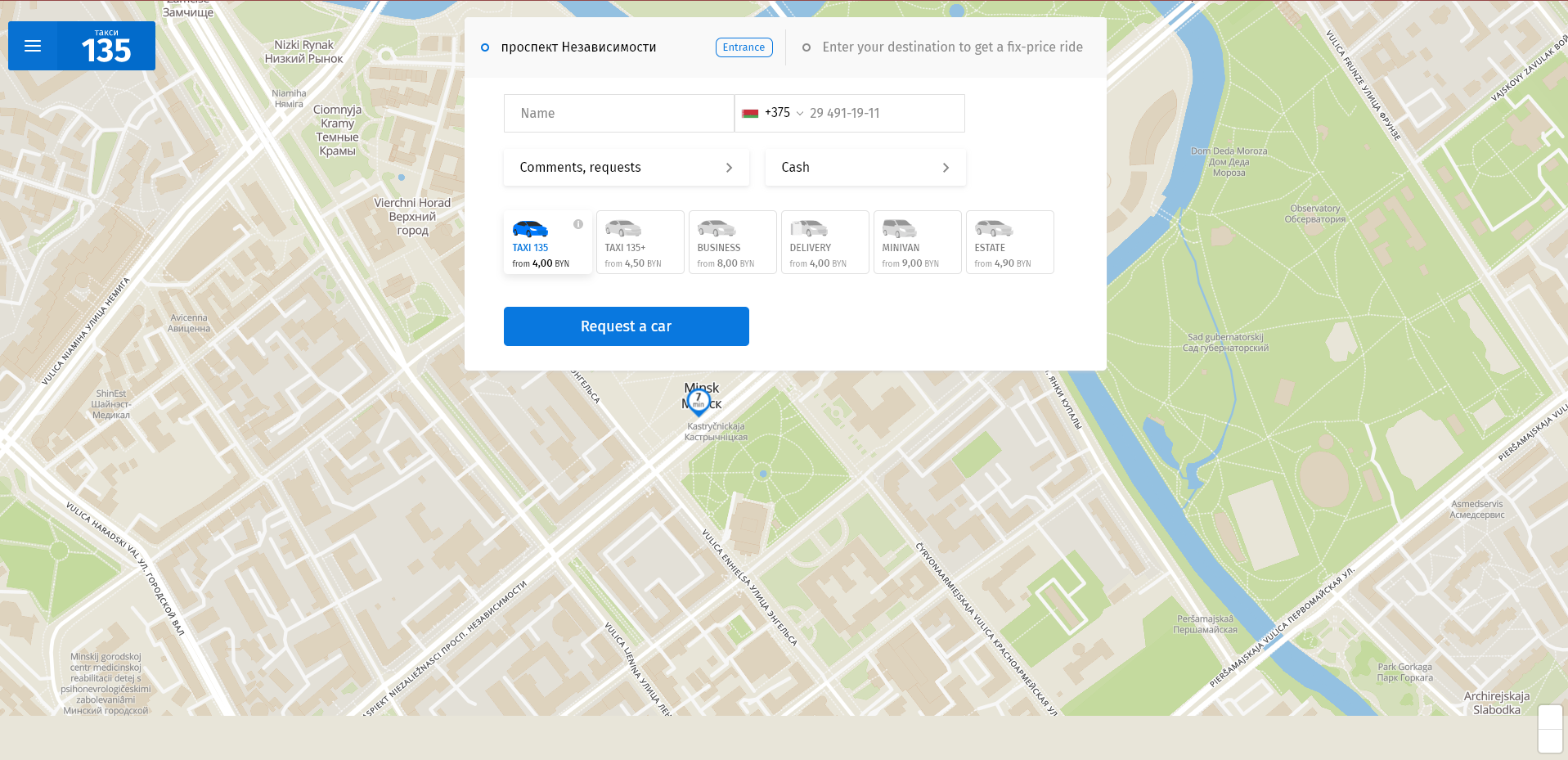


Рисунок 1.2 – Скриншот аналога веб-приложения «135.by»

«135.by» — приложение, похожее на «taxi.yandex.by». На данном сайте можно заказать такси используя поля для ввода адреса. Нету возможности завести личный профиль пользователя. Присутствует карта на заднем фоне, которая покажет маршрут поездки. [2]

Проанализировав «135.by», можно выделить некоторые плюсы и минусы программного средства.

Основные плюсы приложения:

* возможность просмотреть маршрут поездки;
* возможность просмотреть историю поездок;
* возможность выбора тарифа поездки;
* отзывчивый интерфейс.

Основные минусы:

* отсутствует возможность выбора точки подачи на карте;
* отсутствие возможности оставить отзыв о поездке;
* отсутствие личного профиля пользователя.

1. **Программное средство «taximaxim.by»**

На рисунке 1.3 представлен скриншот веб-приложения «taximaxim.by».

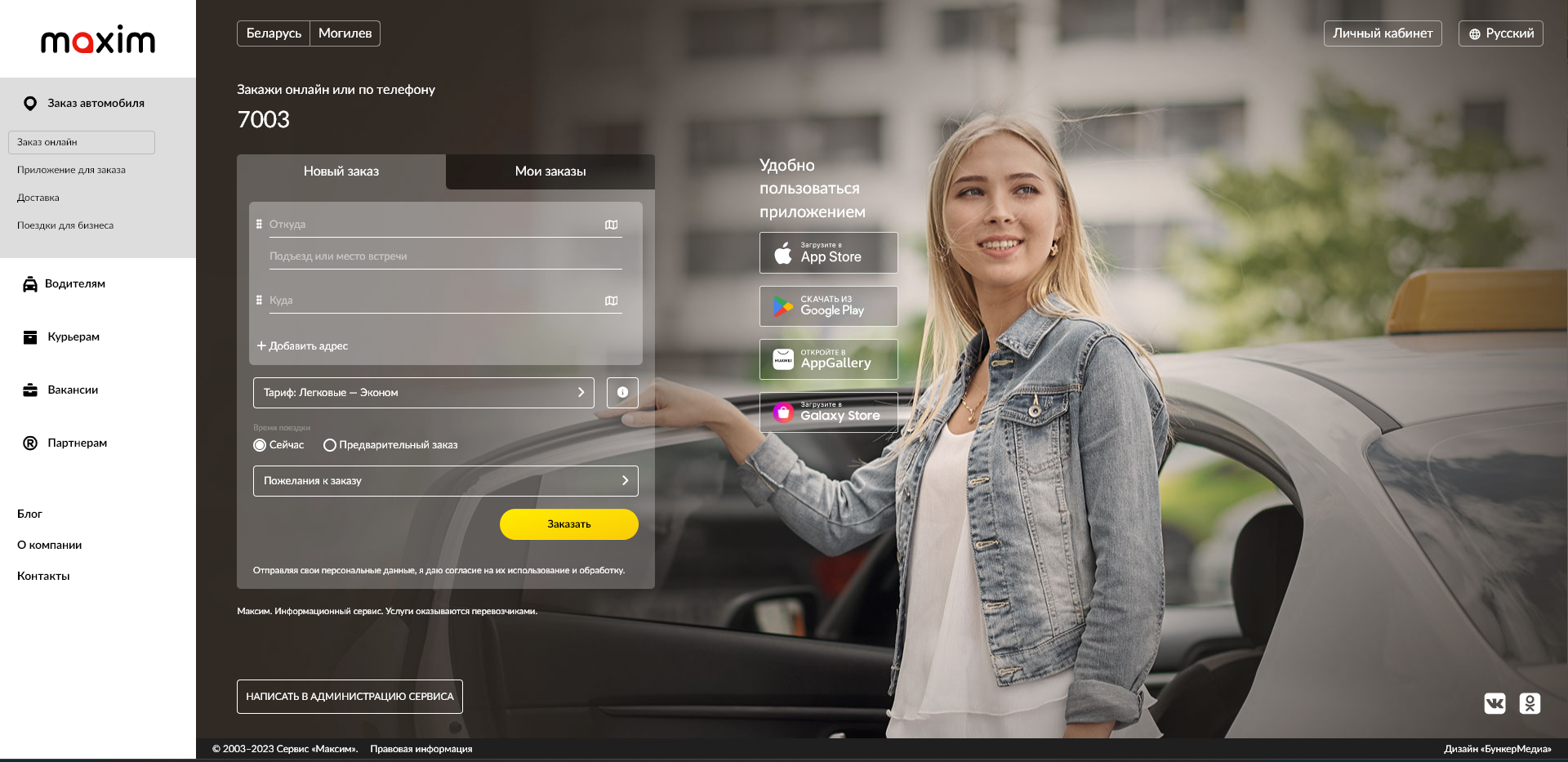


Рисунок 1.3 – Скриншот аналога веб-приложения «taximaxim.by»

На этом сайте можно заказать такси по введенным пользователем данным. Пользовать может завести свой личный аккаунт и просматривать историю своих заказов. Отсутствует функциональная карта, следовательно нельзя будет выбрать точку подачи и просмотреть сам маршрут поездки. [3]

Проанализировав «taximaxim.by», можно выделить некоторые плюсы и минусы программного средства.

Основные плюсы:

* отзывчивый интерфейс;
* возможность зарегистрировать аккаунт пользователя;
* возможность просмотреть историю поездок.

Основные минусы:

* отсутствие функциональной карты;
* низкое качество фонового фота на главной странице.
  1. **Вывод анализа требований**

При анализе трех ведущих веб-приложений для такси были выявлены их уникальные функции, а также плюсы и минусы. Ни одно из приложений не является идеальным: «taxi.yandex.by» имеет сложный дизайн, «135.by» не предлагает личного профиля пользователя, а «taximaxim.by» не имеет функциональной карты. На основе этих выводов, был сформирован вывод основанный на необходимости наличия функциональной карты в приложении, отзывчивого интерфейса и личного кабинета пользователя.

1. **Анализ требований к программному средству и разработка функциональных требований**

Прежде чем приступить к формулировке функциональных требований, проведем теоретический анализ предметной области - системы управления таксопарком. Рассмотрим основные принципы взаимодействия администратора и клиента с программным средством.

Важным аспектом разработки таксопарка является математическое обоснование ключевых операций, таких как расчет стоимости поездки, управление данными водителей и автомобилей, а также анализ статистики. Это включает в себя формулы для расчета стоимости поездки, алгоритмы управления данными и другие математические аспекты.

На основе теоретического анализа и математического обоснования проведем моделирование технических объектов системы. Это включает в себя диаграммы классов, диаграммы последовательности и другие модели, отражающие взаимодействие между различными компонентами системы.

Рассмотрим ключевые функции программного средства, обеспечивающие взаимодействие администратора и клиента с таксопарком.

Функции администратора:

* Добавление водителя. Администратор может добавлять новых водителей в систему, предоставляя необходимую информацию.
* Удаление водителя. Возможность удаления водителя из базы данных системы.
* Редактирование водителя. Возможность внесение изменений в профиль существующего водителя.
* Добавление автомобиля. Возможность регистрации нового автомобиля в системе таксопарка.
* Удаление автомобиля. Администратор имеет право удалять информацию о существующих в базе автомобилях.
* Редактирование автомобиля. Возможность внесение изменений в информацию о уже существующем автомобиле.
* Просмотр статистики. Доступ к обобщенным данных и статистике по работе таксопарка.
* Изменение стоимости за километр поездки. Администратор может устанавливать новые тарифы для расчета стоимости поездок.

Функции клиента:

* Регистрация и авторизация. Возможность создания нового аккаунта и входа в систему для оформления заказов.
* Заказ такси через функциональную карту. Клиент может выбирать метро отправления и место назначения на карте для заказа такси.
* Заказ такси через поле адреса. Возможность ввода точного адреса для заказа такси.
* Просмотр истории заказов. Клиент может просматривать историю своих предыдущих заказов.
* Оставление отзывов. Клиент имеет возможность оставлять отзывы о выполненных поездах.

Диаграмма вариантов использования для администратора должна включать в себя действия связанные с контролем самого приложения и иметь наивысшие уровни действия над другими. Пользователь или гость имеет ограничение по действиям связанные с конкретной тематикой приложения.

Диаграммы вариантов использования для администратора, пользователя, гостя можно увидеть на рисунке 2.2.

Отдельной роли для водителя отсутствует. Все управление водителями, автомобилями принадлежит администратору.

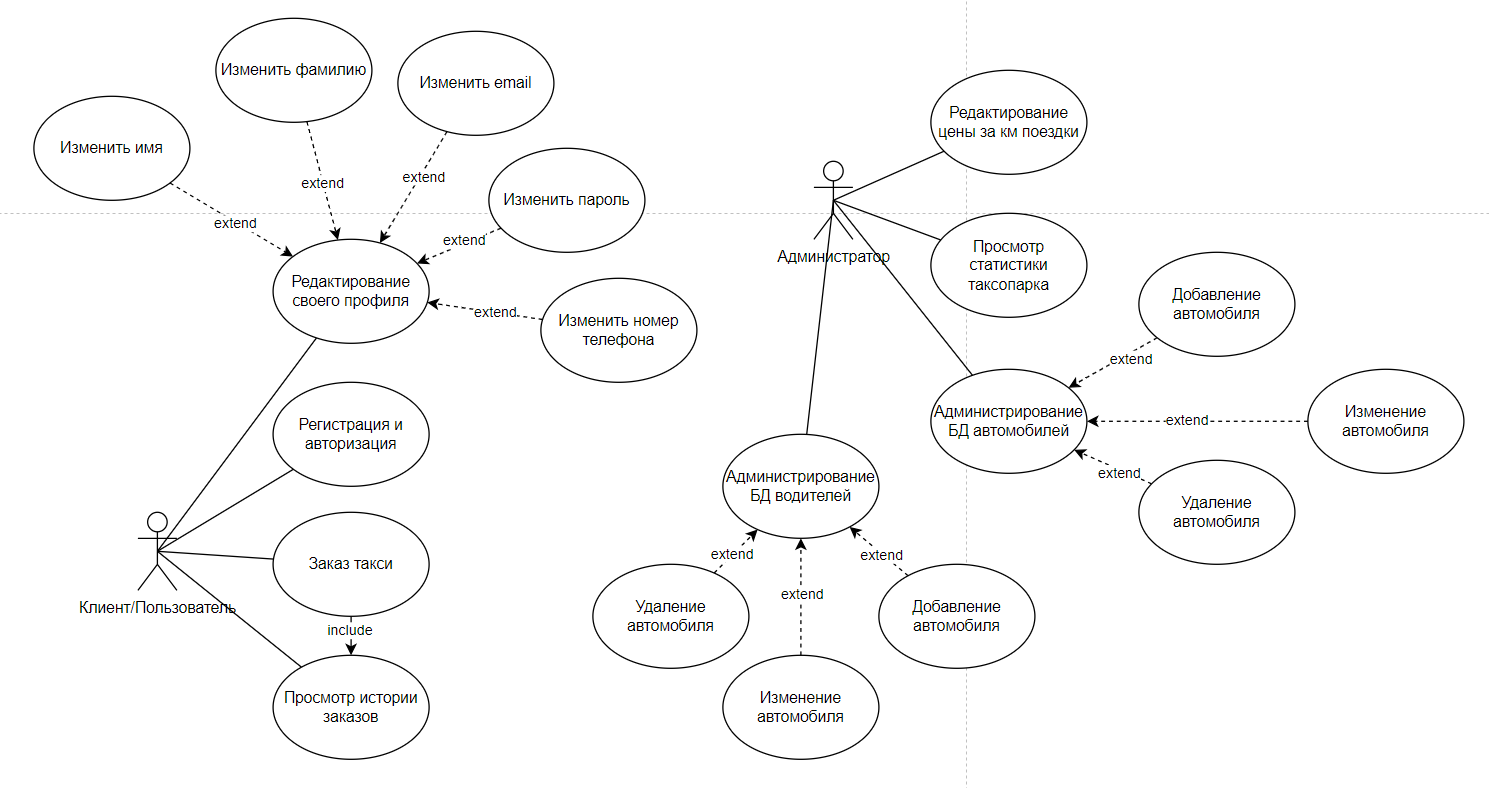


Рисунок 2.2 – Диаграмма вариантов использования

## **Вывод по проектировании модели базы данных**

В результате была спроектирована необходимая модель базы данных на основе диаграммы вариантов использования. Была определена информация, которая будет хранится в таблицах.

# **Проектирование программного средства**

1. **Проектирование базы данных**

Первым этапом курсового проекта будет проектирование логически взаимосвязанных таблиц. Чтобы составить визуальную взаимосвязанную структуру базы данных, нам необходимо продумать, какая информация будет храниться в этих таблицах, после этого спроектировать связи с помощью первичных и внешних ключей.

Диаграмма базы данных, спроектированной в ходе разработки показана на рисунке 2.1.

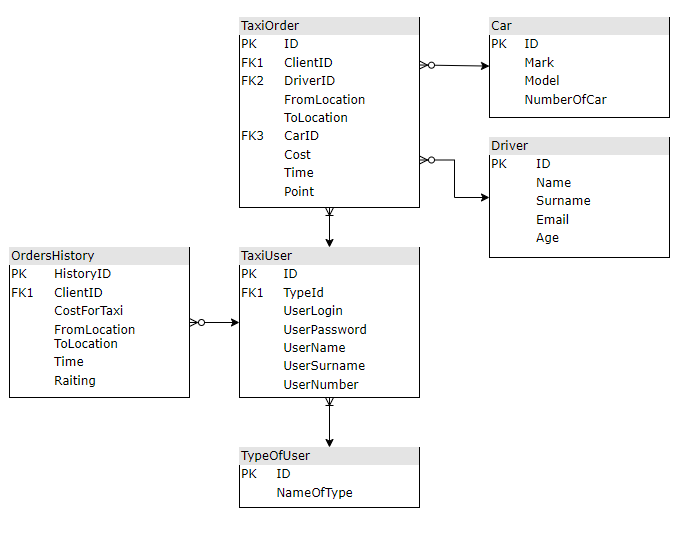


Рисунок 2.1 – Диаграмма базы данных

База данных, основанная на реляционной модели данных. В ее структуру входят следующие таблицы: TypeOfUser, TaxiUser, OrdersHistory,\_TaxiOrder, Driver, Car. Таблицы хранят в себе данные необходимые для описания и представления возможности заказа такси. А также информацию о самих клиентах и администраторе. База данных и таблицы были спроектированы на основе диаграммы вариантов использования, разработанной для гостя, клиента и администратора.

При разработке приложения для курсового проекта была использована база данных Oracle.

В базе данных представлены 6 таблиц: TypeOfUser, TaxiUser, OrdersHistory, TaxiOrder, Driver, Car.

Таблица TypeOfUser хранит типы пользователей, состоит из столбцов (таблица 3.1):

Таблица 3.1 – Столбцы таблицы TypeOfUser

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| Typeid | Идентификатор типа, первичный ключ | Number |
| Nameoftype | Имя типа | Nvarchar2(50) |

Таблица TaxiUser хранит список пользователей и их персональные данные, состоит из столбцов (таблица 3.2):

Таблица 3.2 – Столбцы таблицы TaxiUser

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| Id | Идентификатор пользователя, первичный ключ | Number |
| Typeid | Идентификатор типа, внешний ключ | Number |
| Userlogin | Логин пользователя | Nvarchar2(50) |
| Userpassword | Пароль пользователя | Nvarchar2(50) |
| Username | Имя пользователя | Nvarchar2(50) |
| Usersurname | Фамилия пользователя | nvarchar2(50) |
| Usernumber | Номер телефона пользователя | Nvarchar2(50) |

Таблица OrdersHistory хранит историю заказов пользователя, таблица состоит из столбцов (таблица 3.3):

Таблица 3.3 – Столбцы таблицы OrdersHistory

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| Historyid | Идентификатор истории, первичный ключ | Number |
| Clientid | Идентификатор пользователя, внешний ключ | Number |
| Costfortaxi | Стоимость поездки | Number |
| Tolocation | Точка подачи | Nvarchar2(50) |
| Fromlocation | Точка завершения поездки | Nvarchar2(50) |
| Time | Время начала поездки | Timestamp(2) |
| Raiting | Оценка | Number |

Таблица TaxiOrder представляет список заказов такси, состоит из столбцов (таблица 3.4):

Таблица 3.4 – Столбцы таблицы Feedback\_flight

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| Id | Идентификатор заказа, первичный ключ | Number |
| Clientid | Идентификатор пользователя, внешний ключ | Number |
| Criverid | Идентификатор водителя, внешний ключ | Number |
| Carid | Идентификатор машины, внешний ключ | Number |
| Fromlocation | Точка подачи | Nvarchar2(50) |
| Tolocation | Точка завершения поездки | Nvarchar2(50) |
| Cost | Стоимость поездки | Number |
| Time | Время начала поездки | Timestamp(2) |
| Point | Оценка | Number |

Таблица Driver представляет список водителей таксопарка, состоит из столбцов (таблица 3.5):

Таблица 3.5 – Столбцы таблицы Orders

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| Id | Идентификатор водителя, первичный ключ | Number |
| Name | Имя водителя | Nvarchar2(50) |
| Surname | Фамилия водителя | Nvarchar2(50) |
| Email | Email водителя | Nvarchar2(50) |
| Age | Возраст водителя | Number |

Таблица Car представляет список машин таксопарка. Данная таблица состоит из столбцов (таблица 3.6):

Таблица 3.6 – Столбцы таблицы Flight

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| Id | Идентификатор машины, первичный ключ | Number |
| Mark | Марка машины | Nvarchar2(50) |
| Model | Модель машины | Nvarchar2(50) |
| Numberofcar | Номер машины | Number |

### **3.1.1 Индексы**

Индекс — объект базы данных, создаваемый с целью повышения производительности поиска данных. Таблицы в базе данных могут иметь большое количество строк, которые хранятся в произвольном порядке, и их поиск по заданному критерию путём последовательного просмотра таблицы строка за строкой может занимать много времени. В связи с этим был разработан индекс столбца costfortaxi таблицы ordershistory, код которого вы можете увидеть на рисунке 5.1.



Рисунок 3.1 – Индекс costfortaxi

### **3.1.2 Процедуры**

Хранимая процедура – это поименованный код на языке PL/SQL. Хранимая процедура может быть создана с помощью CREATE, изменена с помощью ALTER и удалена с помощью оператора DROP. Процедура может принимать входные и формировать выходные параметры. Результатом ее выполнения может быть целочисленное значение, либо один или более результирующих наборов, сформированных операторами SELECT. Вызов процедуры осуществляется оператором EXECUTE (EXEC). В хранимых процедурах допускается применение основных DDL, DML и TCL-операторов, курсоров, временных таблиц.

Листинг PL/SQL-кода для создания процедур находится в приложении Б.

Процедуры, разработанные в рамках курсового проекта:

* CheckUser – проверка пользователя на нахождении в БД.
* CheckAdmin – проверка администратора на нахождении в БД.
* InsertUser– добавление пользователя.
* InsertOrder– добавление заказа.
* DeleteUser– удаление пользователя.
* GetHistory– получение истории заказов пользователя.
* InsertDriver – добавление водителя.
* InsertCar – добавление машины.
* GetDrivers – получение списка водителей.
* GetCars – получение списка машин.
* DeleteDriver – удаление водителя.
* DeleteCar – удаление машины.
* GetProfit – получение полной прибыли таксопарка.
* GetStatistics – получение статистики таксопарка.
* UpdateDriver – изменение водителя.
* UpdateCar – изменение машины.
* GetCarsNames – получение марки и модели машин таксопарка.
* getDectyptedPassword – получение расшифрованного пароля пользователя.
* GetUser – получение пользователя по логину.

### **3.1.3 Триггеры**

Триггер является именованным модулем PL/SQL, который хранится в базе данных и может быть вызван повторно. Вы можете включать и отключать триггер, но не можете явно вызывать его. Когда триггер включен, база данных автоматически вызывает его - то есть триггер срабатывает - всякий раз, когда происходит событие которое вызывает триггер. Пока триггер отключен, он не срабатывает.

Вы создаете триггер с помощью оператора CREATE TRIGGER. Вы указываете инициирующее событие в терминах запускающих операторов и объекта, на который они действуют. Считается, что триггер создан или определен для объекта, который является либо таблицей, представлением, схемой или базой данных. По умолчанию триггер создается во включенном состоянии.

Для добавления заказов в историю заказов пользователя был разработан триггер add\_history.

Пример создания триггера приводится на рисунке 3.2.

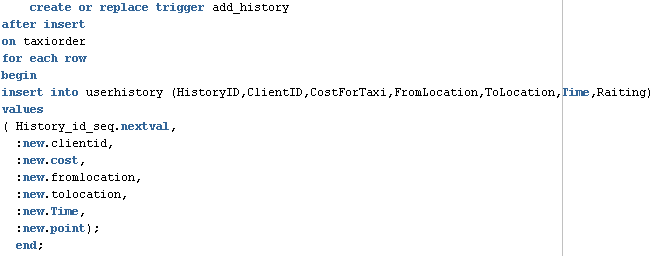


Рисунок 3.2 – Пример создания триггера add\_history

### **3.1.4 Пользователи**

Пользователь базы данных — это физическое или юридическое лицо, которое имеет доступ к БД и пользуется услугами информационной системы для получения информации. На каждом этапе развития БД (проектирование, реализация, эксплуатация, модернизация и развитие, полная реорганизация) с ней связаны разные категории пользователей.

При проектировании базы данных было создано 2 типа пользователей: клиент, осуществляющий доступные ему операции и администратор имеющий полный доступ ко всем объектам базы данных.

Создание пользователей представлено на рисунке 3.3.



Рисунок 3.3 – Создание пользователей «Клиент» и «Администратор» для базы данных

Далее для созданного пользователя базы данных были выданы необходимые привилегии необходимые для функционала клиента.

Частичная выдача привилегий клиенту представлено на рисунке 3.4.

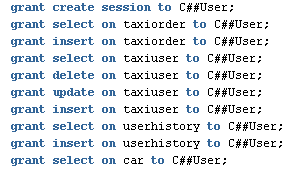


Рисунок 3.4 – Выдача привилегий клиенту

Далее для созданного пользователя базы данных были выданы необходимые привилегии необходимые для функционала администратора.

Выдача привилегий клиенту представлено на рисунке 3.5.

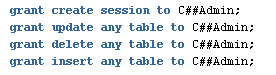


Рисунок 3.5 – Выдача привилегий администратору

# **Руководство пользователя**

* 1. **Описание руководства пользователя**

При запуске программы первой откроется страница входа в аккаунт. Она приведена на рисунке 4.1.

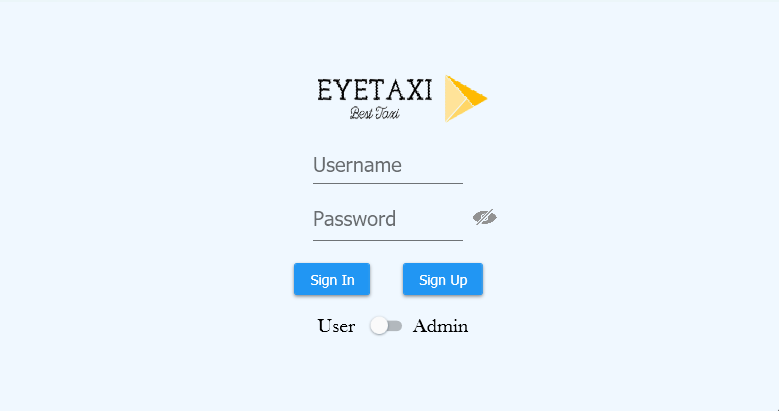


Рисунок 4.1 – Страница входа в аккаунт

Пользователь может зарегистрироваться в приложении выбрав соответствующую кнопку рядом с кнопкой входа. Страница регистрации приведена на рисунках 4.2.

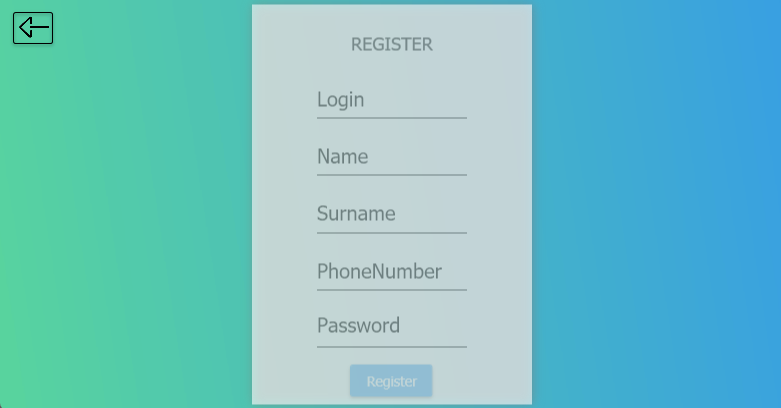


Рисунок 4.2 – Регистрация пользователя

При регистрации и авторизации происходит проверки по всем необходимым полям. Также осуществляется проверка на существование пользователя.

После авторизации в зависимости от роли пользователя предоставляются различные функции.

При авторизации в роли администратора видим страницу панели управления администратора. Страница панели управления администратора приведена на рисунке 4.4.

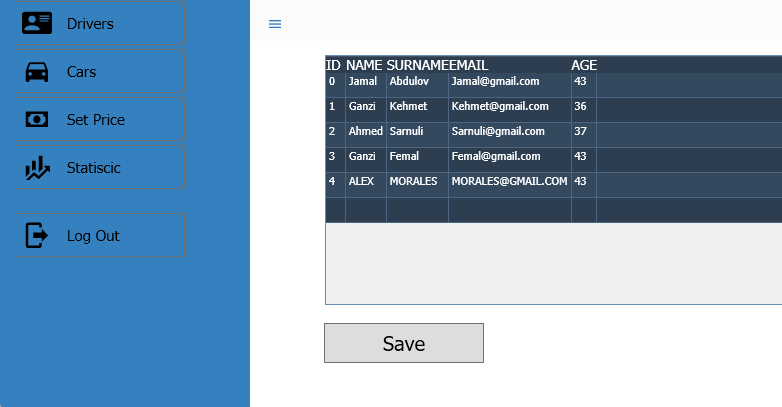


Рисунок 4.4 – Панель управления администратора

В панели управления администратору представляется большой функционал для администрирования таксопарка.

Администратор может посмотреть информацию о: водителях, автомобилях, статистике таксопарка. Страницы с реализацией перечисленного функционала представлены на рисунках 4.5 – 4.7.

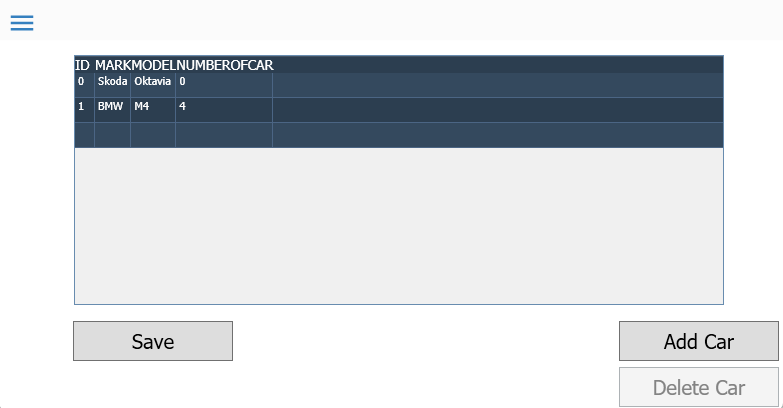


Рисунок 4.5 – Страница автомобилей

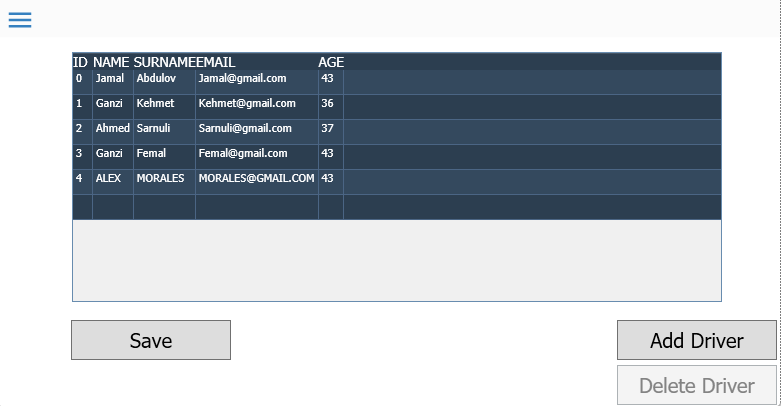


Рисунок 4.6 – Страница водителей

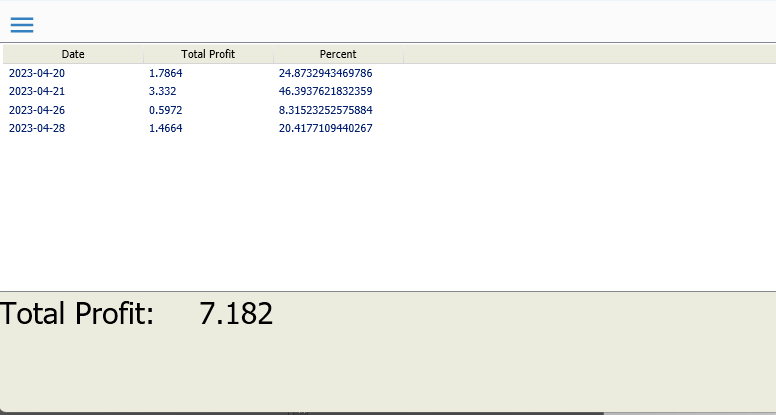


Рисунок 4.7 – Страница статистики

Основным функционалом администратора является: добавление автомобилей, добавление водителей, изменение стоимости поездки за километр. Страницы с реализацией перечисленного функционала представлены на рисунках 4.8 – 4.10.

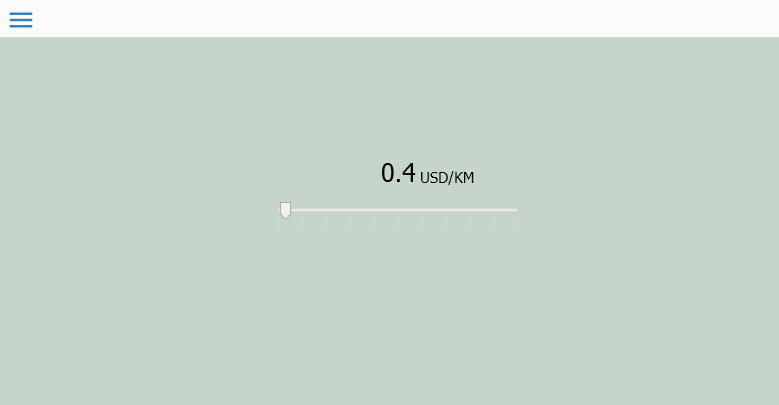


Рисунок 4.8 – Страница изменения стоимости поездки

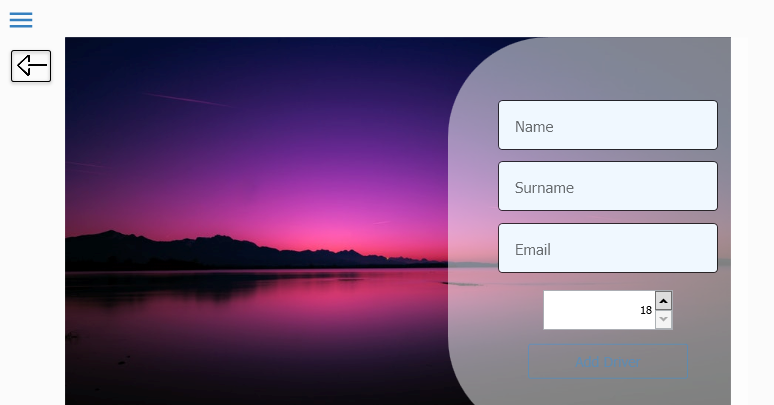


Рисунок 4.7 – Страница добавления водителя

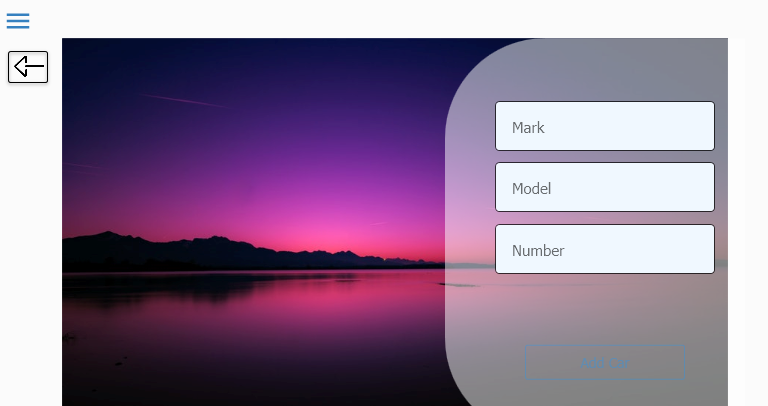


Рисунок 4.10 – Страница добавления автомобиля

При авторизации в роли пользователя видим главную страницу авторизованного пользователя. Страница пользователя приведена на рисунке 4.11.

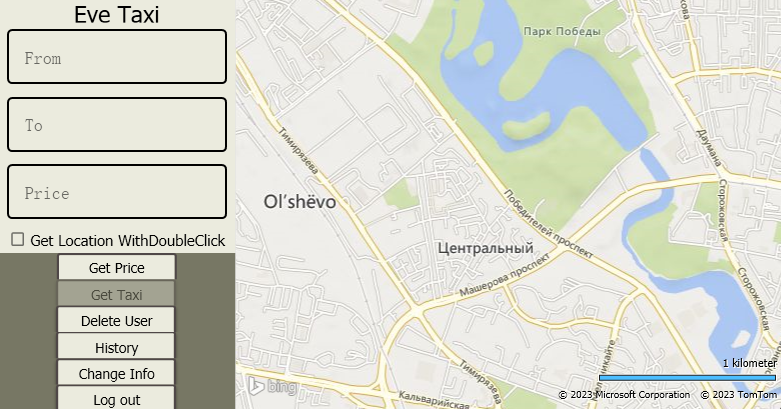


Рисунок 4.11 – Страница пользователя

Пример выбора окна, после выбора точек подачи и окончания поездки показан на рисунке 4.12.

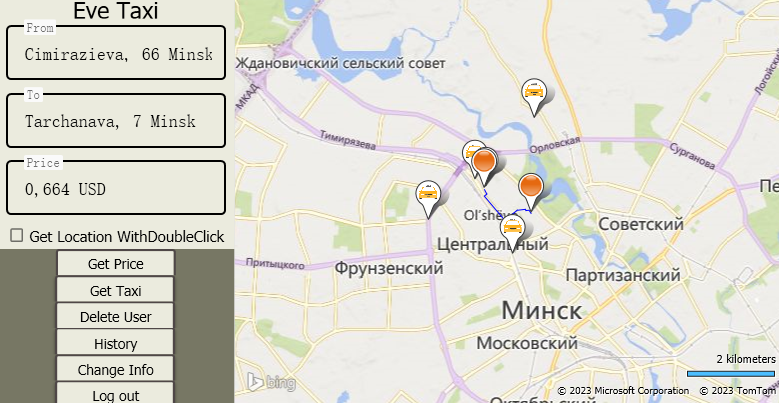


Рисунок 4.12 – Пример окна пользователя

При нажатии на кнопку заказа будет дана возможность выбора машины из доступных в таксопарке. Пример данного окна вы можете увидеть на рисунке 4.13.

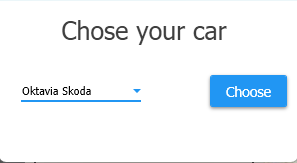


Рисунок 4.13 – Пример окна выбора автомобиля

После окончания поездки пользователю появится окно, в котором пользователю будет дана возможность поставить оценку за поездку. Данное окно вы можете увидеть на рисунке 7.14.



Рисунок 4.14 – Пример окна оценки за заказ

При нажатии на кнопку «Change Info» появится окно, в котором можно изменить данные пользователя. Пример данного окна показан на рисунке 4.15.

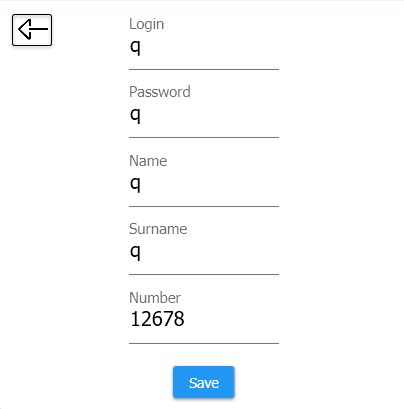


Рисунок 4.15 – Пример окна редактирования пользователя

При нажатии на кнопку истории можно увидеть историю заказов пользователя. Пример истории заказов пользователя представлена на рисунке 4.16.



Рисунок 4.16 – Поиск билета и предоставленная информация

Сам заказ такси происходит в 5 этапов. Первый этап заключается в заполнении полей адресов и нажатии кнопки получения стоимости заказа. Следующий этап заключается в нажатии кнопки заказа такси. Дальше появляется окно с выбором машины для поездки. После данных этапов машина едет к точке подачи, точка подачи пропадает и машина по правилам ПДД едет к точке окончания поездки. В конце заказа, когда машина прибыла к точке окончания выскакивает окно об оценке за поездку.

Так как тематика курсового проекта является «Таксопарк», то все возможные действия связанные с заказом и организацией такси были описаны выше. Главное для пользователя при заказе такси это удобный интерфейс представленный функциональной картой, личным аккаунтом пользователя. Всё данное было реализовано в курсовом проекте.

* 1. **Вывод об руководстве пользователя**

В данном разделе было описано руководство пользователя об приложении курсового проекта, такие как все окна приложения, действия и т.п.