|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | Министерство науки и высшего образования Российской Федерации  **Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования**  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

**ФАКУЛЬТЕТ** \_ ***ИУК «Информатика и управление»\_\_\_\_\_\_\_\_\_***

**КАФЕДРА** \_\_ ***ИУК5 «Системы обработки информации»***

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовой работе на тему:**

***реализация технологии Reflective DLL injection с помощью Windows API***

по дисциплине ***Системное программирование***

Студент гр.ИУК5-52Б \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_*Фролов К. Д.* \_\_\_)

(подпись) (Ф.И.О.)

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_*Фролов П. В.* \_\_\_)

(подпись) (Ф.И.О.)

Оценка руководителя \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_\_

30-50 (дата)

Оценка защиты \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_\_

30-50 (дата)

Оценка проекта \_\_\_\_\_ баллов \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка по пятибалльной шкале)

Комиссия: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(подпись) (Ф.И.О.)

Калуга, 2021

**Калужский филиал   
федерального государственного бюджетного образовательного учреждения высшего образования**

***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»   
(КФ МГТУ им. Н.Э. Баумана)***

УТВЕРЖДАЮ

Заведующий кафедрой **\_\_ИУК5\_\_\_**

\_\_\_\_\_\_\_\_\_\_\_(Е.В. Вершинин)

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине ***Базы данных***

Студент\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, инициалы, индекс группы)

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, инициалы)

График выполнения проекта: 25% к\_4\_нед., 50% к\_7\_нед., 75% к\_10\_нед., 100% к\_14\_нед.

***1. Тема курсового проекта***

***реализация технологии Reflective DLL injection с помощью Windows API***

***2. Техническое задание***

Разработать приложение с использованием функций Windows API для внедрения кода в целевой процесс\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***3. Оформление курсового проекта***

3.1. Расчетно-пояснительная записка на\_\_\_\_\_\_\_\_ листах формата А4.

3.2. Перечень графического материала КП (плакаты, схемы, чертежи и т.п.)\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г.

Руководитель курсового проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_ \_Фролов П. В.\_\_\_\_/

(подпись) (Ф.И.О.)

Задание получил\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_ Фролов К. Д.\_\_\_\_\_\_/ «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_2021г.

(подпись) (Ф.И.О.

СОДЕРЖАНИЕ

[1. ТЕХНИЧЕСКОЕ ЗАДАНИЕ 5](#_Toc89651231)

[1.1 Наименование 5](#_Toc89651232)

[1.2 Общие сведения 5](#_Toc89651233)

[1.2.1 Наименования предприятий разработчика и заказчика 5](#_Toc89651234)

[1.2.2 Перечень документов, на основании которых создается система 5](#_Toc89651235)

[1.2.3 Плановые сроки начала и окончания работы по созданию системы 5](#_Toc89651236)

[1.1.4 Сведения об источниках и порядке финансирования работ 5](#_Toc89651237)

[1.3 Назначение и цели создания (развития) системы 5](#_Toc89651238)

[1.3.1 Назначение системы 5](#_Toc89651239)

[1.3.2 Цели создания системы 5](#_Toc89651240)

[1.4 Характеристики объекта автоматизации 5](#_Toc89651241)

[1.5 Требования к системе 5](#_Toc89651242)

[1.5.1 Требования к системе. 5](#_Toc89651243)

[1.6 Состав и содержание работ по созданию системы 5](#_Toc89651244)

[1.7 Порядок контроля и приемки системы. 5](#_Toc89651245)

[1.7.1 Состав, объем и методы испытаний системы и ее составных частей. 5](#_Toc89651246)

[1.7.2 Общие требования к приемке работ. 5](#_Toc89651247)

[1.8 Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие. 5](#_Toc89651248)

[1.9 Требования к документированию 5](#_Toc89651249)

[1.10 Источники разработки 5](#_Toc89651250)

[2. НАУЧНО – ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ 5](#_Toc89651251)

[2.1 Постановка задачи проектирования 5](#_Toc89651252)

[2.2 Описание предметной области. 5](#_Toc89651253)

[2.3 Анализ аналогов. 5](#_Toc89651254)

[2.3.1 «АвтоСалон» 5](#_Toc89651255)

[2.3.2 «Альфа-Авто» 5](#_Toc89651256)

[2.4 Перечень задач, подлежащих решению в процессе разработки. 5](#_Toc89651257)

[2.5 Обоснование выбора инструментов и платформы для разработки. 5](#_Toc89651258)

[3. ПРОЕКТНО – КОНСТРУКТОРСКАЯ ЧАСТЬ 5](#_Toc89651259)

[3.1 Разработка структуры приложения. 5](#_Toc89651260)

[3.2 Разработка алгоритмов обработки информации. 5](#_Toc89651261)

[3.2.1 Алгоритм подсчета ежемесячного платежа 5](#_Toc89651262)

[3.2.2 Алгоритм создания Word документа 5](#_Toc89651263)

[3.3 Логическая схема базы данных 5](#_Toc89651264)

[3.4 Разработка интерфейса взаимодействия пользователя с системой. 5](#_Toc89651265)

[3.5 Разработка архитектуры приложения. 5](#_Toc89651266)

[3.6 Реализация функционирующего приложения. 5](#_Toc89651267)

[4. ПРОЕКТНО – ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ 5](#_Toc89651268)

[4.1 Проектирование начального и тестового наполнения базы данных. Процедуры автоматизации. 5](#_Toc89651269)

[4.2 Технологические решения, поддерживающие эксплуатационный цикл программы. 5](#_Toc89651270)

[4.3 Тестирование и отладка рабочей программы. 5](#_Toc89651271)

[5. ЗАКЛЮЧЕНИЕ 5](#_Toc89651272)

[6. СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТУРАТУРЫ 5](#_Toc89651273)

[ПРИЛОЖЕНИЕ 1 5](#_Toc89651274)

# ТЕХНИЧЕСКОЕ ЗАДАНИЕ

## 1.1 Наименование

Реализация корпоративного приложения, для сотрудников автосалона, в котором предусмотрено создание заявки на покупку автомобиля и быстрого просмотра информации об имеющихся автомобилях или аксессуаров к ним

## 1.2 Общие сведения

### 1.2.1 Наименования предприятий разработчика и заказчика

Разработчик: студент группы ИУК5-52Б Фролов К.Д.

Заказчик: КФ МГТУ им. Баумана кафедра ИУК5.

### 1.2.2 Перечень документов, на основании которых создается система

Методические указания к выполнению курсовой работы по предмету “Базы данных”

### 1.2.3 Плановые сроки начала и окончания работы по созданию системы

Начало работы: 06.09.2021

Окончание работы: 01.12.2021

### 1.1.4 Сведения об источниках и порядке финансирования работ

Источников и порядков финансирования работ не имеется.

## 1.3 Назначение и цели создания (развития) системы

### 1.3.1 Назначение системы

В данной работе рассматривается автосалон по продажи автомобилей. Таким образом, можно утверждать, что основной целью такой организации, как сеть автосалонов, является извлечение прибыли посредством продажи новых автомобилей своим клиентам. Как правило, в структуру такой организации должна входить система сервисного обслуживания автомобилей, однако, данный вопрос не включен в данную работу. Описание организации строится из представления о наличии автосалона и продажи автомобилей разных брендов, объединенных в одну коммерческую организацию.

Корпоративное приложение разрабатывается как система, которая имеет функции проведения заказов на автомобиль, она обеспечивает доступ к информационной базе данных, в которой хранятся имеющиеся автомобили и аксессуары к ним. Система позволяет выдать данные об имеющемся автомобиле и о его характеристиках, информацию о клиентах, активных заказах и выполненных заказах.

База данных разрабатывается как система для надежного и удобного хранения данных. Используется в том числе в клиент-серверной архитектуре. Базы данных позволяют быстро фильтровать данные, искать определенные экземпляры и сортировать данные с учетом параметров.

### 1.3.2 Цели создания системы

* Анализ предприятия и выделение необходимых сущностей для системы;
* Повысить выручку компании за счет быстрого проведения заказов;
* Повышение эффективности работников
* Увеличить качество обслуживания покупателей.
* Предотвращение воровства
* Контролировать наличие автомобилей и автозапчастей на складе.
* Обучение грамотному оформлению технической документации.

## 1.4 Характеристики объекта автоматизации

Автосалон — это магазин по продаже новых или подержанных автомобилей. Здесь можно выбрать и купить машину самостоятельно или с помощью сотрудников салона. Если автомобиль есть в наличии, можно сразу оформить его покупку и забрать из салона. Можно также оформить заказ на машину, если её нет в наличии, и забрать при поступлении в автосалон.

## 1.5 Требования к системе

### 1.5.1 Требования к системе.

Система должна быть централизованной и иметь сущности, необходимые для работы и выделенные при анализе предприятия и определения того, что нужно автоматизировать. Система должна быть устойчивой к чрезвычайным ситуациям, таким как:

* Случайное или предумышленное удаление базы данных;
* Случайное или предумышленное изменение модели данных.

Система должна предоставлять удобный и постоянный доступ к просмотру, редактированию и удалению данных.

#### 1.5.1.1 Требования к структуре и функционированию системы

Система должна иметь эргономичный интерфейс, который позволит сотрудникам любой квалификации получать доступ к данным и использовать базу данных для поиска и фильтрации информации.

#### 1.5.1.2 Требования к надежности

Программа должна обеспечивать корректную обработку исключительных ситуаций, вызванных вводом недопустимых дат, и любых других неверных данных.

#### 1.5.1.3 Требования к функциям (задачам), выполняемым системой

Система должна выполнять следующий ряд функций:

* Хранение, изменение и управление данными. Предоставление доступа к данным в любое время суток;
* Поиск и фильтрация данных на основе запросов к базе данных;
* Иметь защиту от повреждений в виде резервного копирования базы данных;
* Предоставлять возможность добавления автомобиля или аксессуара в базу данных при новом поступлении;
* Учет расходов;
* Импорт данных из таблиц Excel;
* Экспорт данных в Excel, Word;
* Склад аксессуаров и дополнительного оборудования, а также возможность их продажи;
* Каталог автомобилей с поиском по любой характеристике;

## 1.6 Состав и содержание работ по созданию системы

Таблица 1. – Стадии и этапы разработки ТЗ

|  |  |  |
| --- | --- | --- |
| Стадии | Этапы работ | Сроки исполнения |
| 1. Формирование требований к АС | Исследование объекта и подтверждение  необходимости создания АС.  Формирование требований пользователя к АС.  Оформление отчёта о  выполненной работе и заявки на разработку АС. | 01.10.2021 |
| 2. Разработка концепции АС | Изучение выбранного объекта.  Проведение научно- исследовательских работ.  Проектирования концепции АС, удовлетворяющей  потребности пользователя.  Оформление отчёта о выполненной работе. | 08.10.2021 |
| 3. Техническое задание | Разработка и утверждение технического задания на  создание АС. | 22.10.2021 |
| 4. Разработка системы | Разработка системы согласно  техническому заданию. | 05.11.2021 |
| 5.Защита курсовой работы | Создание грамотной презентации и речи для защиты курсовой работы.  Защита курсовой работы. | 01.12.2021 |

## 1.7 Порядок контроля и приемки системы.

### 1.7.1 Состав, объем и методы испытаний системы и ее составных частей.

В результате предварительных испытаний на нормализацию должны быть исправлены недочёты.

Для проверки корректной работы внесённых изменений должны быть проведены повторные испытания разработанной программы.

### 1.7.2 Общие требования к приемке работ.

В процессе приемки работ должна быть осуществлена проверка на соответствие требованиям настоящего «Технического задания». По результатам испытаний возможны доработки и исправления.

## 1.8 Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие.

Требований к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие не имеется.

## 1.9 Требования к документированию

По окончанию работы предъявлена расчетно-пояснительная записка, в состав которой входят:

* техническое задание;
* научно-исследовательская часть;
* проектно-конструкторская часть;
* проектно-технологическая часть.

Также должна быть предоставлена графическая часть работы, выполненная в формате А1 на 2 листах, в которую входят:

* демонстрационные чертежи;

## 1**.10 Источники разработки**

Гост 34.601-90 «Автоматизированные системы. Стадии создания»

Гост 34.602-89 «Техническое задание на создание автоматизированной системы»

# 2. НАУЧНО – ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ

## 2.1 Постановка задачи проектирования

Разрабатываемая база данных будет предназначена для реализации корпоративного приложения, со следующей функциональностью.

* Учет прихода и продажи автомобилей;
* Учет прихода и продажи товаров для автомобилей (аксессуары);
* Проведение необходимых документов;
* Хранение информационной базы по клиентам, автомобилям и аксессуарам;
* Возможность рассчитать кредит;

База данных будет содержать информацию о автомобилях, их производителе, модели и комплектации. Также, она позволит узнавать информацию о аксессуарах которые имеются в наличие у автосалона. Пользователь сможет добавлять товары в базу данных, создавать договоры и заявки, вести базу данных клиентов.

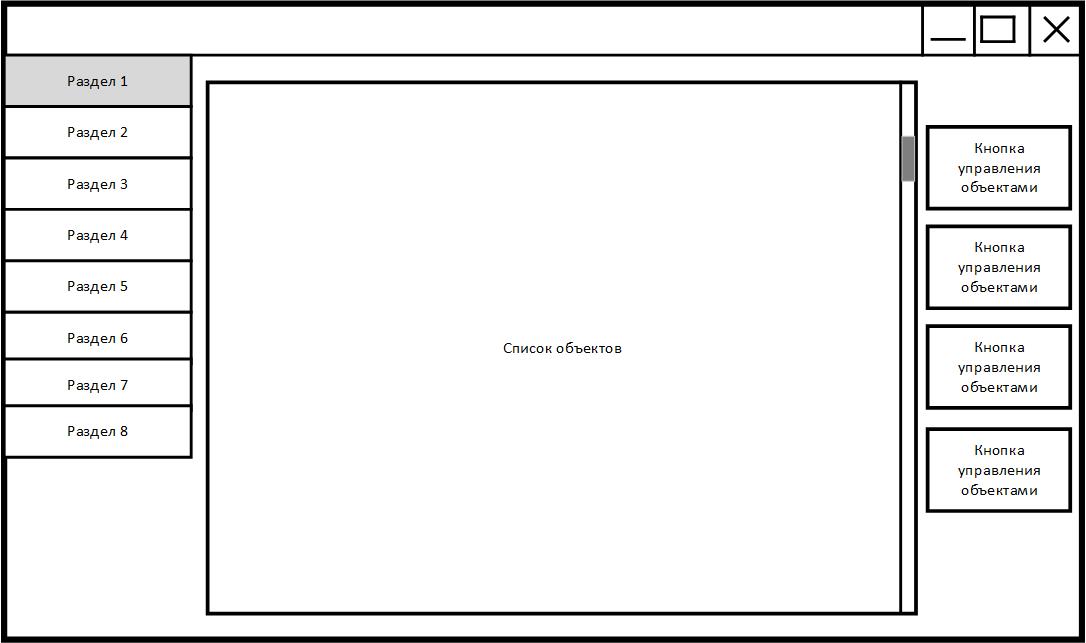
****

Рисунок 2.1 – Макет главного рабочего экрана приложения

## 2.2 Описание предметной области.

Автосалон — это некий аналог магазина, в котором представлены существующие образцы автомобилей. Автосалон осуществляет автомобильную дилерскую логистику, планирование производственного заказа на выпуск автомобилей, посредством официального дилерского договора.

Тема продаж и покупок автомобилей очень популярна в наше время, так как сам предмет – автомобиль пользуется большой популярностью у населения. Для облегчения поиска нового автомобиля или места сбыта старого авто во многих городах организуются специальные стоянки или закрытые магазины, называемые автосалоном, в которых клиенты могут подобрать себе автомобиль по вкусу, а также избавиться от старого авто, выставив его на продажу.

Основные функции автосалона:

* осуществляет продажу новых автомобилей;
* совместно с дистрибьютором формирует производственную программу завода-производителя посредством производственного заказа на выпуск автомобилей. Формированием заказа на выпуск автомобилей с учётом предпочтений и вкусов клиентов в дилерском центре занимается диспонент;
* осуществляет предпродажную подготовку автомобиля и его послепродажное и гарантийное обслуживание — прохождение во время гарантийного периода планового технического обслуживания (замена эксплуатационных материалов, техническая диагностика, гарантийный ремонт и так далее);
* осуществляет продажу специальных банковских продуктов по кредитованию и страхованию проданных автомобилей;
* осуществляет продажу оригинальных (фирменных) запасных частей. Для этого на территории демонстрационного зала оборудуется специальное рабочее место специалиста по запасным частям для подбора и заказа нужной детали по каталогу или выдачи со склада дилерского центра.

Основными элементами предметной области являются поставляемые автомобили, которые обозначены следующими атрибутами – комплектация, цвет, vin номер. Также выделены основные составляющие автомобиля – производитель, модель. Данная предметная область содержит информацию о покупателях, сотрудниках, заявках, договорах и оплате клиента

Предполагаемая база данных должна обеспечивать работу автосалона по учету продаж и поставок автомобилей различных видов, база данных будет относиться к классу база данных управления предприятием, т. к. она будет обслуживать конкретный один автосалон, а не систему автосалонов в целом. Приложение упрощает процедуру поиска необходимой информации о товарах и ценах на них. С ее помощью сотрудник может легко узнать информацию о поставленном или проданном товаре.

Приложение упрощает процедуру поиска необходимой информации о товарах и ценах на них. С ее помощью сотрудник может легко узнать информацию о поставленном или проданном товаре, клиент может просматривать существующий в наличии модельный ряд авто, руководитель отслеживать деятельность сотрудников и динамику продаж.

**Описание понятий предметной области:**

* Производитель — автозавод, компания, фирма, занимающаяся разработкой, изготовлением или сборкой автомобилей.
* Модель автомобиля — это название или номер, которые производители используют для идентификации и продажи ряда похожих автомобилей.
* **Комплектация автомобиля — это определенный набор опций, которые завод изготовитель, устанавливает в свои автомобили по умолчанию.**
* Автомобиль - моторное дорожное и внедорожное транспортное средство, используемое для перевозки людей и грузов.
* Аксессуары – это вещи, которые обеспечивают комфорт в автомобиле
* Договор – это документ купли-продажи, который заключается между автосалоном и клиентом.
* Клиент – человек, которому предоставляются услуги автосалона.
* Сотрудник – человек, который работает в автосалоне и имеет доступ к системе.
* Заявки – документ, в котором четко прописаны характеристики товаров, заказываемых автосалоном.
* Журнал событий – таблица с сообщениями которые происходят в системе.

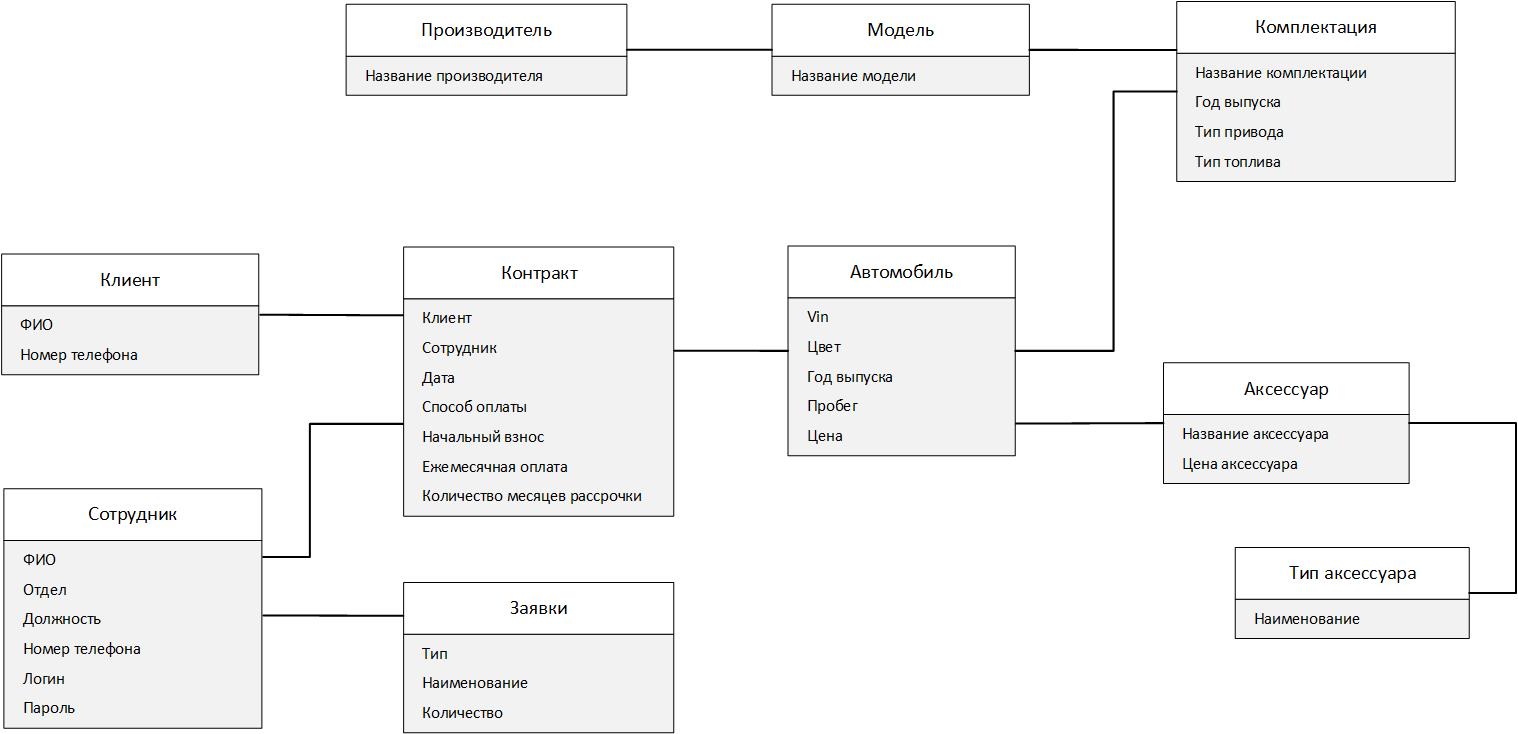


Рисунок 2.2 Логическая схема базы данных

## 2.3 Анализ аналогов.

### 2.3.1 «АвтоСалон»

Программа «АвтоСалон» предназначена для автоматизации деятельности автосалонов, торгующих автомобилями, в части, касающейся ведения учета, продаж автомобилей и формирования договорных документов. Также предусмотрено формирование и печать документов, специфичных для данного вида деятельности (справка-счет, ПТС, транзитный номер).

Возможности системы:

* Печать документов на бланках строгой отчетности (справка-счет, ПТС, транзитный номер);
* Неограниченное количество рабочих мест в рамках одной локальной сети;
* Сколько угодно много предприятий (торгующих организаций);
* Полная мультивалютность;
* Ведение различных справочников (клиенты, автомобили, комплектации и т.д.);
* Ведение Склада и Журнала продаж;
* Формирование различных отчетов и статистики;

### 2.3.2 «Альфа-Авто»

Программный продукт «Альфа-Авто» предназначен для комплексной автоматизации учета на предприятиях автобизнеса (магазинов оптовой и розничной продажи запчастей, автосервисов и станций технического обслуживания автомобилей, автосалонов и дилерских центров).

С помощью «Альфа-Авто» можно:

* оформлять заказы и продажи автомобилей;
* выполнять предпродажную подготовку автомобилей;
* оказывать услуги по ремонту и обслуживанию автомобилей;
* вести оптовую и розничную торговлю запасными частями;
* организовать работу склада;
* учитывать оплаты и отслеживать состояние взаиморасчетов с покупателями и поставщиками;

Конфигурация позволяет:

1. Учет запчастей:

* оптовая продажа запчастей;
* розничная продажа запчастей;
* работа по заказам запчастей;
* закупки запчастей;
* организация внутрифирменного товародвижения;

1. Автосервис:

* планирование ресурсов;
* оформление ремонтов;
* автоматизированное рабочее место сотрудника автосервиса;

1. Автосалон:

* клиентские и складские заказы;
* покупка и продажа автомобилей;
* предпродажная подготовка автомобилей;
* сделки «Trade-in»;
* ответственное хранение автомобилей;

1. Финансовый блок:

* оплаты покупателей и поставщикам;
* ведение взаиморасчетов с подотчетными лицами;
* бюджетирование;

1. Обмен данными:

* обмен данными с бухгалтерскими системами;
* обмен данными с каталогами производителей;

Пользователи программы имеют возможность быстро формировать необходимые документы. Руководство может оперативно получать и использовать данные о различных аспектах деятельности компании. Система предоставляет информацию, необходимую для принятия управленческих решений.

## 2.4 Перечень задач, подлежащих решению в процессе разработки.

1. Определить состав и структуру данных;
2. разработка пользовательского интерфейса;
3. Определить архитектуру приложения;
4. разработка и реализация базы данных;
5. разработка функционала автоматизированной информационной системы;
6. реализация программного обеспечения системы;
7. Тестирование и проверка разработанной автоматизированной информационной системы и базы данных.
8. Исправление выявленных ошибок.

## 2.5 Обоснование выбора инструментов и платформы для разработки.

Для выполнения курсовой работы были выбраны:

1. Операционная система Windows 10.

Данная операционная система является одной из самых распространенных на данный момент, она обладает так же рядом существенных преимуществ. Одним из критериев выбора Windows 10 является то, что она установлена на многих предприятиях

Это единый (для всех программ-приложений Windows), стандартизированный и интуитивно-понятный интерфейс взаимодействия с пользователем. Т.е., даже начинающему пользователю персонального компьютера, не требуется особых усилий для освоения навыков работы с данным программным средством.

Причины выбора Операционной системы Windows 10:

1. Простота эксплуатации. Даже неподготовленному человеку легко разобраться в принципах работы операционной системы, ведь все интуитивно понятно, также система не требует знаний в программировании.

2. Большое количество программ сторонних компаний выпускают именно под данную операционную систему. Есть как платные, так и бесплатные версии.

3. Большая часть выпускаемого оборудования для ПК выпускается с поддержкой ОС Windows. Веб-камеры, сканеры, принтеры, игровые манипуляторы и так далее. Все драйвера, в первую очередь, выходят под данную ОС, и лишь по прошествии большого промежутка времени выходят под другие операционные системы.

4. Большая распространённость. Данная операционная система, по состоянию на апрель 2020 года, заняла второе место в общей мировой статистике использования ОС.

5. Различные формы и виды оформления. Кроме стандартных вариантов ОС позволяет сторонним программам вносить изменения во внешний вид рабочего стола, папок, заставок и так далее.

2. Язык программирования С#

В настоящее время C # и чаще всего используются для создания коммерческих и бизнес-приложений. Эти языки подходят для большинства программистов. Но на самом деле развитие производительности не было правильным. Существует язык, который в настоящее время разрабатывается из-за потери привычной эластичности и необходимости программистов на C # / C ++. Эти решения не очень удобны для программистов и предлагают значительно меньше возможностей. Они не ориентированы на текущее взаимодействие с системой и часто несовместимы с существующими методами разработки для Интернета. Большинство разработчиков хотят использовать современные языки, которые позволяют писать, читать и разрабатывать программы с простотой Visual Basic, обеспечивая при этом мощь и гибкость C ++ для доступа ко всем функциям системы. Взаимодействовать с существующими стандартными веб-приложениями

Основные преимущества языка:

1. По умолчанию C # запрещает прямое управление памятью, возвращая различные типы систем и сборку мусора. Работа напрямую с памятью все еще существует в специальном режиме кода «опасности», но это должно быть четко объявлено. По этой причине в C # активно используется только один поставщик доступа «.».
2. В C # нет методов множественного наследования. Но вместо этого предлагается использовать создания нескольких интерфейсов. Хотя многие мнения о наследовании разные. Но отсутствие этого механизма в C # должно, по крайней мере, облегчить разработку компилятора.
3. Система типов C # представляет систему типов. NET как большинство современных современных языков. Важной особенностью системы этого типа является четкое разделение всех категорий на типы значений и ссылочные типы.
4. В C # можно организовать данные в структуру, состоящую из переменных любого типа или распределения, состоящего из множества переменных одного типа. Важной особенностью распределения в C # является необходимость явного преобразования в базовый тип, если вы хотите интерпретировать значения из перечисления в числа

Кроме того, следующие языковые механизмы могут быть отнесены к особенностям языка:

1. Исключения
2. Встроенный механизм контроля версий (Цель этого механизма - гарантировать, что пользователям придется изменять или компилировать старые библиотеки только тогда, когда это абсолютно необходимо.)
3. Возможность отключить контроль переполнения (Ключевые слова, которые не проверены)

Исходя их этого, C # является ориентированным на программирование для платформы. NET и используется как для прототипирования программ и для разработки крупномасштабных приложений

C # поддерживает .NET Framework 4, что позволяет использовать опыт разработки приложений для устройств.

Исходя из всего вышеперечисленного, можно прийти к выводу, что язык программирования C#, несомненно, подходит для разработки автоматизированной информационной системы.

3. Объектно-реляционная система управления базами данных PostgreSQL

Из характеристик СУБД, которые могут определить выбор, одной из важнейших является модель данных. Теоретически любую информацию можно представить в виде реляционной модели. Сила реляционных баз данных в том, что эта модель очень хорошо подходит для предприятий, которые располагают немалыми средствами для активного внедрения передовых систем. Эта модель имеет наиболее проработанное математическое основание и хорошо проработанные стандарты. Реляционная модель данных отличается большой гибкостью с точки зрения изменения структуры данных. Здесь можно менять физическую структуру данных, не переписывая приложения.

Выбор СУБД напрямую связан с анализом и выбором аппаратных ресурсов. Стандартные тесты проводятся на современных быстродействующих системах, в то время как целью клиента может быть просто работоспособная система в рамках отведенного бюджета. Не менее важны и взаимодействие с унаследованными системами, и возможности переноса накопленных данных на новую систему.

Важным критерием выбора с точки зрения перспектив становится наличие эффективных средств разработки. Такие средства, обладающие удобным интерфейсом, позволяют специалистам предприятия самостоятельно и быстро настраивать информационные системы в соответствии с требованиями бизнеса.

PostgreSQL - это мощная объектно-реляционная система управления базами данных с открытыми исходными текстами. Она разрабатывается на протяжении более 15 лет и улучшает архитектуру, чем завоевала репутацию надежной, интегрированной и масштабируемой СУБД. Она запускается на всех основных платформах, включая Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), и Windows. Она полностью соответствует ACID, имеет полную поддержку ключей, объединений, представлений, триггеров, и хранимых процедур (на разных языках). Она включает большинство типов данных SQL92 и SQL99, включая integer, numeric, boolean, char, varchar, date, interval, и timestamp. Она также поддерживает хранение больших двоичных объектов (BLOB's), включая картинки, звук, или видео. Она имеет API для C# C/C++, Java, Perl, Python, Ruby, Tcl, ODBC и др.

Являясь СУБД класса предприятия, PostgreSQL предоставляет такие особенности как Multi-Version Concurrency Control (MVCC), восстановление по точке во времени, табличное пространство, асинхронная репликация, вложенные транзакции (точки сохранения), горячее резервирование, планировщик/оптимизатор запросов, и упреждающее журналирование на случай поломки. Он поддерживает международные кодировки, в том числе и многобайтовые, при использовании различных кодировок можно использовать сортировку и полнотекстовый поиск, различать регистр. Большое количество подконтрольных данных и большое число одновременно работающих пользователей, тем не менее, не сильно влияет на масштабируемость системы. Есть действующие PostgreSQL системы, которые управляют более чем 4 терабайтами данных.

Средства обеспечения целостности данных включают составные первичные ключи, внешние ключи с поддержкой запрета и каскадирования изменений/удалений, проверку ограничений (constraints), ограничения уникальности и ограничения на непустые значения.

PostgreSQL может выполнять хранимые процедуры, написанные на различных языках программирования, включая Java, Perl, Python, Ruby, Tcl, C/C++, и собственном PL/pgSQL, аналогичном Oracle's PL/SQL. В стандартную библиотеку функций включены сотни встроенных функций — от базовых математических и строковых операций до криптографических функций и функций, обеспечивающих совместимость с Oracle. Триггеры и хранимые процедуры могут быть написаны на Cи и загружены в базу данных в качестве библиотеки, позволяя тем самым расширять ее возможности. Также PostgreSQL включает средства разработки, позволяющие создавать пользовательские типы данных вместе с функциями и операторами, описывающими их поведение. Как результат, созданы и могут быть добавлены к системе различные типы данных — от геометрических и пространственных примитивов до типов данных, определенных в ISBN/ISSN (International Standard Book Number/International Standard Serial Number).

3. Среда разработки Microsoft Visual Studio 2019

Microsoft Visual Studio 2019— это набор инструментов разработки, основанных на использовании компонентов, и других технологий для создания мощных, производительных приложений. Кроме того, среда Visual Studio оптимизирована для совместного проектирования, разработки и развертывания корпоративных решений. Visual Studio предоставляет средства для проектирования, разработки b отладки.

Среда разработки Visual Studio представляет собой полный набор средств разработки для создания настольных приложений. Visual C# использует единую интегрированную среду разработки (IDE), которая позволяет совместно использовать средства и упрощает создание решений на базе нескольких языков. Можно создать обычные приложения Microsoft Windows и приложения с архитектурой "клиент-сервер" с помощью использования конструкторов в Visual Studio.

Почему именно эта версия продукта, потому что в нем есть такие удобные нововведения как:

1. Visual Studio IntelliCode повышает эффективность разработки программного обеспечения с помощью искусственного интеллекта (ИИ). Для создания рекомендаций IntelliCode анализирует 2000 проектов с открытым кодом на GitHub
2. Рефакторинг В C#. Есть много новых удобных возможностей рефакторинга, которые помогают упорядочить код. Они отображаются как предложения со значком лампочки и включают такие действия, как перемещение элементов в интерфейс или базовый класс, настройку пространств имен в соответствии со структурой папок.

# 3. ПРОЕКТНО – КОНСТРУКТОРСКАЯ ЧАСТЬ

## 3.1 Разработка структуры приложения.

Данный проект имеет 3 уровневую архитектуру с применением шаблона MVVM

Для представления принципа работы приложения ниже приведена его UML диаграмма вариантов использования

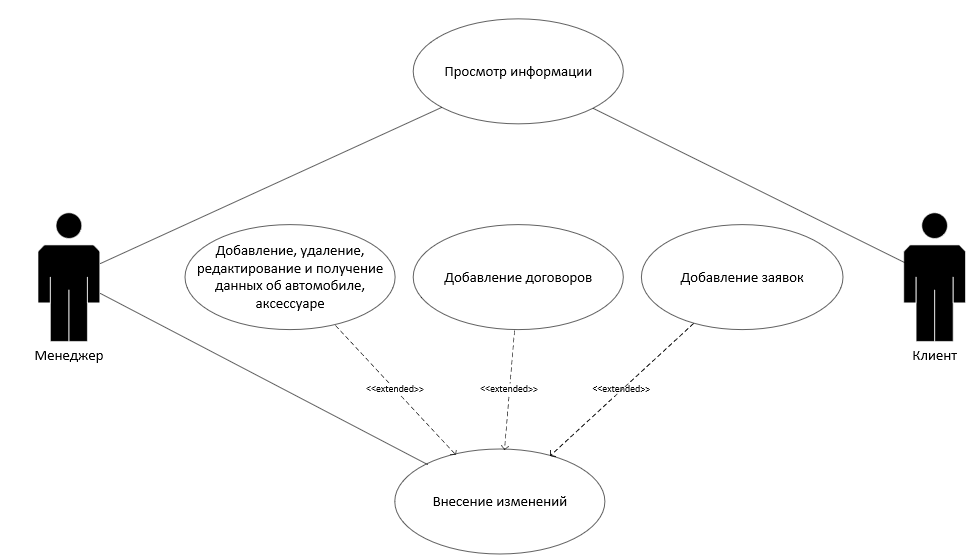


Рисунок 3.1 – Диаграмма использования приложения

В проекте база данных хранить в себе всю информацию о товарах, предоставляемых автосалоном, все действия пользователей приложения и составляемых договоров и заявок. Запросы формируются в слое Model, дальше передаются на обработку в ViewModel и после обработки готовые обработанные данные предоставляться пользователю в слое View. Передача данных также проходит через все слои, пользователь задает параметры или вводит информацию в слое View, дальше данные обрабатываются в ViewModel и передаются в Model, где уже обработанные данные и вносятся или изменяются в базе данных.

Каждой сущности из базы данных соответствует свой класс, для более удобного предоставления информации в слое View. Такое разделение на слои позволяет расширять функционал приложения и постоянно его усовершенствовать. Бизнес-логика выделена в отдельные классы для масштабируемости системы.

Для обеспечения точности и надежности данных были введены ограничения на уровне базы данных. Они включают в себя ограничения уникальности и проверочные ограничения (например, в некоторых таблицах используется ограничение NOT NULL что означает, что нельзя вставить новую запись или обновить запись без добавления значения в это поле). Также были введены ограничения на уровне Бизнес-логики. Например, каждый автомобиль должен иметь свой уникальный VIN номер

Самое главное преимущество MVVM – это технология двусторонней привязки: при изменении модели модель представления будет автоматически обновляться, и представление также автоматически изменится. Это означает хорошую согласованность данных. А также MVVM обеспечивает изолированное модульное тестирование.

## 3.2 Разработка алгоритмов обработки информации.

### 3.2.1 Алгоритм подсчета ежемесячного платежа

Размер аннуитетного ежемесячного платежа высчитывался по формуле

;[[1]](#footnote-1)

Где S — сумма займа, P — 1/100 доля процентной ставки (в месяц), N — срок кредитования (в месяцах).

И также подсчитывалась доля процентов в ежемесячном взносе по формуле

Где S — остаточный объем средств, P — процентная ставка.

От пользователя требовалось указать процентную ставку, срок кредитования в месяцах и первоначальный взнос (если такой присутствовал). Соответственно методу передавались обработанные входные параметры и производился подсчет. В ходе написания алгоритма использовались использовался класс Math из пространства имен System для получения степени числа.

Для тестирования был написан unit-test по результату, которого не было выявлено никаких ошибок

### 3.2.2 Алгоритм создания Word документа

Для реализации этого алгоритма использовался .NET Framework с использованием библиотеки Microsoft.Office.Interop, а также был выделен отдельный класс ProcessingWord. Для того чтобы реализовать этот алгоритм необходимо было создать заранее шаблон word документа, в котором присутствовали «тэги» для замены их данными.

Конструктор класса ProcessingWord принимал в качестве параметра название шаблона, который располагался в ресурсах проекта и проверял на его наличие. Далее вызывался метод Process который принимал список типа Dictionary<string, string> в котором в качестве ключа хранился «тэг» который был записан в шаблоне, и в качестве значение хранились данные для замены, а также передавалось название папки, куда сохранить созданный word документ. В методе открывается шаблон word документа и далее в цикле перебираются все значения из списка Dictionary<string, string> и ищутся ключи(тэги) уже в самом шаблоне и заменяются на значения из списка. Когда цикл отработал он передает измененный шаблон в функцию SaveAs2 путь куда необходимо сохранить изменений word документ. Функция SaveAs2 копирует все содержимое документа в новый который передаться в качестве параметра, причем после копирования изменённый шаблон не сохраняется. Далее необходимо было предусмотреть все возможные ошибки и избежать ситуации, такой в которой word документ мог не закрыться в связи с окончанием работы или с ошибкой алгоритма. Поэтому в конструкции try-catch-finally в блоке finally закрываются и очищаются все используемые ресурсы. В итоге на выходе мы получаем новый word документ и неизменённый шаблон, который также остается в ресурсах программы.

## 3.3 Логическая схема базы данных

Проектирование логической структуры БД должно решать задачи выбора наиболее эффективной структуры данных, обеспечения быстрого доступа к данным; исключения дублирования данных, обеспечения целостности данных таким образом, чтобы при изменении одних объектов автоматически происходило соответствующее изменение связанных с ними объектов.

При неправильно спроектированной схеме БД могут возникнуть аномалии модификации данных. Они обусловлены отсутствием средств явного представления типов множественных связей между объектами ПО и неразвитостью средств описания ограничений целостности на уровне модели данных.

Таблица Производители содержит:

* Уникальный идентификатор, который определяет брэнд производителя;
* Наименование производителя;

Таблица Модели содержит:

* Номер модели;
* Идентификатор, который хранит код производителя для связи таблиц;
* Наименование модели;

Таблица Комплектация содержит:

* Код комплектации;
* Идентификатор модели;
* Наименование комплектации;
* Тип привода, 4 вида привода подключаемый, передний, задний или полный;
* Тип топлива, рекомендация производителя для данной комплектации, самые известные типы топлива «АИ-92», «АИ-95», «АИ-98»;

Таблица Автомобиль содержит:

* Код автомобиля для данного автосалона;
* Цвет;
* VIN номер, уникальный номер на кузове транспортного средства который определяет все основные характеристики;
* Год выпуска;
* Номер договора, это поле, может быть, не заполнено если автомобиль продается;
* Код комплектации хранит информацию о комплектации;
* Пробег — это общее расстояние, пройденное транспортным средством;
* Цена;

Таблица Тип аксессуара содержит:

* Код типа аксессуара;
* Наименование типа аксессуара;

Таблица Аксессуар содержит:

* Код аксессуара;
* Код типа аксессуара, который определяет тип аксессуара;
* Наименование аксессуара;
* Цена;

Таблица Договоры содержит:

* Номер договора;
* Код клиента – клиент, с которым заключается договор;
* Код сотрудника– сотрудник, который предоставляет договор;
* Дата – число, месяц и год создания договора;
* Способ оплаты, предоставляемый автосалоном
* Начальный взнос – сумма, которую вносит клиент в качестве оплаты
* Ежемесячный платеж может отсутствовать, но при выдаче рассрочки автосалоном это сумма уплаты в месяц клиентом
* Количество месяцев рассрочки период выдачи рассрочки

Таблица Клиенты содержит:

* Код клиента;
* ФИО – фамилия, имя и отчество клиента;
* Номер телефона – контактный номер телефона для связи с клиентом;

Таблица Сотрудники содержит:

* Код сотрудника;
* ФИО – фамилия, имя и отчество сотрудника;
* Номер телефона – контактный номер телефона для связи с сотрудником;
* Отдел – подразделение, в которое входит сотрудник;
* Должность - служебное место, связанное с исполнением обязанностей;
* Логин уникальный идентификатор пользователя, который используется для входа в систему;
* Пароль – набор символов, которые знает только сотрудник, используется для входа в систему;

Таблица Заявки содержит:

* Код заявки;
* Тип – тип заказываемого товара;
* Сотрудник - уникальный идентификатор пользователя;
* Наименование заказываемого товара;
* Количество;

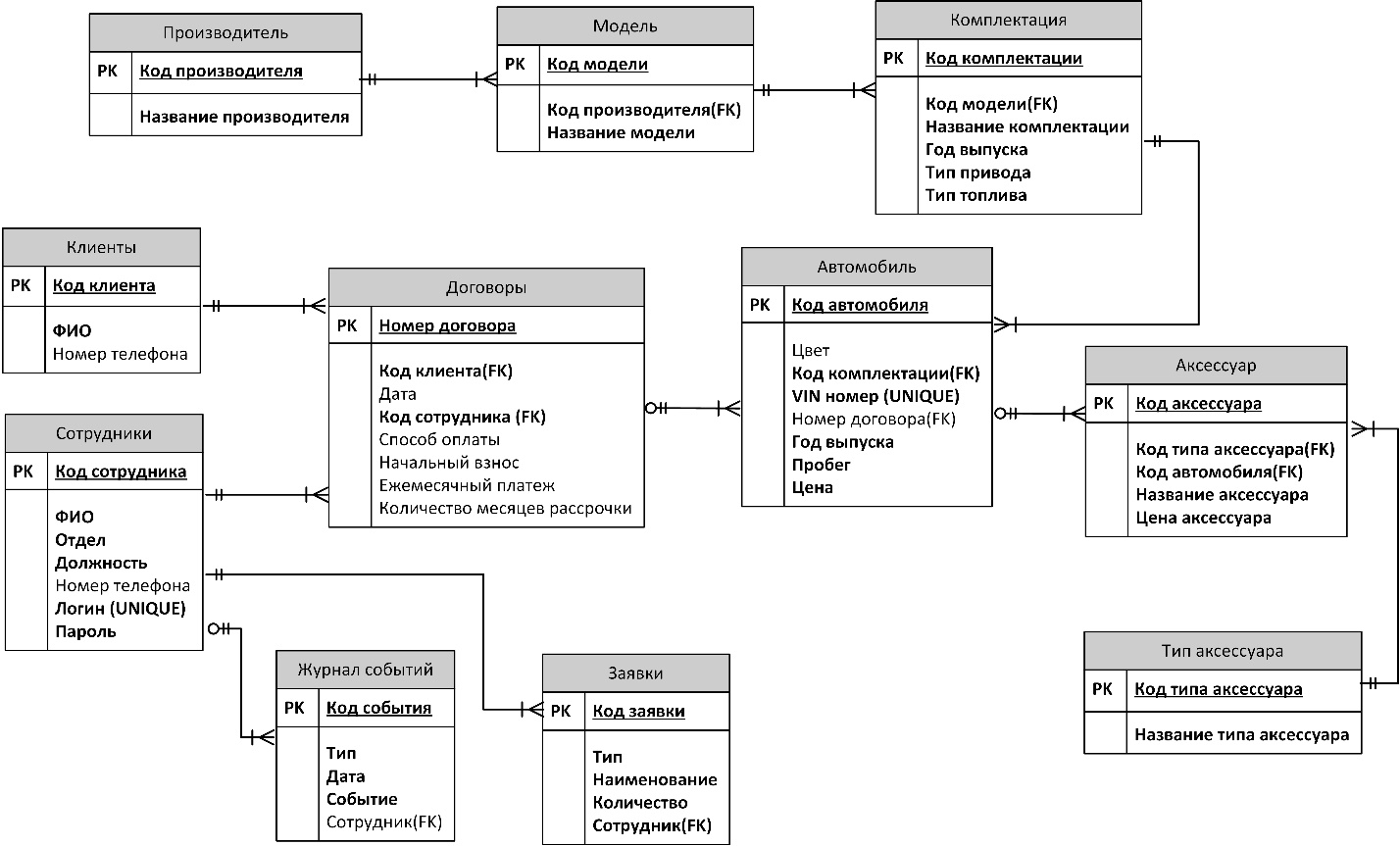


Рисунок 3.2 – Логическая схема базы данных

## 3.4 Разработка интерфейса взаимодействия пользователя с системой.

Для разработки красивого и удобного интерфейса применялась библиотека Material Design. А также были разработаны макеты пользовательского интерфейса:

1. Окно авторизации (Рисунок 3.3).

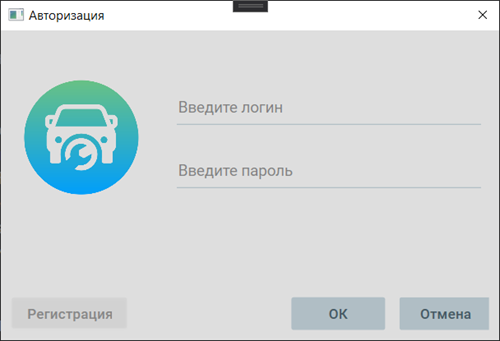


Рисунок 3.3 – Окно авторизации.

1. Окно регистрации (Рисунок 3.4).

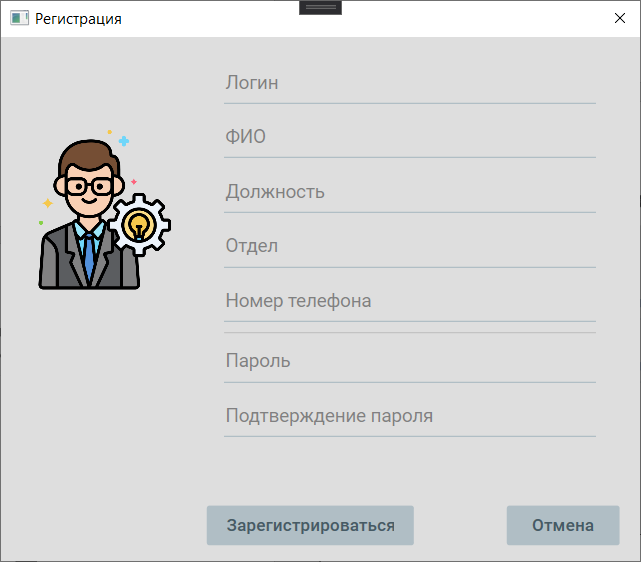


Рисунок 3.4 – Окно регистрации.

1. Страница для работы с автомобилями, добавление, удаление и изменение (Рисунок 3.5).

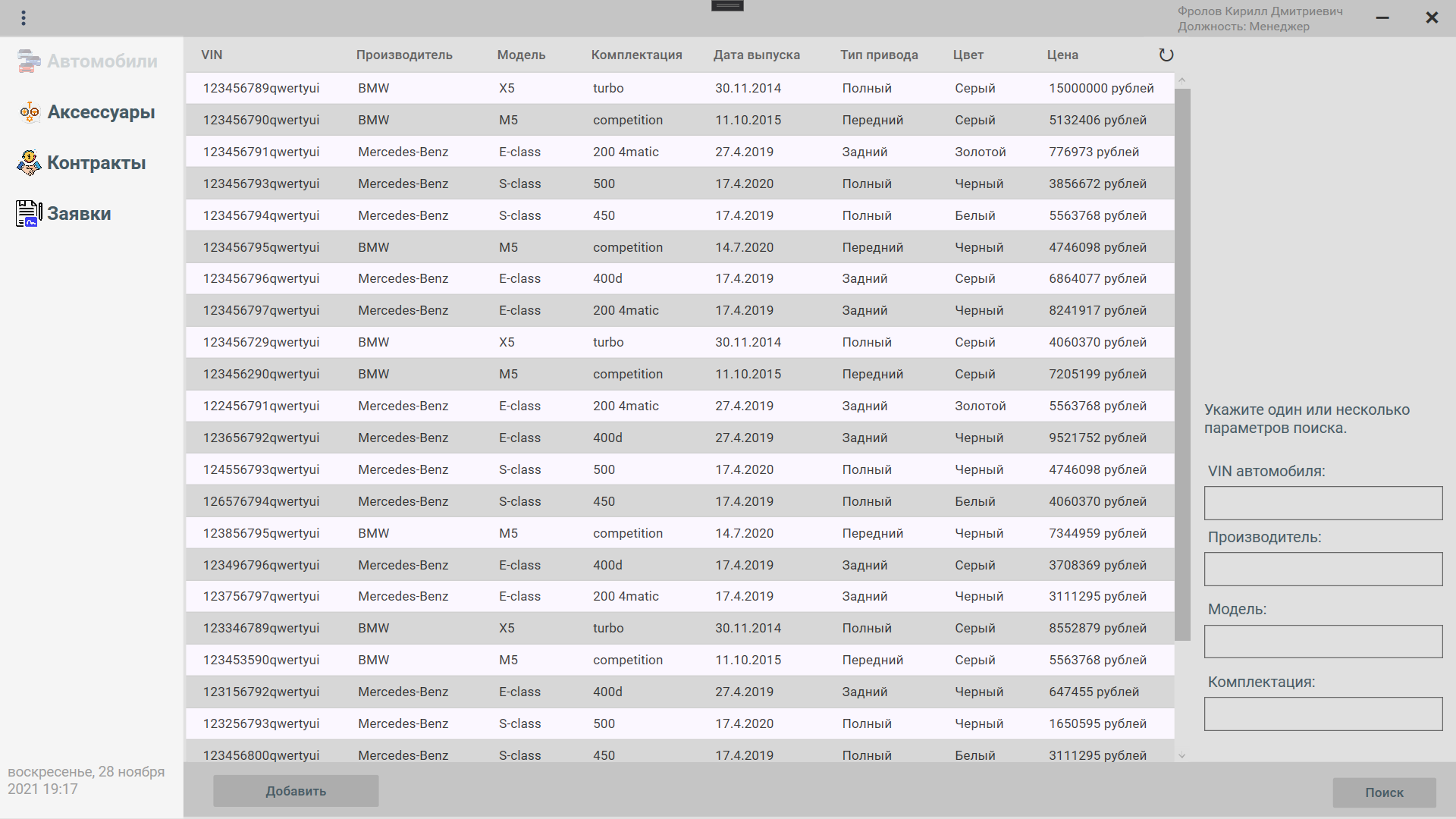


Рисунок 3.5 – Страница с автомобилями.

1. Окно для добавления производителей, моделей, комплектаций и автомобилей (Рисунок 3.6).

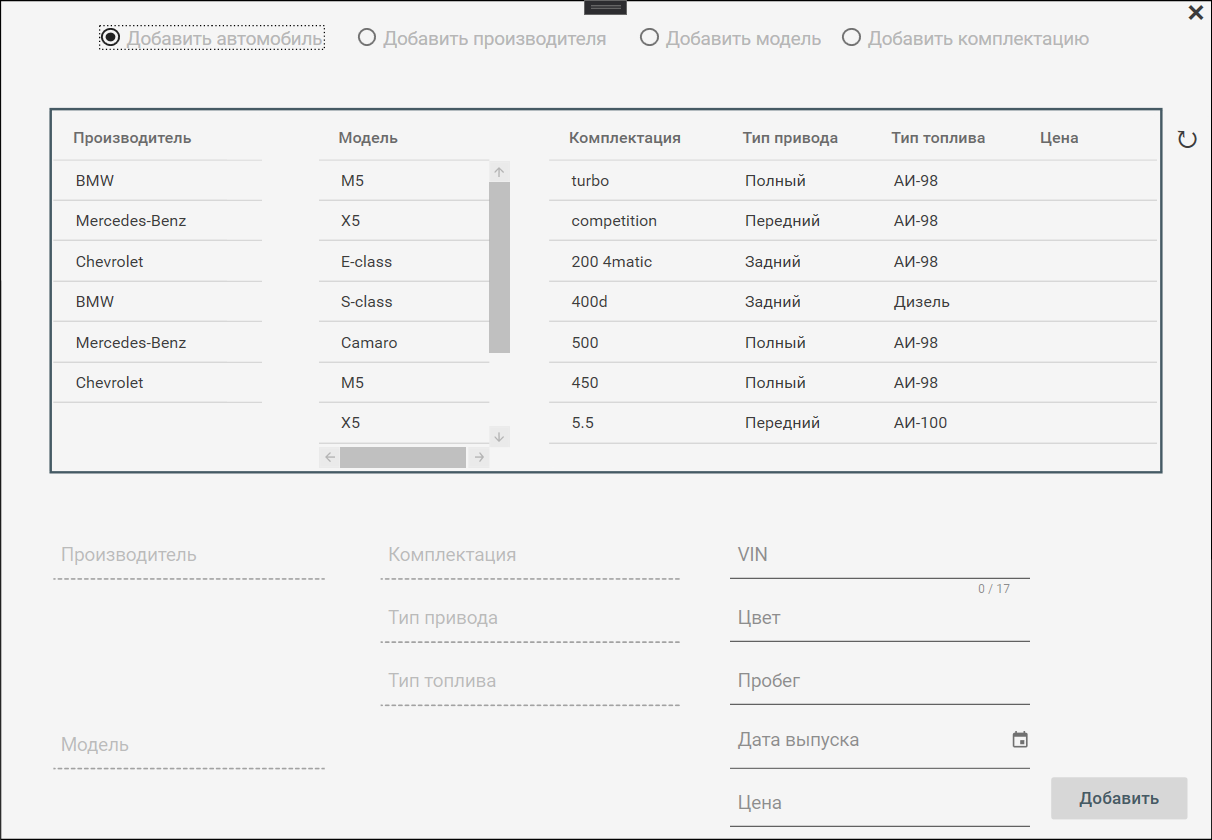


Рисунок 3.6 – Окно добавления автомобилей.

1. Информационное окно об автомобиле, где можно изменить некоторые характеристики (Рисунок 3.7).

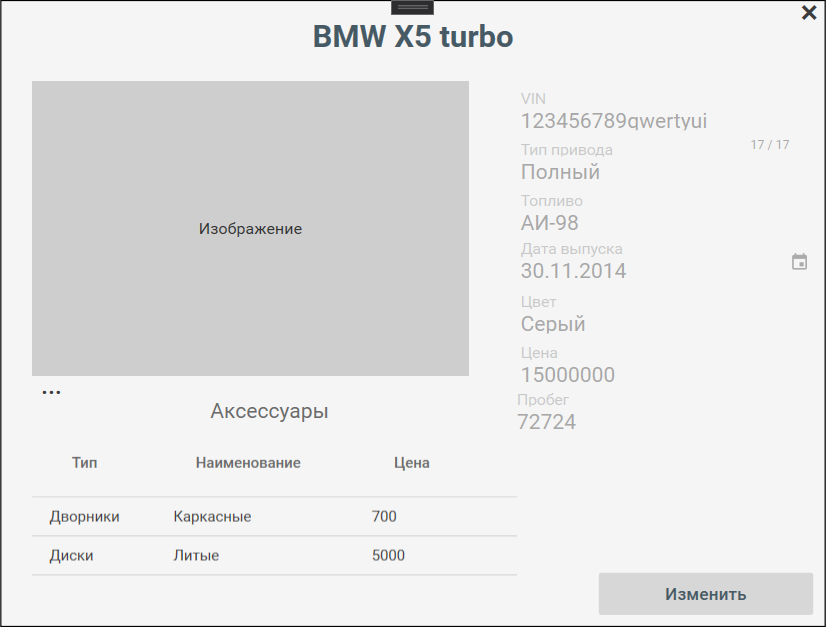


Рисунок 3.7 – Карточка автомобиля.

1. Страница для работы с аксессуарами, добавление, удаление и изменение (Рисунок 3.8).

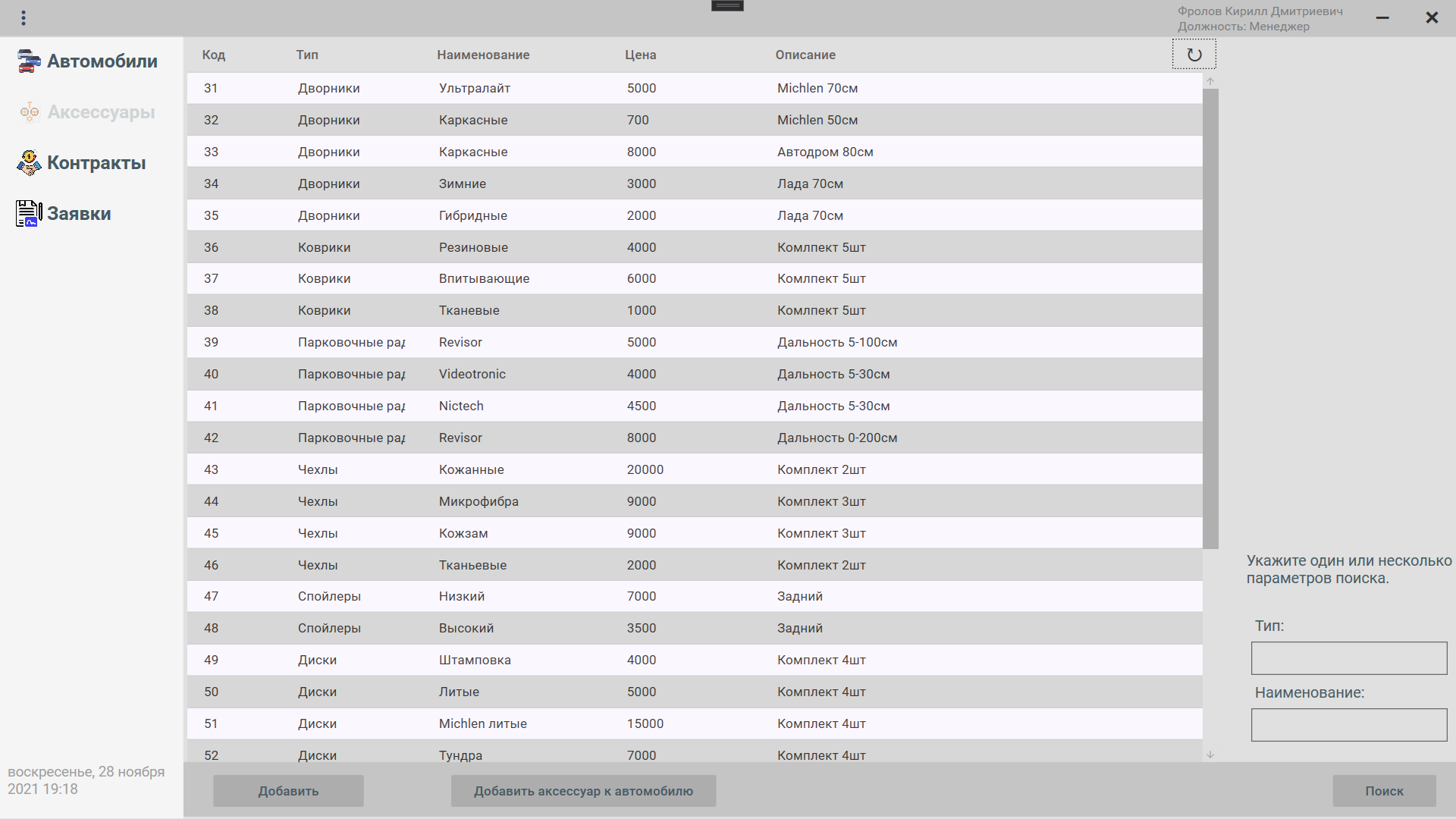


Рисунок 3.8 – Страница с аксессуарами.

1. Окно для добавления типов аксессуаров и аксессуаров (Рисунок 3.9).

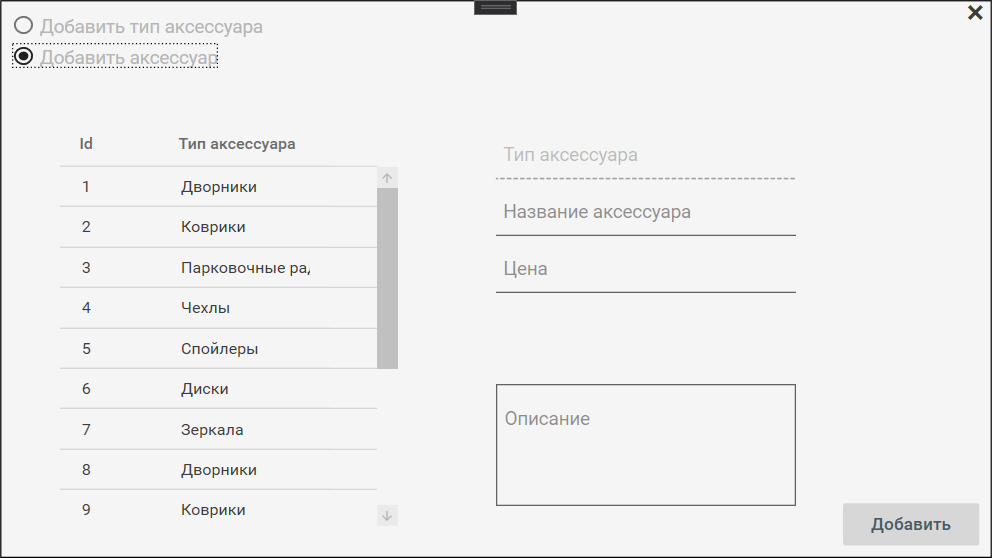


Рисунок 3.9 – Окно добавления аксессуаров.

1. Окно для добавления аксессуаров к автомобилю (Рисунок 3.10).

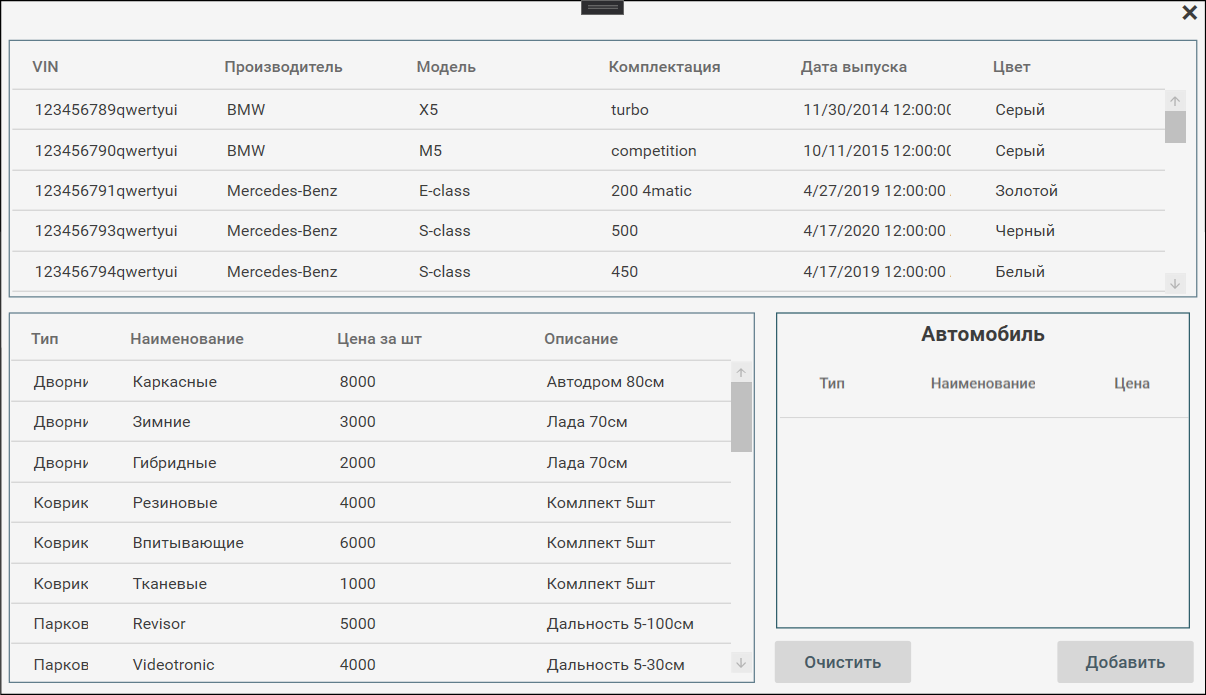


Рисунок 3.10 – Окно добавления аксессуара к автомобилю.

1. Страница для работы с договорами, где можно рассчитать кредит, добавлять автомобили в чек и сформировать документ word на основе введенной и рассчитанной информации (Рисунок 3.11).

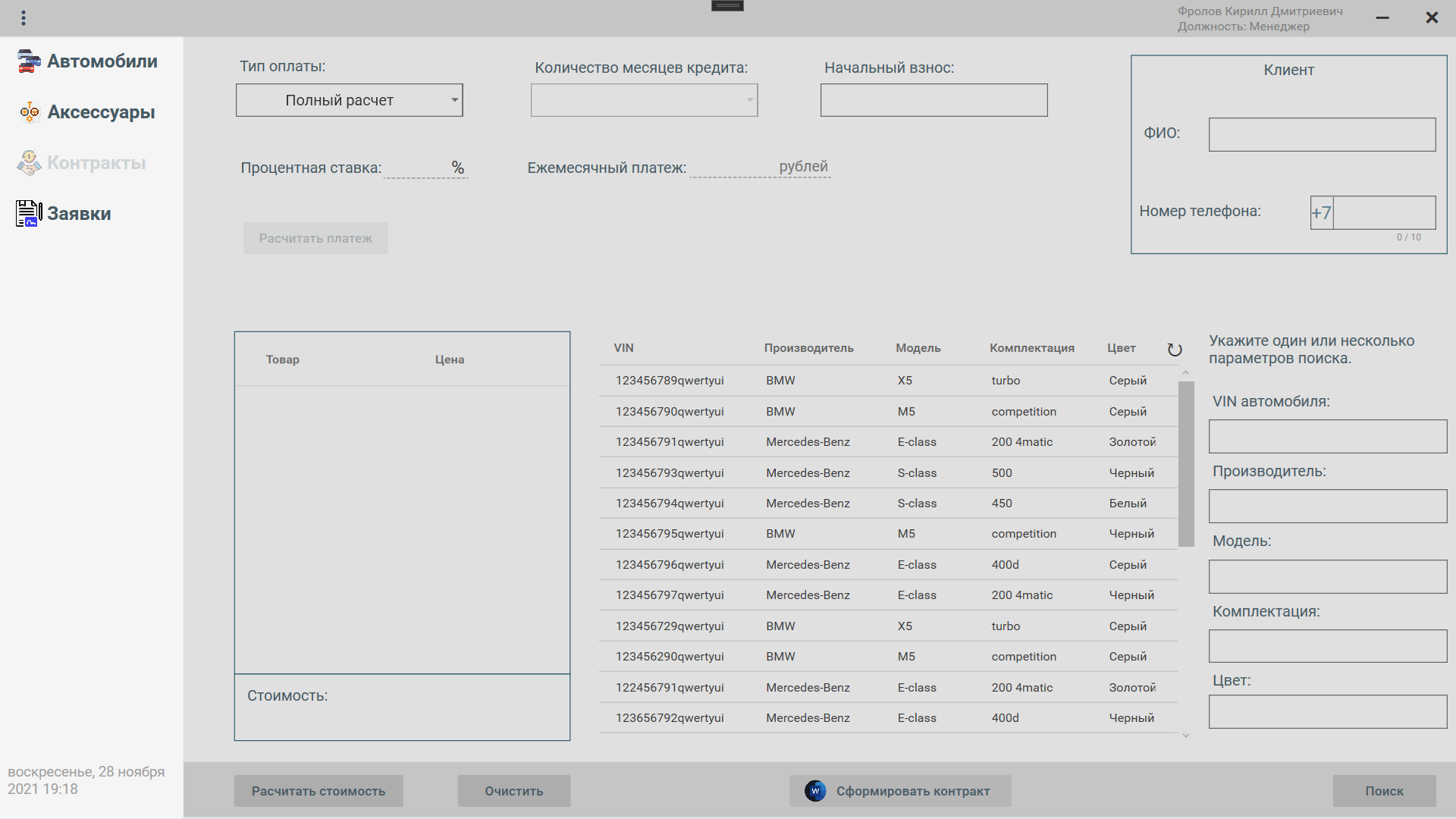


Рисунок 3.11 – Страница для контрактов.

1. Страница для добавления заявок на закупку (Рисунок 3.12).

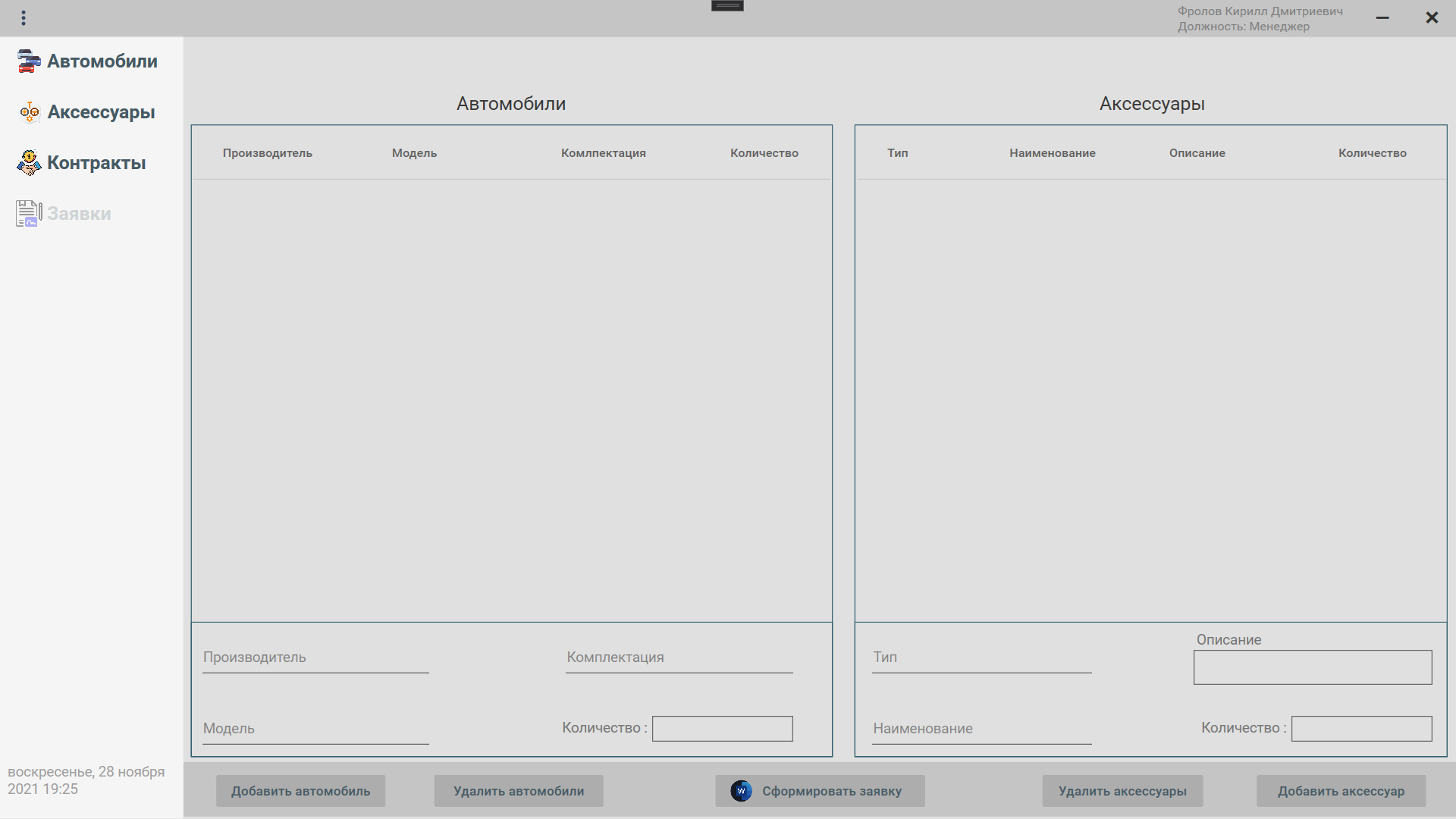


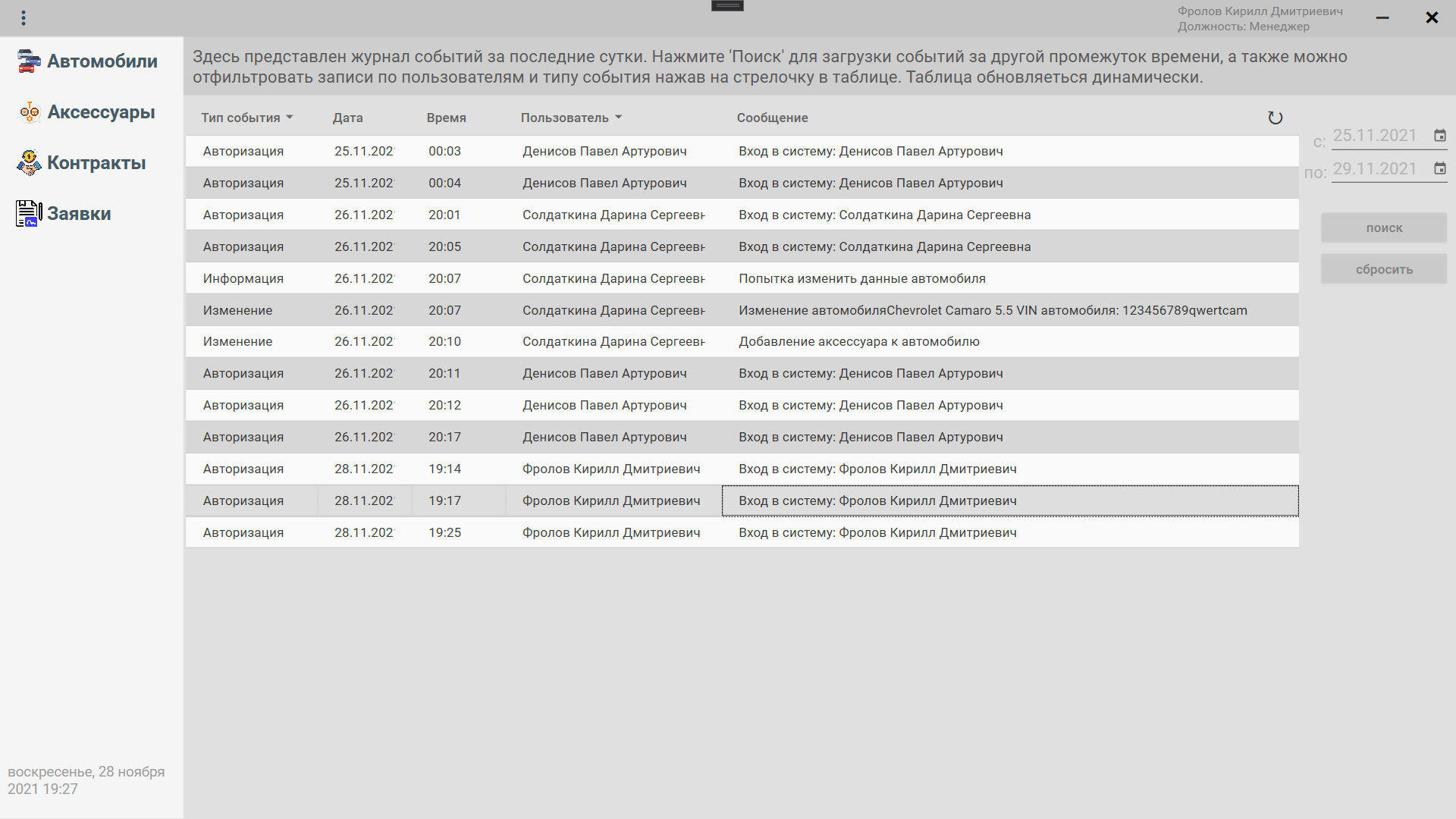
Рисунок 3.12 – Страница с заявками.

1. Страница с информацией о пользователе (личный кабинет) где можно планировать задачи (Рисунок 3.13).



Рисунок 3.13 – Страница личного кабинета.

1. Страница, где можно просматривать последние действия, произошедшие в системе (Рисунок 3.14).

  
Рисунок 3.14 – Страница журнала событий.

## 3.5 Разработка архитектуры приложения.

Было решено создавать настольное приложение, так как оно более эффективно подходит для решения многих задач, связанных с бизнесом, и доступна только в компании. Вэб-приложение не подходило бы для реализации данного проекта, потому что многие компании не часто обновляют системы и соответственно могут быть установлены на ней неподдерживаемые браузеры, которые могут плохо отображать информацию или некорректно работать с данными.

Поэтому был выбран язык программирования C# так как он подробно изучен и хорошо взаимодействует с фреймворком .NET и платформой WPF (Windows Presentation Foundation). При помощи этих инструментов можно создавать клиентские приложения для персональных компьютеров. Так как .NET отличается кроссплатформенностью достаточно будет установить его на операционную систему вместе с приложением и его можно будет использовать. Платформа разработки WPF поддерживает широкий набор компонентов для разработки приложений, включая модель приложения, ресурсы, элементы управления, графику, макет, привязки данных, документы и безопасность. Эта платформа является частью платформы .NET.

В данном проекте используется СУБД PostgreSQL так как она уже хорошо изучена. А также у данной СУБД есть поддерживаемый пакет Nqgsql.EntityFrameworkCore.PostgreSQL для удобной работы с базой данных.

Для связи с базой данных использовался Entity Framework это решение для работы с базами данных, которое используется в программировании на языках семейства .NET. Фреймворк позволяет взаимодействовать с СУБД с помощью сущностей, а не таблиц. Также код с использованием Entity Framework пишется гораздо быстрее. А также использовался язык интегрированных запросов LINQ to Entities для более удобного написания запросов.

В процессе разработки были обнаружены ограничения, связанные с формированием документа word. .Net Core не позволяет использовать в своем решении библиотеку Microsoft.Interop.Office которая предоставляет набор классов и функций для работы с word. Поэтому было принято решение создать отдельный проект на платформе .NET Framework которая позволяет работать с библиотекой Office и использовать этот проект в виде dll библиотеки в основном решении.

## 3.6 Реализация функционирующего приложения.

Физическая схема данных (Приложение 1). В ходе изучения предметной области было принято решение, ограничить

# 4. ПРОЕКТНО – ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

## 4.1 Проектирование начального и тестового наполнения базы данных. Процедуры автоматизации.

В системе основным источником данных является база данных. Данные обрабатываются уже непосредственно в системе.

Пользователь имеет доступ к данным, которые хранятся в базе данных, но не напрямую. Система обрабатывает данные, которые предоставляет или получает пользователь.

На страницах и окнах, представленных на рисунках 4.1 и 4.2 пользователю предоставляется доступ к данным из базы данных. Он может оперировать с данными, добавлять новые автомобили или аксессуары, изменять информацию о них или удалять. А также на странице (рисунки 4.4 и 4.10), пользователь сможет изменять данные о договорах добавляя в них автомобили, тем самым изменяя данные.

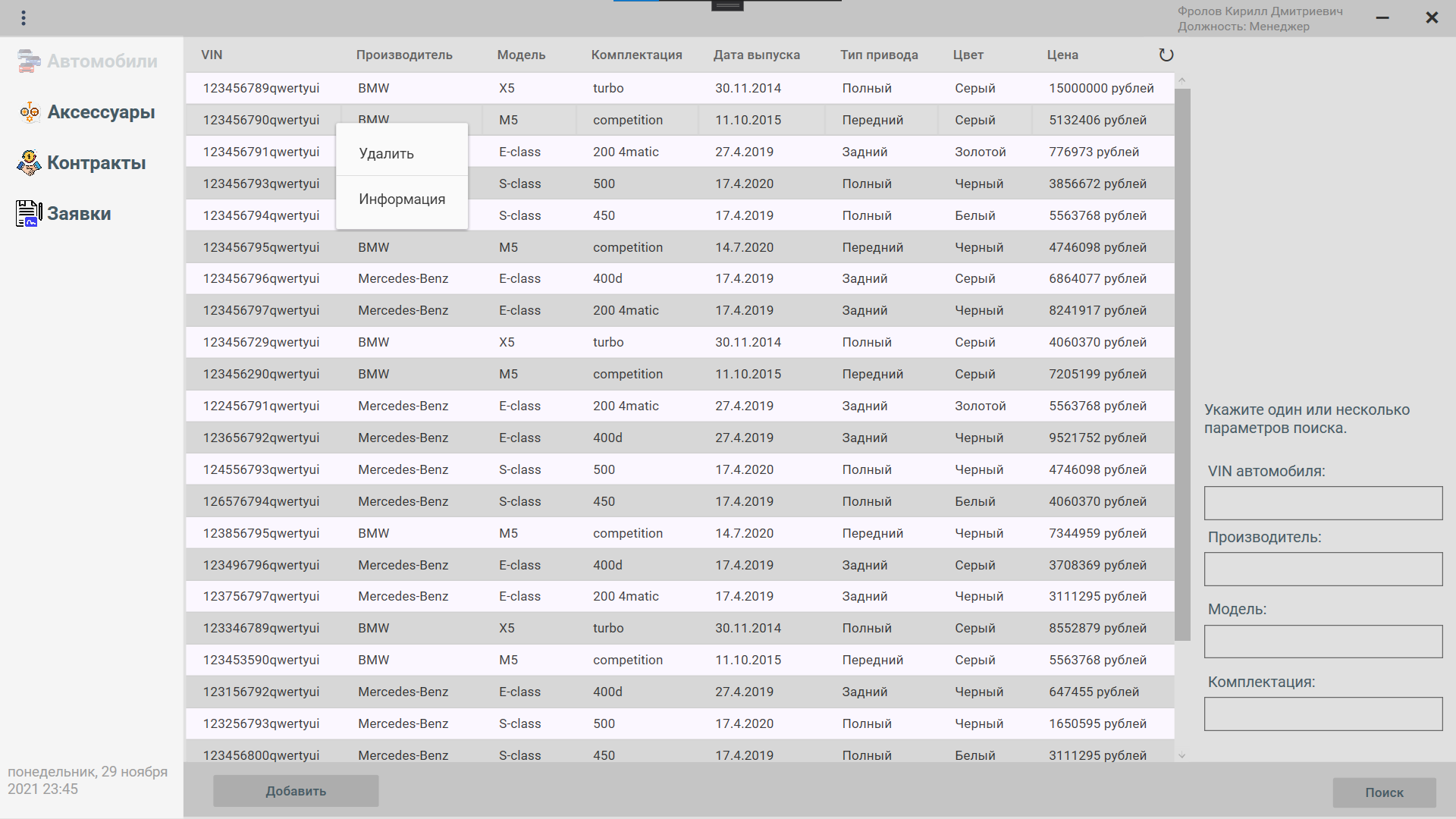


Рисунок 4.1 – Страница с автомобилями.



Рисунок 4.2 – Страница с аксессуарами.

На странице с договорами есть чек в левом углу, куда заносятся товары, которые будут включены в договор. Таблица с автомобилями, ассортимент выбора и поиск по автомобилям. Расчет кредита зависит от 4 параметров: стоимости товаров, количества месяцев кредита начальный взнос и процентная ставка. При выборе типа оплаты полный расчет, строго должен совпадать начальный взнос и стоимость товаров.

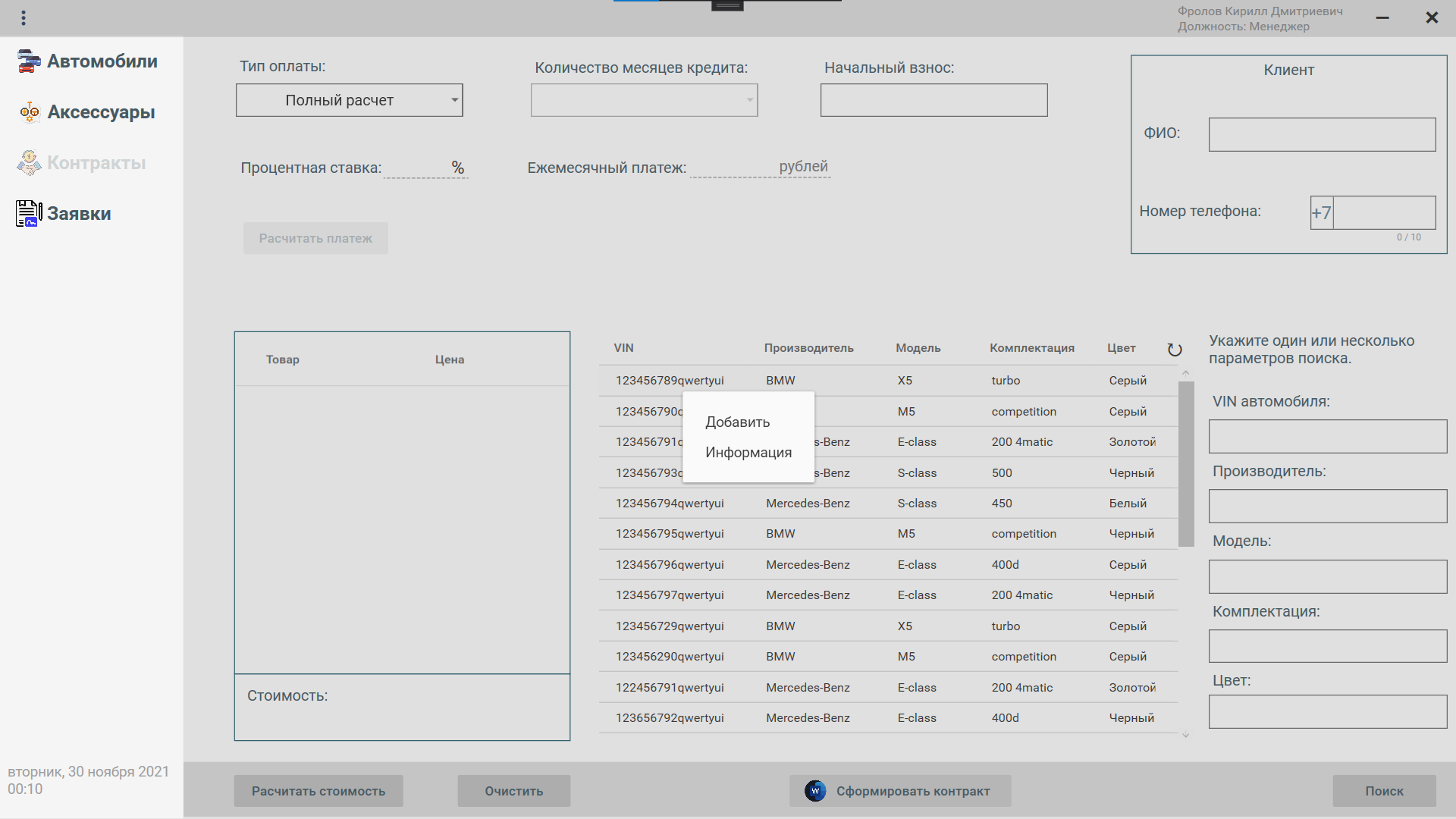


Рисунок 4.3 – Страница с договорами.

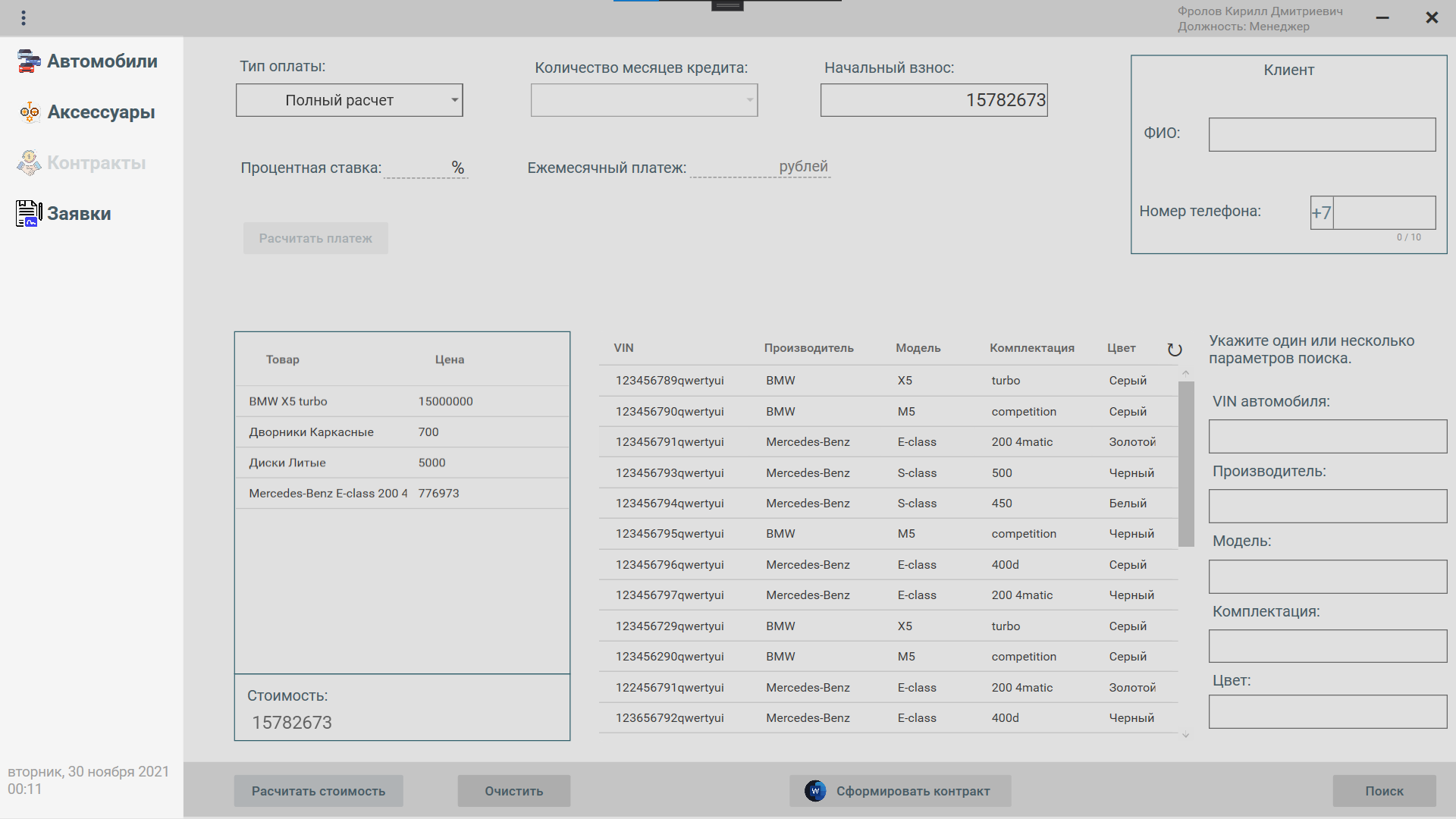


Рисунок 4.4 – Пример расчета стоимости и добавления автомобилей.

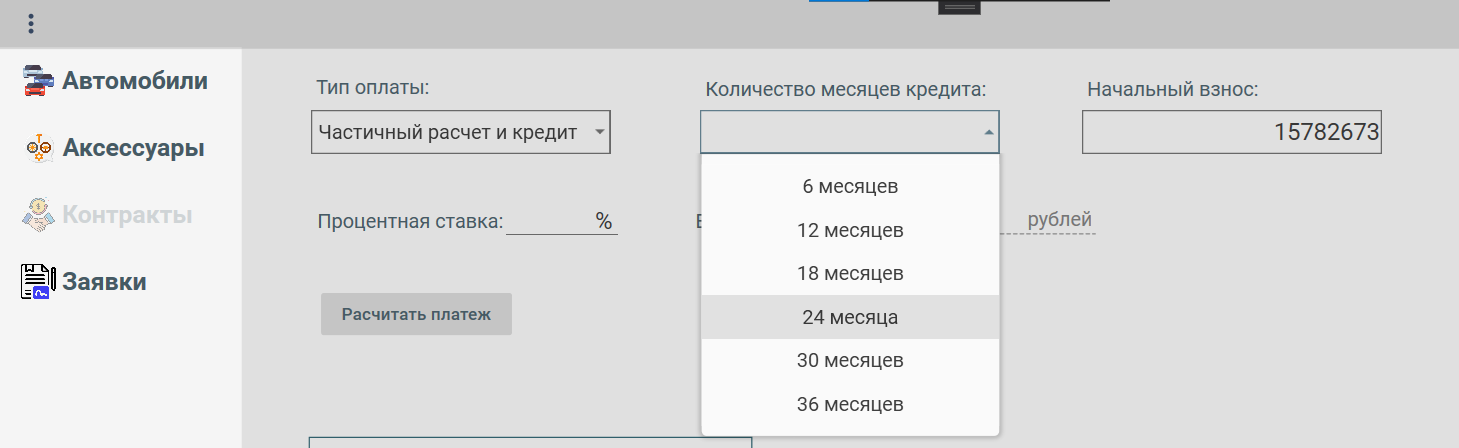


Рисунок 4.5 – Выбор длительности выплаты кредита.

Расчет ежемесячного платежа рассчитывается только при выборе типа оплаты «Частичный расчет и кредит» если начальный взнос отсутствует менеджер может ввести в это поле «0».

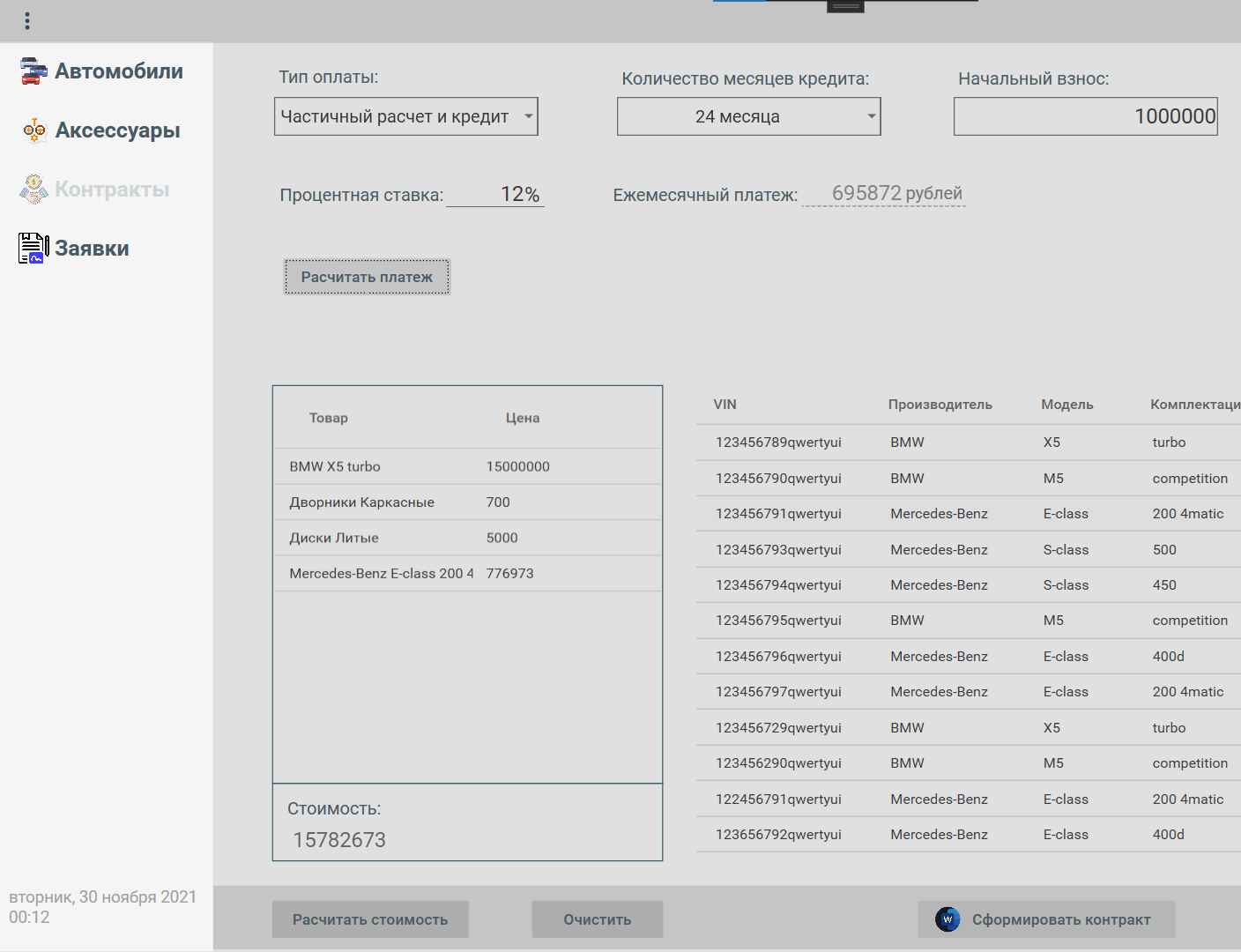


Рисунок 4.6 – Расчет ежемесячного платежа.

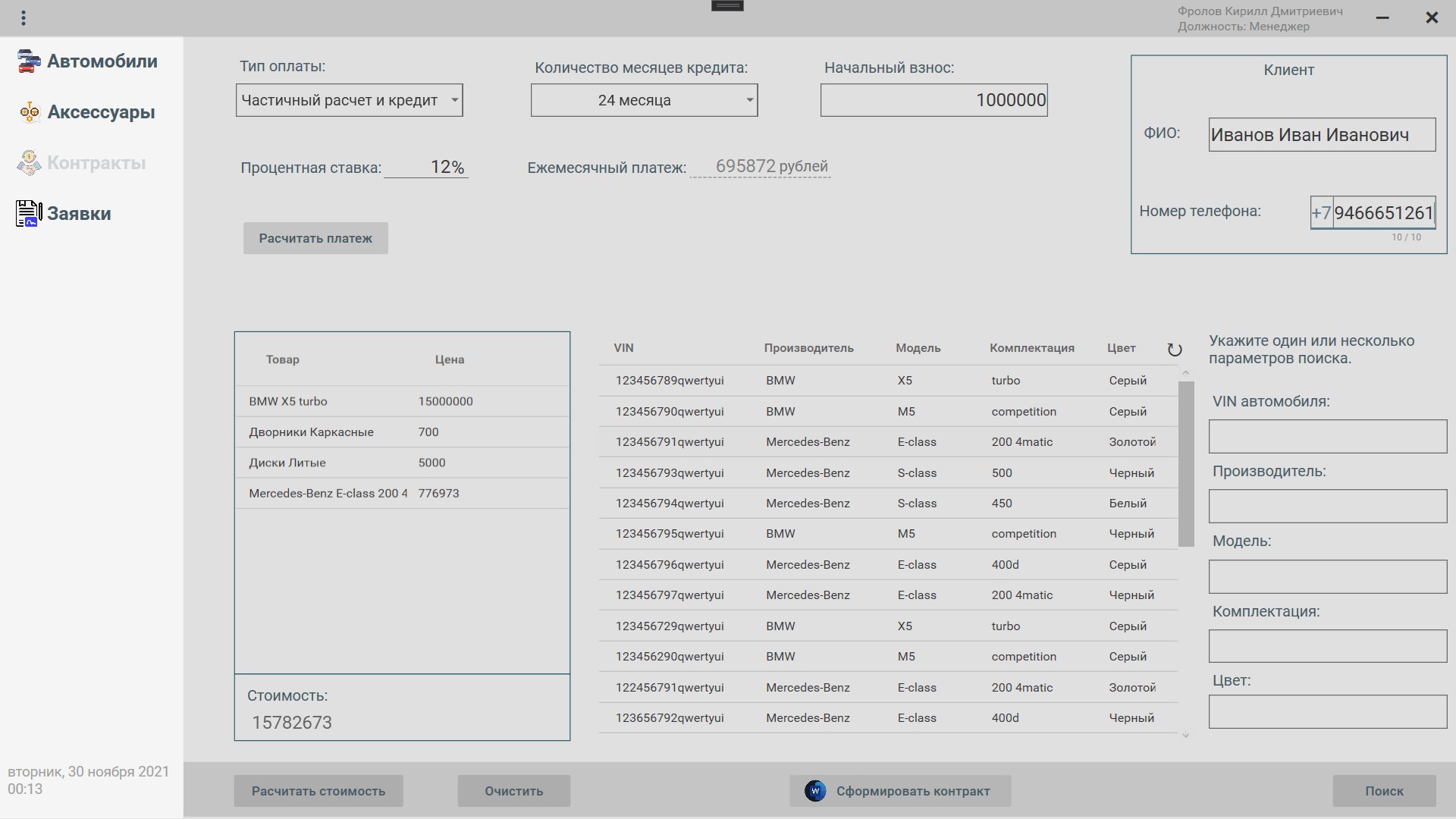


Рисунок 4.7 – Заполнение всех необходимых данных для формирования договора.

После того как все необходимые данные будут введены (система проигнорирует не введенные, ненужные данные) будет сформирован контракт (Рисунок 4.9)

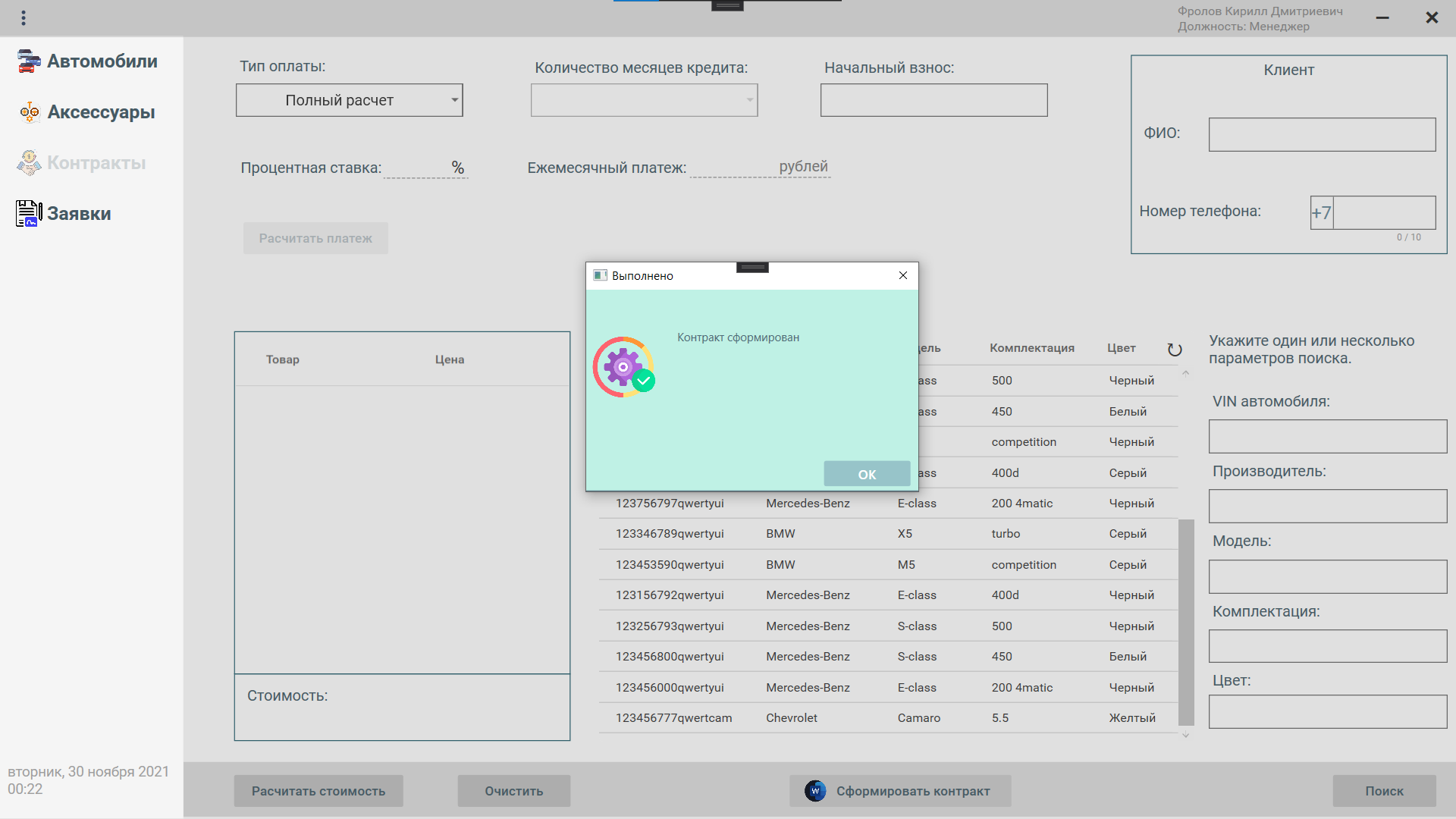


Рисунок 4.8 – Уведомление о сформированном контракте.

Договор формируется на основе введенных данных в системе, а также все данные о контракте будут занесены в базу данных.

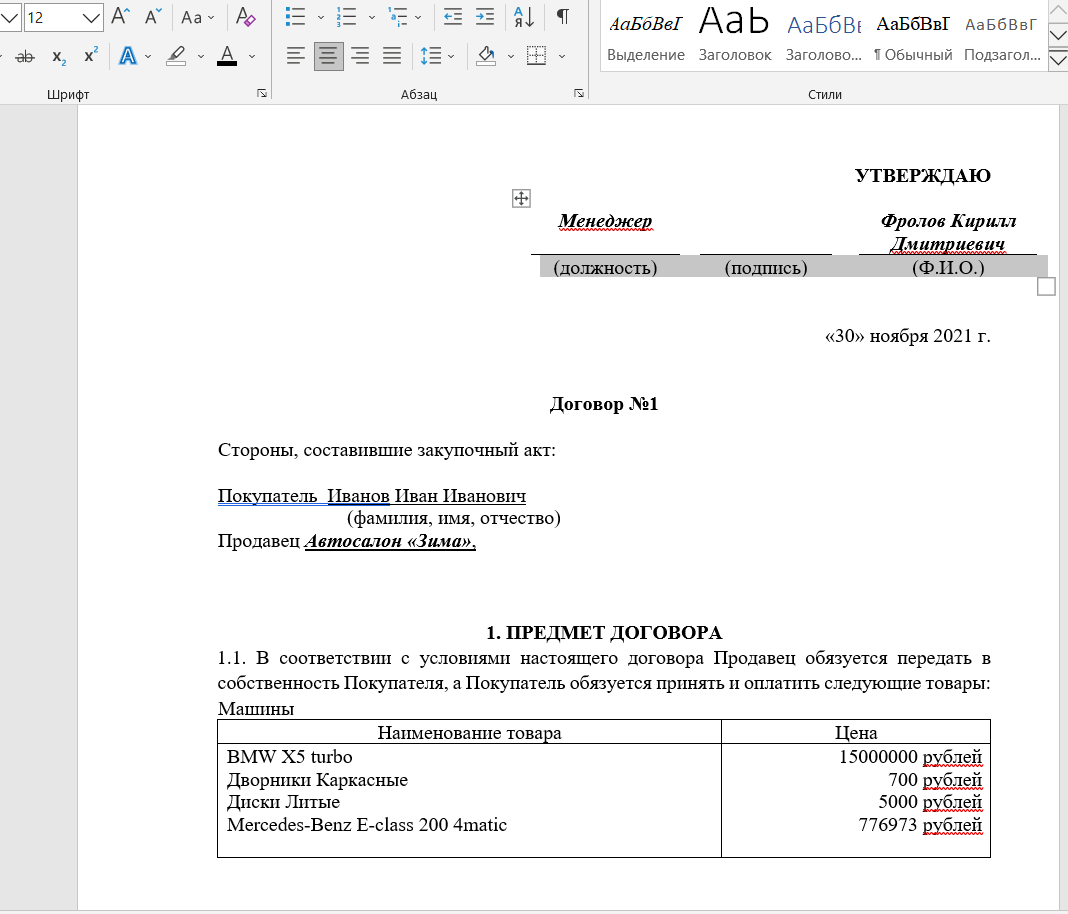


Рисунок 4.9 – Сформированный word документ.

Страница с заявками это два списка которые заполняет пользователь, все заявки записываются в таблицу базы данных, а также формируется документ

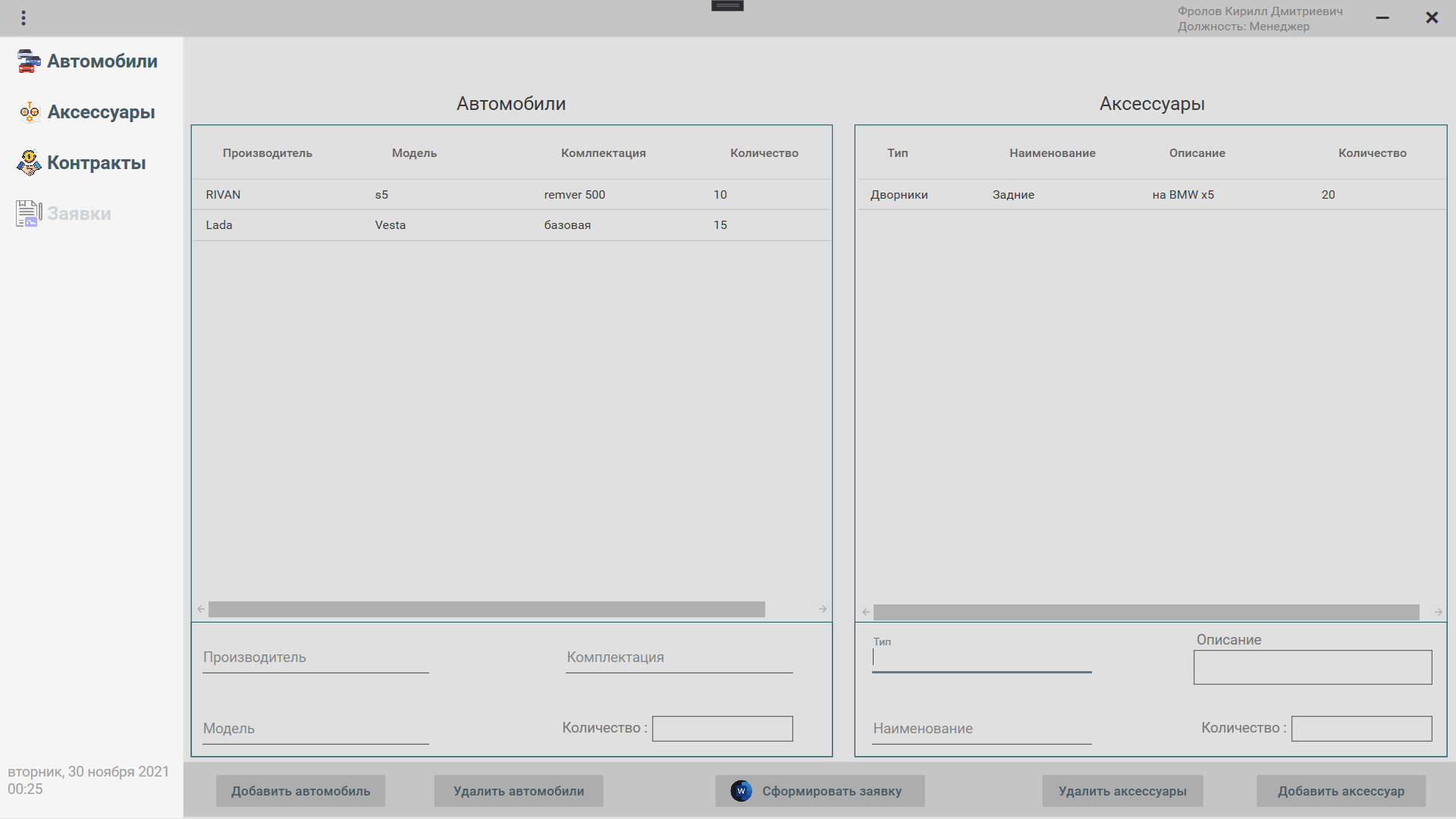


Рисунок 4.10 –Заполнение данных для формирования заявки.

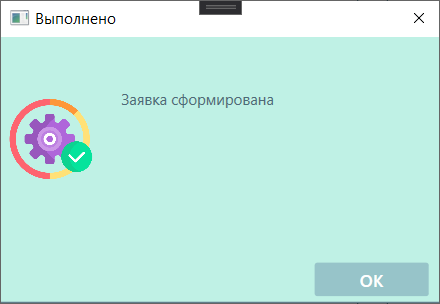


Рисунок 4.11 – Сообщение о сформированной заявке.

После нажатия на кнопку «Сформировать заявку» получаем 2 word документа, где первый заявка на автомобили, а второй на аксессуары

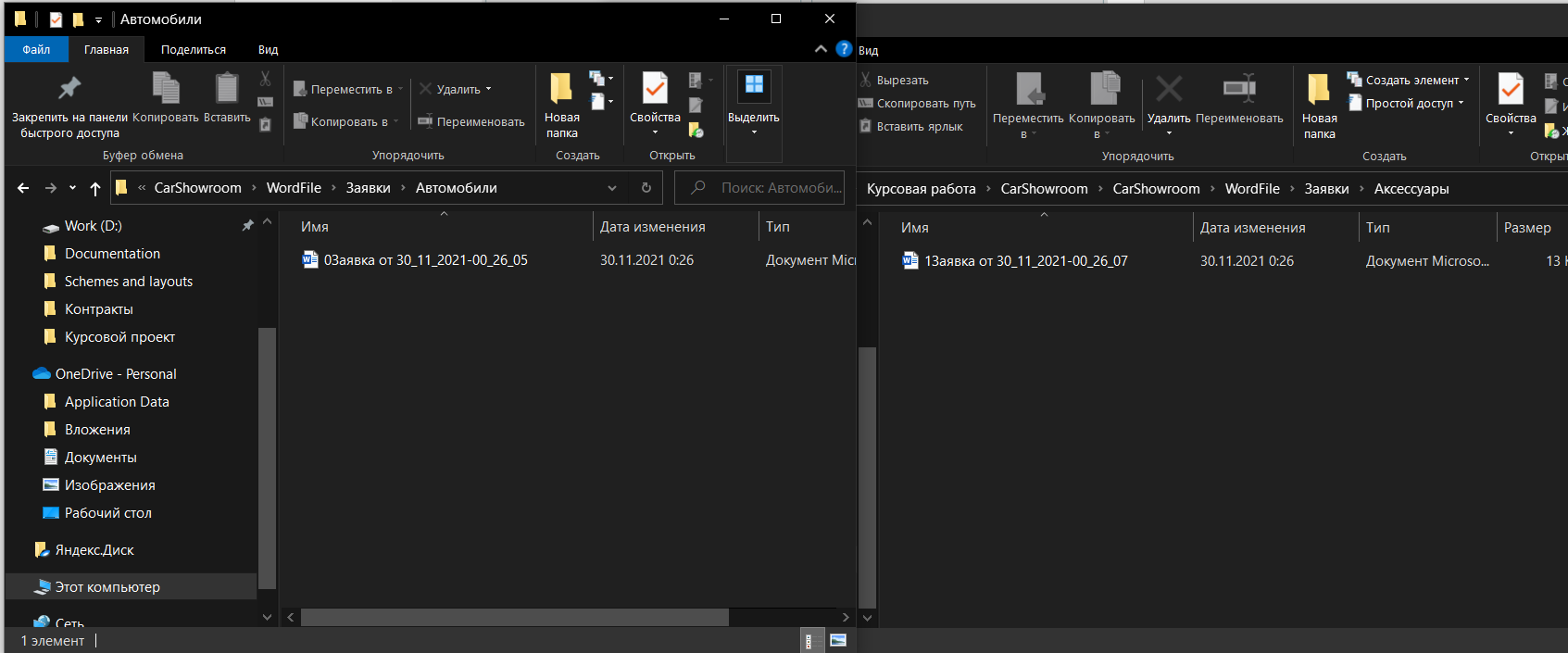


Рисунок 4.12 – 2 сформированных файла для автомобилей и аксессуаров.

Поиск в системе осуществляется по нескольким параметрам, для каждой страницы параметры уникальны.

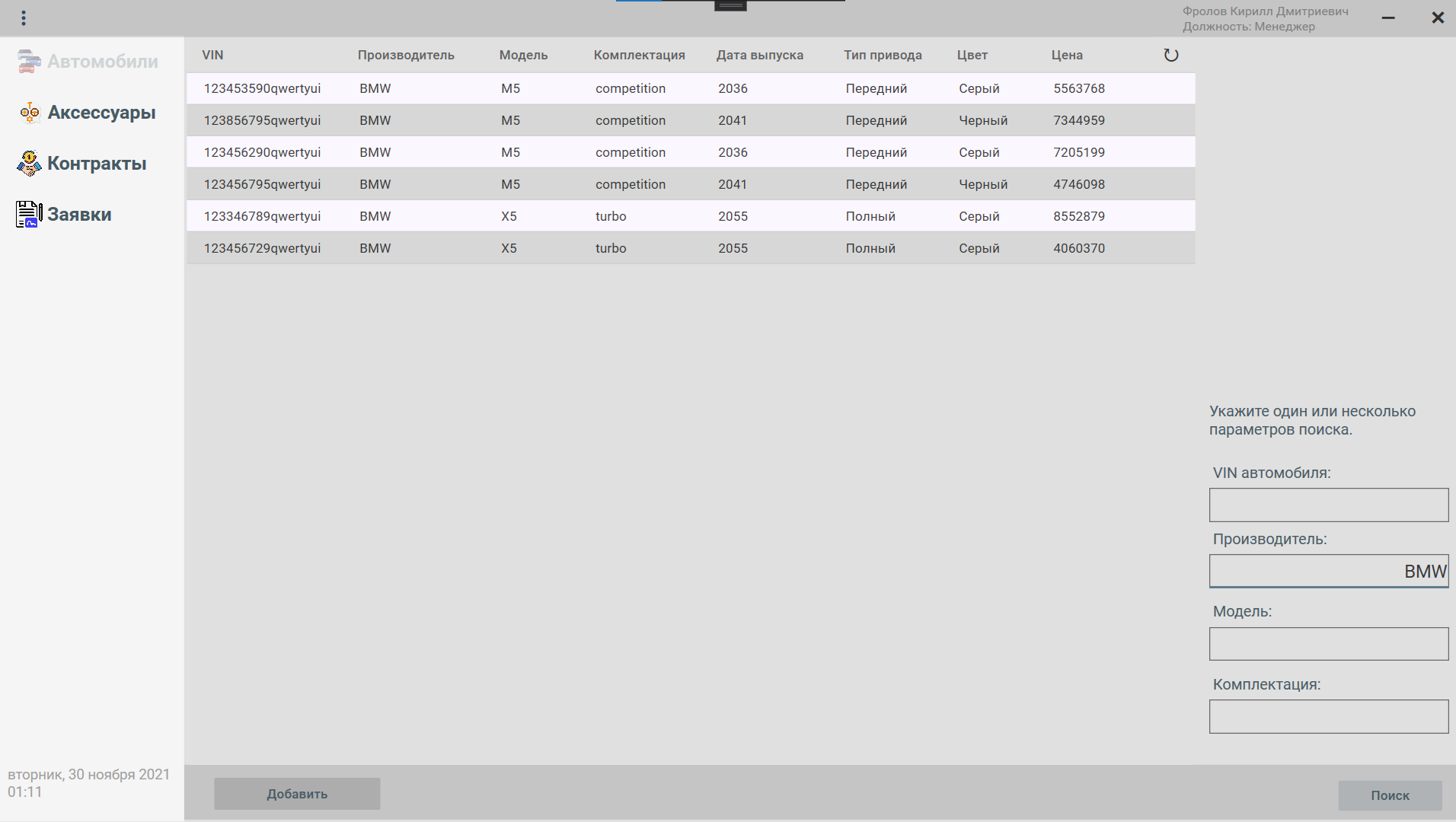


Рисунок 4.13 – Поиск автомобилей.

Для того чтобы добавить автомобиль в базу данных нужно нажать на кнопку «добавить» на странице с автомобилями, и появится окно (Рисунок 4.14). на странице можно добавить производителя, модель, комплектацию и автомобиль, чтобы добавить автомобиль нужно нажать двойным кликом по производителю, модели и комплектации или нажать правую кнопку мыши и вся необходимая информация добавиться в заблокированные поля, в активных полях пользователь должен сам заполнить информацию.

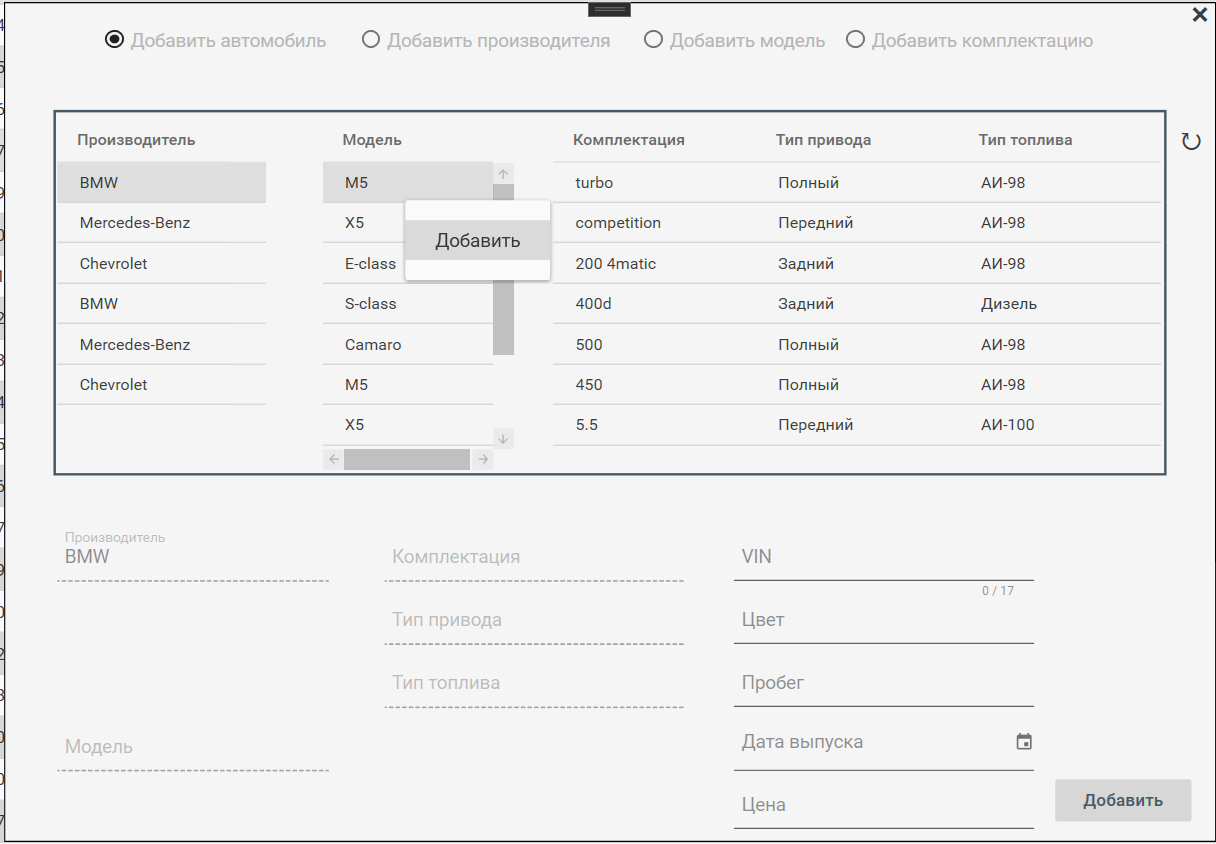


Рисунок 4.14 – Пример добавления автомобиля.

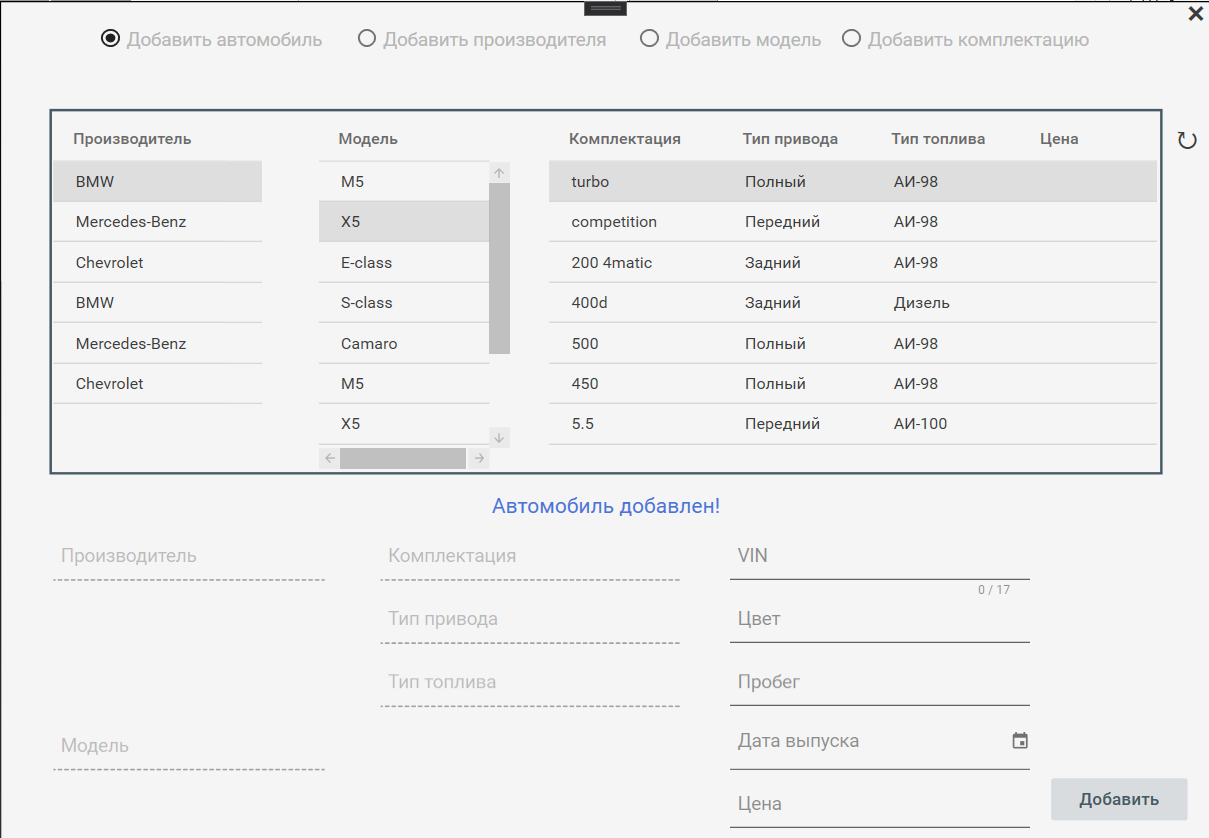


Рисунок 4.15 – Добавление автомобиля произошло успешно.

Так как автомобили могут продаваться с уже встроенными аксессуарами, в системе предусмотрено добавление аксессуаров к автомобилям

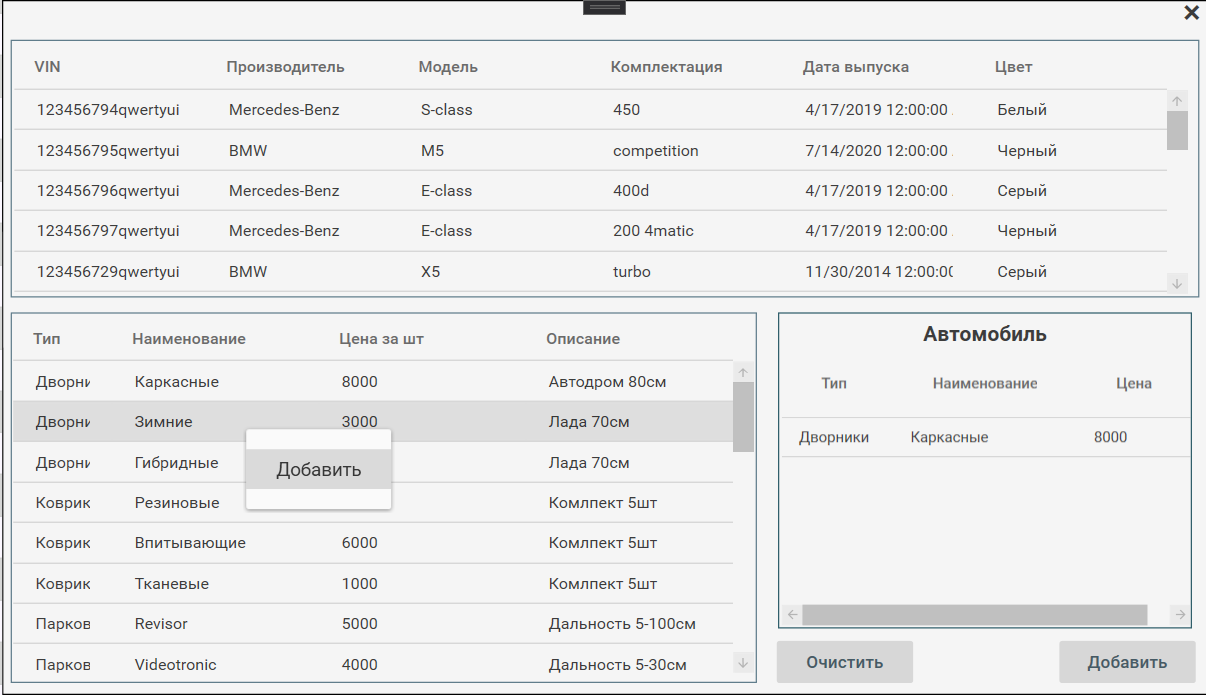


Рисунок 4.16 – Пример добавления аксессуара к автомобилю.



Рисунок 4.17 – Успешное добавление аксессуара к автомобилю.

Добавить аксессуар можно на странице с аксессуарами, нажав на кнопку «Добавить» откроется окно (Рисунок 4.18). Принцип добавления похож на добавление автомобилей.

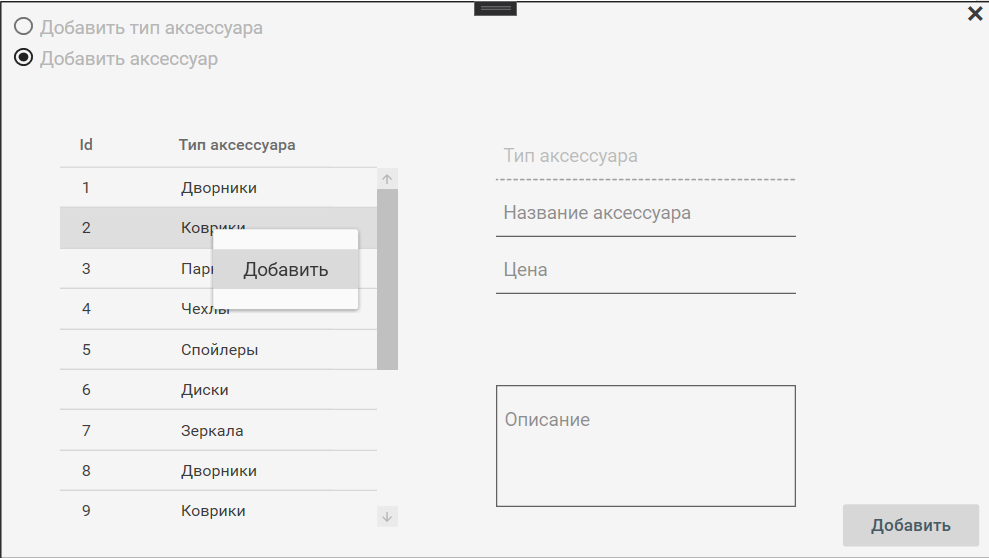


Рисунок 4.18 – Пример добавления аксессуара.

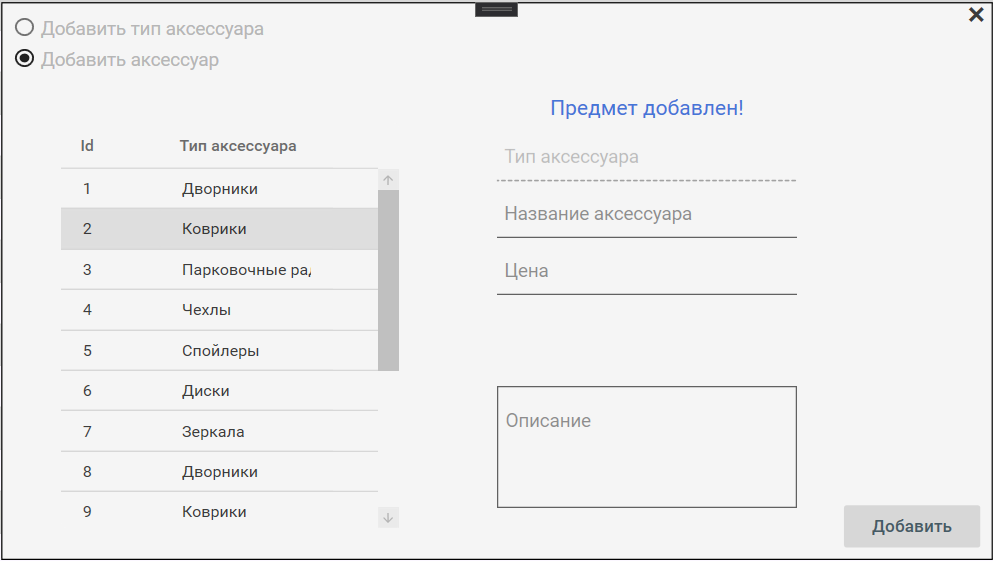


Рисунок 4.19 – Успешное добавление аксессуара.

В журнал событий записывается все действия которые произошли в системе и в базе данных, для удобного поиска нужной информации реализован поиск по дате, сортировка и выбор по критериям.

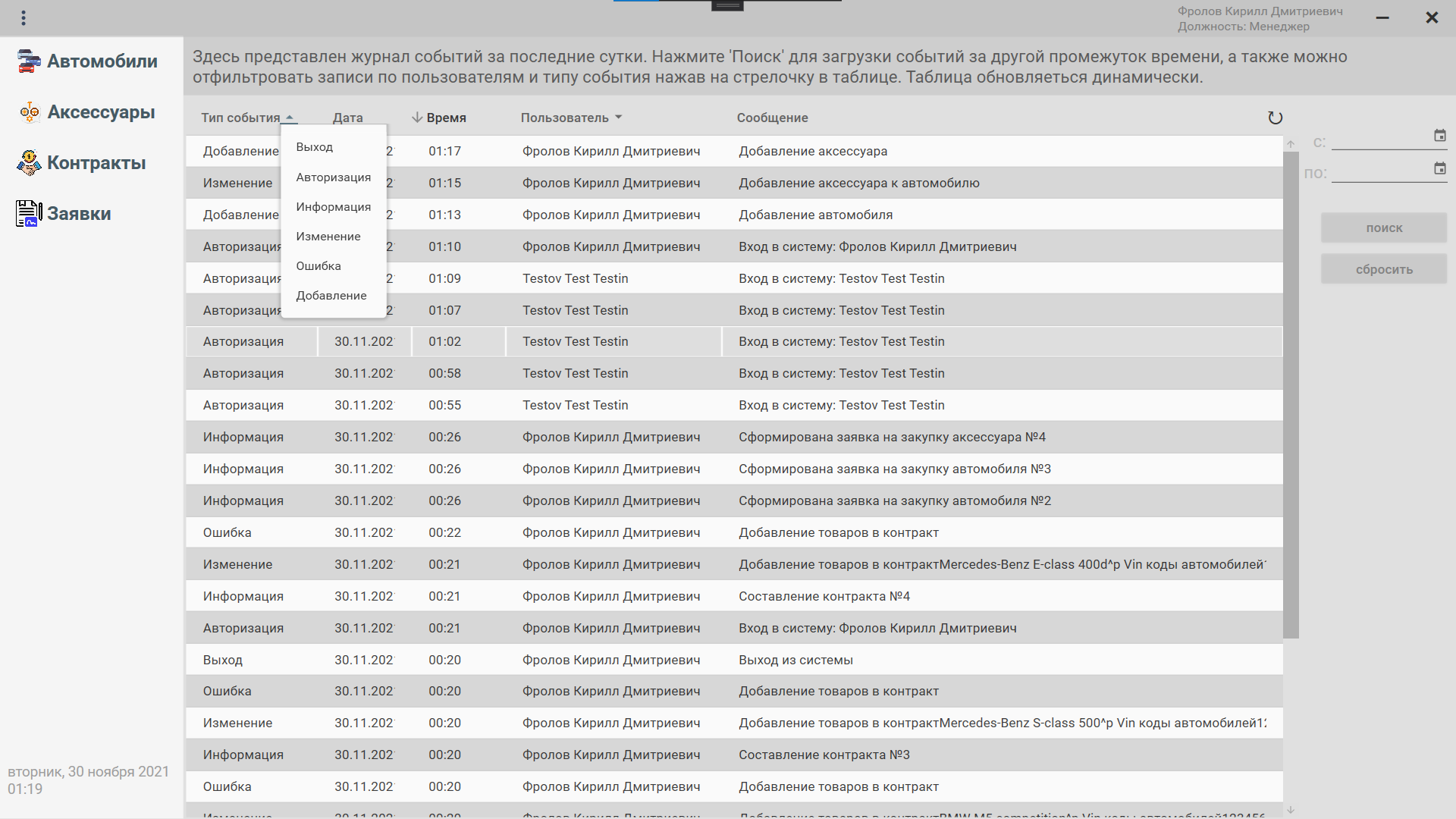


Рисунок 4.20 – Список типов событий в журнале событий.

На рисунке 4.21 был выбран тип события «Информация»

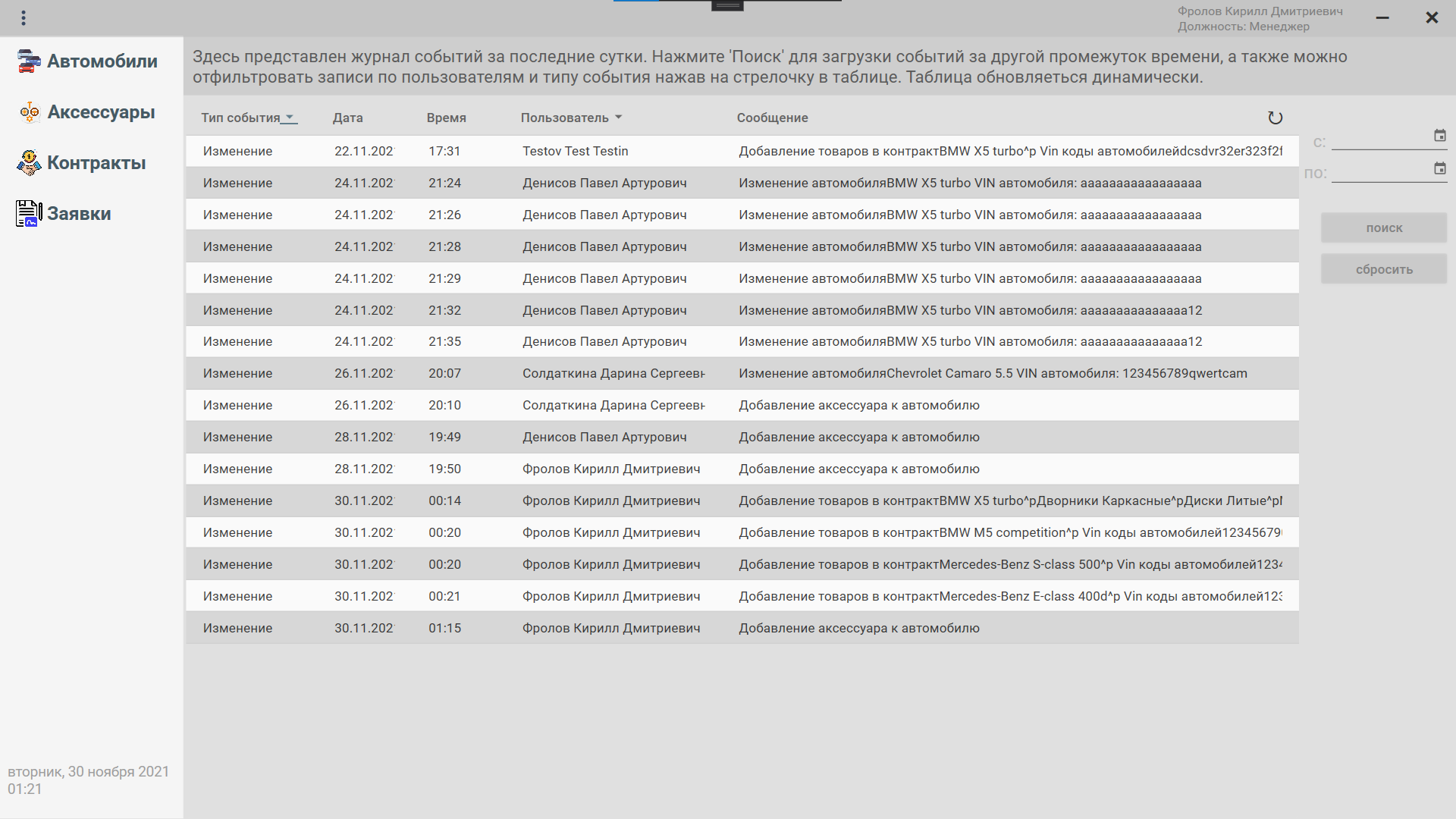


Рисунок 4.21 – Выбор типа события.

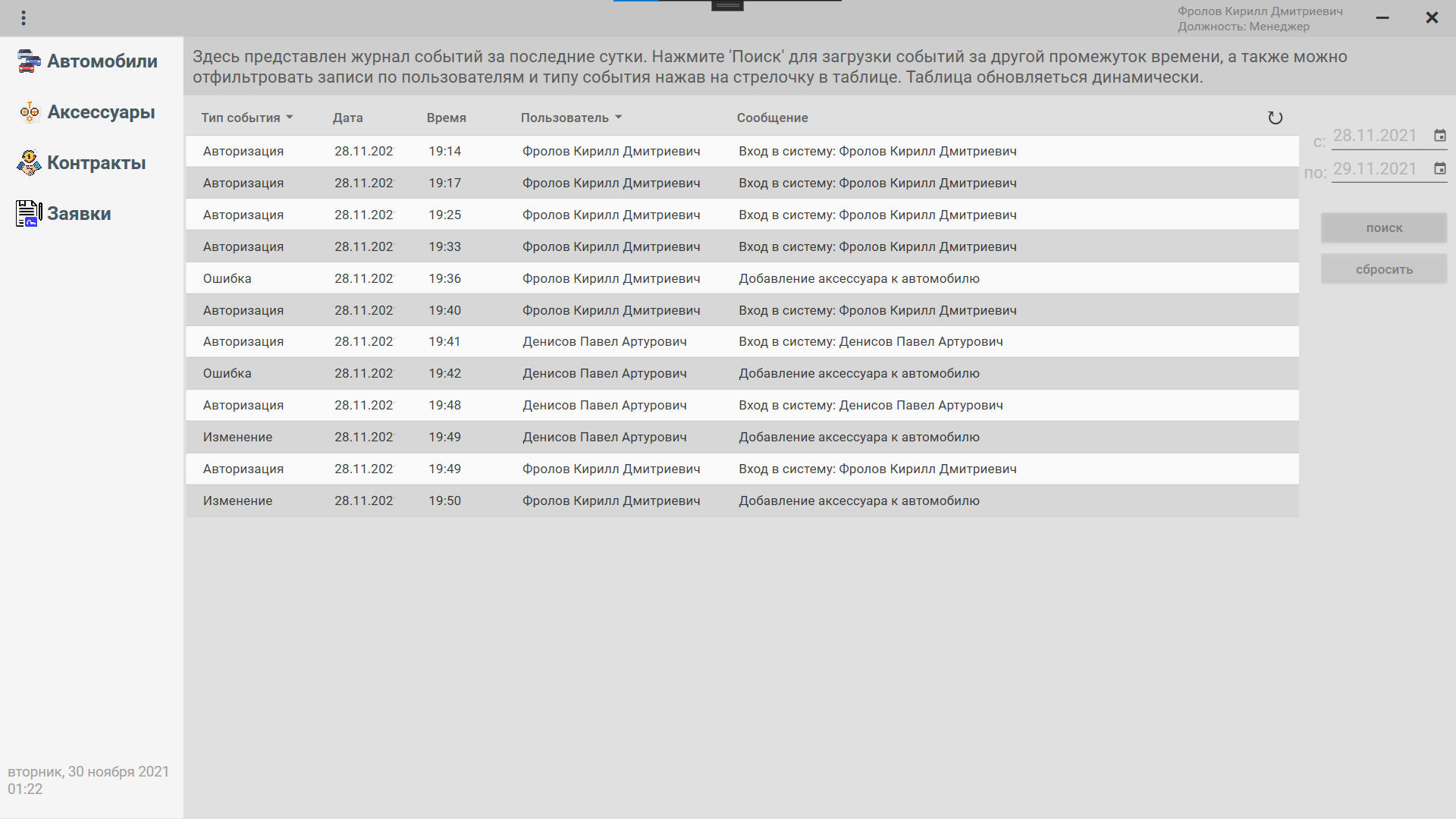


Рисунок 4.22 – Поиск по дате.

## 4.2 Технологические решения, поддерживающие эксплуатационный цикл программы.

Чтобы обеспечить быстро действие программы было принято решение, загружать все необходимые данные на этапе авторизации, это позволило делать быстрый переход между страницами, и не ждать загрузки при переходах.

Шаблон MVVM позволил прикреплять данные к интерфейсу, в свою очередь это повлияло на синхронность данных, а также на модель программы.

Entity Framework вместе с языком интегрированных запросов (LINQ) позволил обращаться не к атрибутам и таблицам из базы данных, а к объектам. Основными плюсами являются:

* Отсутствует проблема несоответствия модели данных в приложении и БД. Все данные сохраняются в БД в том же виде, что и в модели приложения.
* Не требуется отдельно поддерживать модель данных на стороне СУБД.
* Все объекты на уровне источника данных строго типизированы. Рефакторинг объектно-ориентированной базы данных и работающего с ней кода автоматизированный.

Пример запроса, из базы данных:

public Car Get(string vin)

{

using (var db = new CarShowroomContext())

{

var query = db.Cars.FirstOrDefault(x => x.Vin == vin);

return query;

}

}

Запрос возвращает объект типа Car, выборка происходит по уникальному номеру VIN.

В этом методе происходит изменение в базе данных, к автомобилю добавляется идентификатор контракта.

public Car Edit(string vin, int contractID)

{

using (var db = new CarShowroomContext())

{

var car = db.Cars.FirstOrDefault(x => x.Vin == vin);

if(car != null) {

car.ContractId = contractID;

db.Cars.Update(car);

db.SaveChanges();

}

return car;

}

}

## 4.3 Тестирование и отладка рабочей программы.

Для тестирования приложения были созданы тест-кейсы основных окон и страниц приложения.

Таблица 4.1 – Тест-кейс окна авторизации.

|  |  |
| --- | --- |
| Номер | 1 |
| Заголовок | Авторизация |
| Предусловие | Запустить приложение |
| Шаг | Ожидаемый результат |
| Ввести неправильный логин или пароль и нажать кнопку "ок" | Сообщение об ошибке |
| Ввести корректный логин или пароль и нажать кнопку "ок" | Переход на главную страницу приложения |
| Нажать на кнопку "отмена" | Закрытие приложения |
| Нажать на кнопку "Регистрация" | Открытие нового окна "Регистрация" |

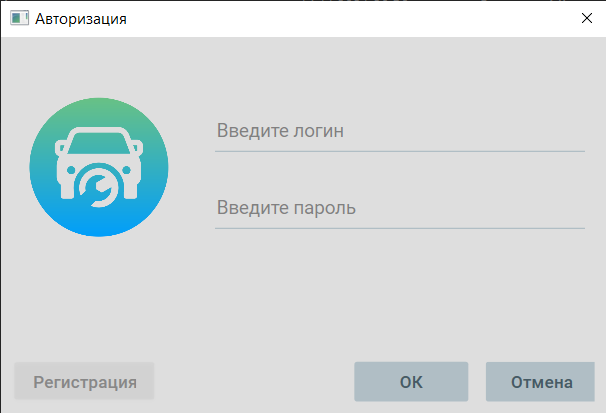


Рисунок 4.23 – Окно авторизации

Таблица 4.2 – Тест-кейс окна регистрации.

|  |  |
| --- | --- |
| Номер | 2 |
| Заголовок | Регистрация |
| Предусловие | Нажать на кнопку "регистрация" в окне авторизации |
| Шаг | Ожидаемый результат |
| Нажать на кнопку "отмена" | Закрытие окна с регистрацией |
| Заполнить все поля с уже существующим логином | Сообщение об уже существующем пользователе с таким логином |
| Ввести буквы в поле с номером телефона | Ввод не будет выполняться |
| Нажать на кнопку "Зарегистрироваться" с пустыми полями | сообщение об ошибке |
| Нажать на кнопку "Зарегистрироваться" скорректно заполнеными полями | сообщение об успешной регистрации, закрытие окна |

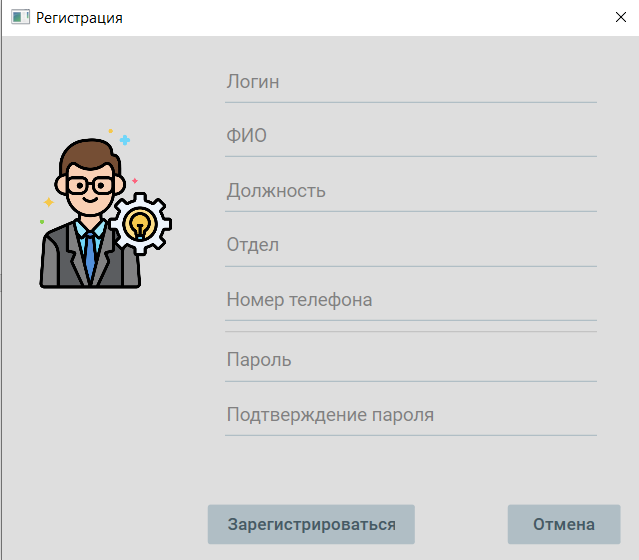


Рисунок 4.24 – Окно регистрации

Таблица 4.3 – Тест-кейс главного окна.

|  |  |
| --- | --- |
| Номер | 3 |
| Заголовок | Главное окно |
| Предусловие | Авторизоваться в приложении |
| **Шаг** | **Ожидаемый результат** |
| Нажать на кнопку с крестиком | Закрытие приложения |
| Нажать на кнопку с подчеркиванием | Сворачивание приложения |
| Нажать на кнопку с тремя точками в левом верхнем углу | Вывод списка страниц с аккаунтом, журналом событий и кнопкой выхода из приложения |
| Нажать на кнопку Автомобили | Открытие новой страницы "Автомобили" |
| Нажать на кнопку Аксессуары | Открытие новой страницы "Аксессуары" |
| Нажать на кнопку Контракты | Открытие новой страницы "Контракты" |
| Нажать на кнопку Заявки | Открытие новой страницы "Заявки" |

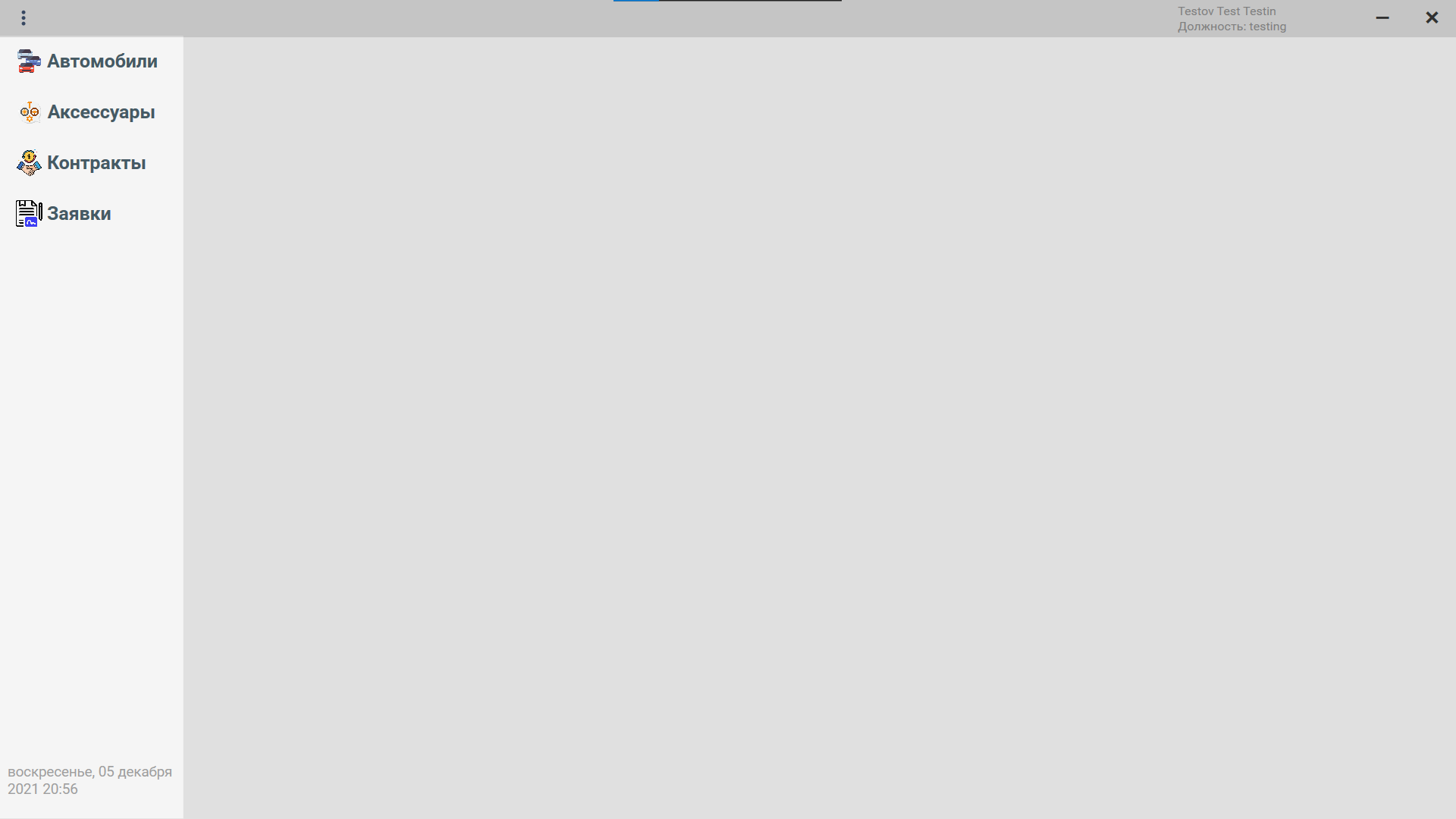


Рисунок 4.25 – Главное окно

Таблица 4.4 – Тест-кейс страницы «Автомобили».

|  |  |
| --- | --- |
| Номер | 4 |
| Заголовок | Станица автомобили |
| Предусловие | Нажать на кнопку автомобили |
| Шаг | Ожидаемый результат |
| Нажать правой кнопкой мыши по элементу таблицы "автомобили" | открытие списка с кнопками "Удалить" и "Информация" |
| Нажать на кнопку "Добавить" | Открытие нового окна, для добавления автомобилей в базу данных |
| Нажать на кнопку "Обновить" | Обновление таблицы "автомобили", если новый автомобиль был добавлен то после обновления он появиться в таблице |
| Ввести существующее значение в поле "Vin" и нажать на кнопку поиск | Сброс значения в поле "Vin" и вывод в таблице "автомобили" автомобилия с задаными vin номером |
| Ввести не существующее значение в поле "Vin" и нажать на кнопку поиск | Сброс значения в поле "Vin" и отсутствие значений в таблице "автомобили" |
| Оставить пустые поля параметров поиска и нажать на кнопку "поиск" | Вывод всех автомобилей в таблице "автомобили" |

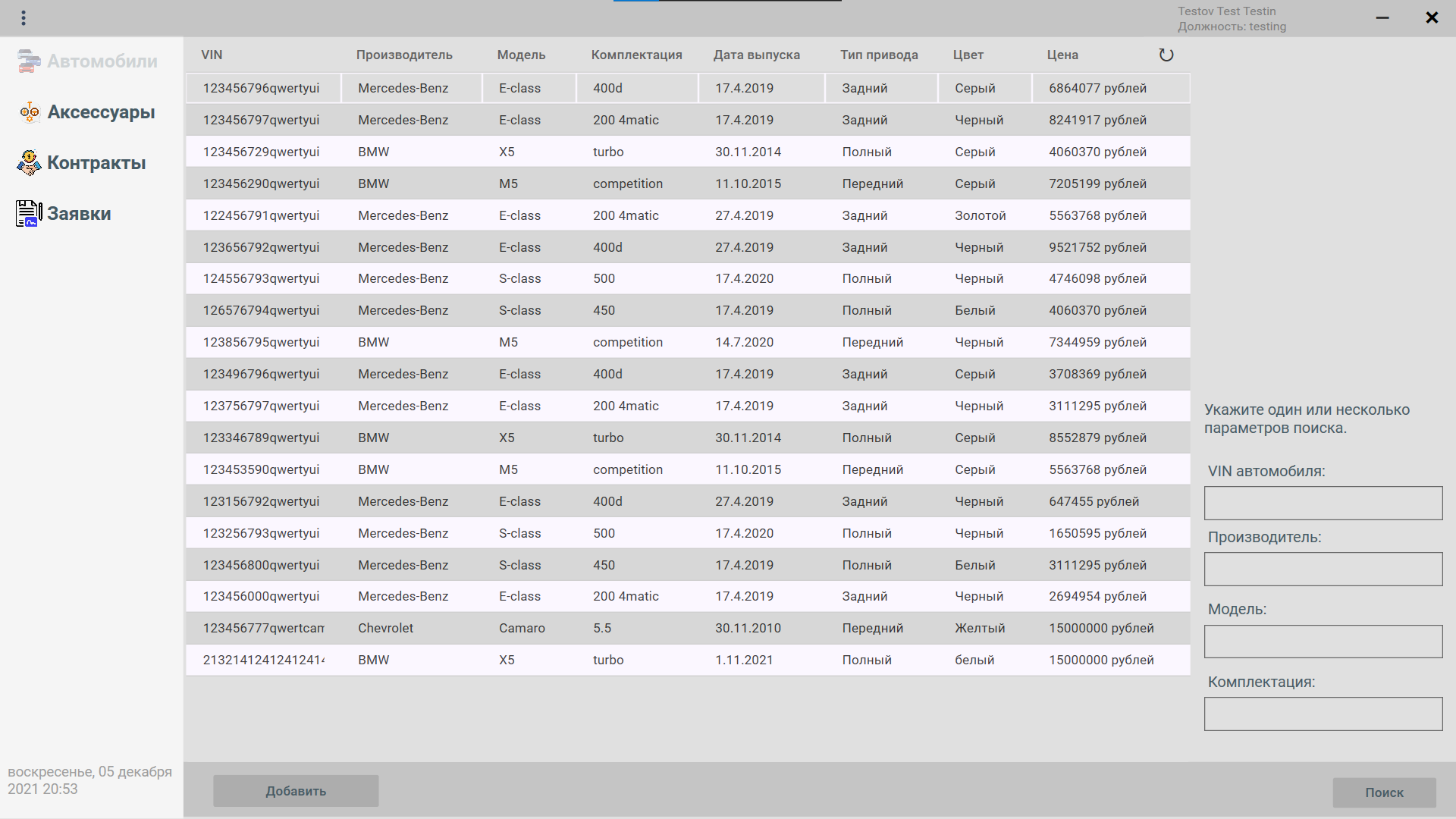


Рисунок 4.26 – Страница «Автомобили»

Таблица 4.5 – Тест-кейс страницы «Аксессуары».

|  |  |
| --- | --- |
| Номер | 5 |
| Заголовок | Станица Аксесуары |
| Предусловие | Нажать на кнопку аксессуары |
| Шаг | Ожидаемый результат |
| Нажать правой кнопкой мыши по элементу таблицы "автомобили" | открытие списка с кнопкой "Удалить" |
| Нажать на кнопку "Добавить" | Открытие нового окна, для добавления аксессуаров в базу данных |
| Нажать на кнопку "Обновить" | Обновление таблицы "аксессуарыи", если новый аксессуар был добавлен то после обновления он появиться в таблице |
| Ввести существующее значение в поле "Тип" и нажать на кнопку поиск | Сброс значения в поле "Тип" и вывод в таблице "аксессуары" аксессуара с задаными типом |
| Ввести не существующее значение в поле "Тип" и нажать на кнопку поиск | Сброс значения в поле "Тип" и отсутствие значений в таблице "аксессуары" |
| Оставить пустые поля параметров поиска и нажать на кнопку "поиск" | Вывод всех аксессуаров в таблице "аксессуары" |
| Нажать на кнопку "Добавить аксессуар к автомобилю" | Открытие нового окна, для добавления аксессуаров к автомобилю |



Рисунок 4.27 – Страница «Аксессуары»

Таблица 4.6 – Тест-кейс страницы «Контракты».

|  |  |
| --- | --- |
| Номер | 6 |
| Заголовок | Станица Контракты |
| Предусловие | Нажать на кнопку контракты |
| Шаг | Ожидаемый результат |
| Выбрать тип оплаты "полный расчет" | Блокировка полей "количество месяцев кредита", "процентная ставка", "ежемесячный платеж" и кнопки "расчитать платеж" |
| Выбрать тип оплаты "Частичный расчет или кредит" | Разблокировка полей "количество месяцев кредита", "процентная ставка", "ежемесячный платеж" и кнопки "расчитать платеж" |
| Добавление товаров в чек: Нажать правой кнопкой мыши на товар в таблице и в открывшемся списке нажать на кнопку "Добавить" или нажать на товар двойным кликом левой кнопки мыши | В чеке появляеться наименование товара и его цена |
| Нажать на кнопку "расчитать стоимость" | В чеке появиться общаяя стоимость товаров |
| Нажать на кнопку "Расчитать платеж" когда все данные заполенны | В поле "ежемесячный платеж" появится сумма ежемесячного платежа, на основе введеных данных |
| Нажать на кнопку "Расчитать платеж" когда некоторые данные не заполенны | Сообщение об ошибке, указывающее какие данные отсутствуют |
| Нажать на кнопку "Сформировать контракт" когда все данные заполнены | Очистка всех заполненых полей, обновление таблицы и сообщение об успешном формировании контракта. Созданный файл с расширением .docx в котором присутствуют ранее заполенные данные |
| Ввести существующее значение в поле "Vin" и нажать на кнопку поиск | Сброс значения в поле "Vin" и вывод в таблице "автомобили" автомобилия с задаными vin номером |
| Ввести не существующее значение в поле "Vin" и нажать на кнопку поиск | Сброс значения в поле "Vin" и отсутствие значений в таблице "автомобили" |

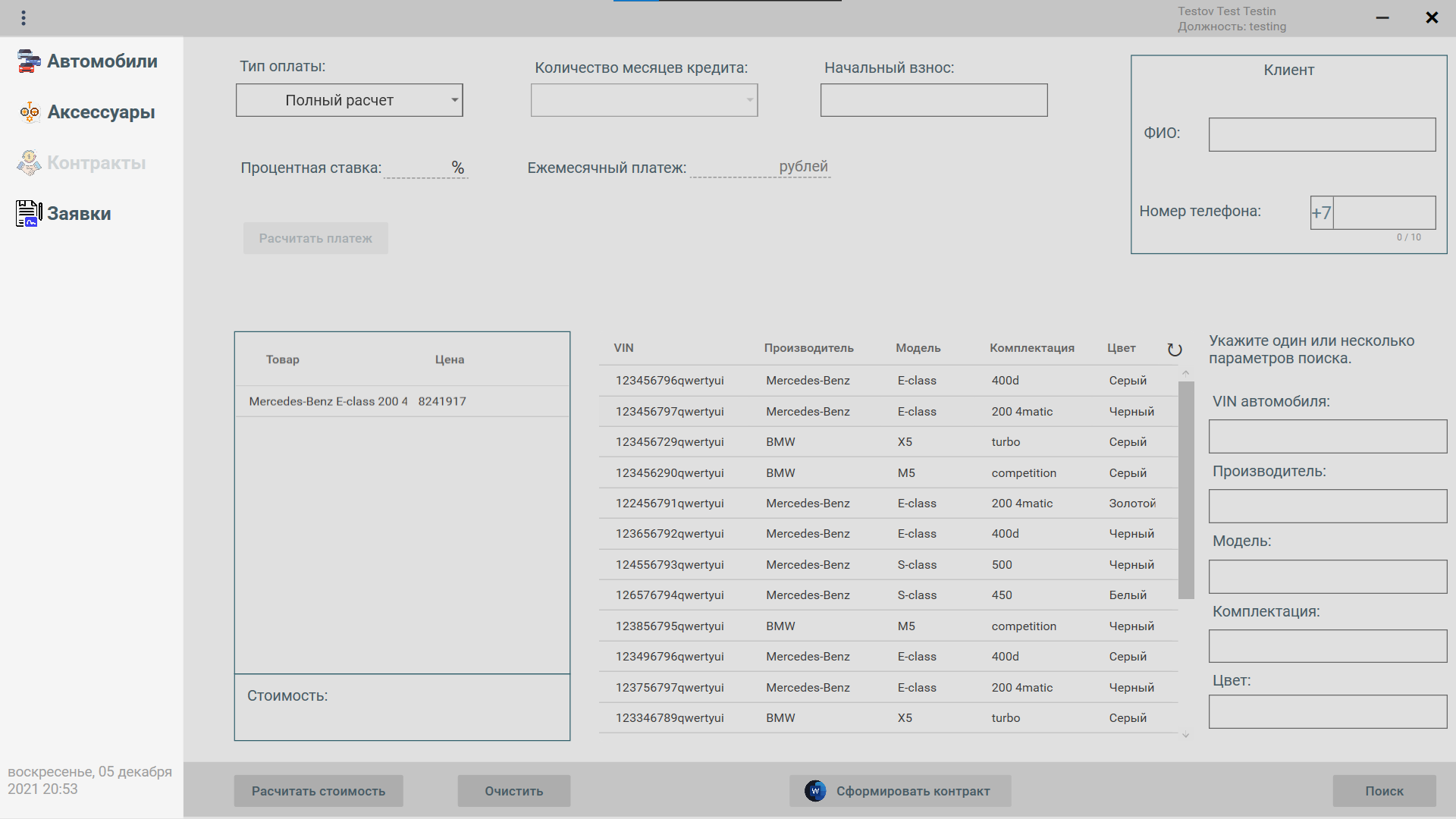


Рисунок 4.28 – Страница «Контракты»

Таблица 4.7 – Тест-кейс страницы «Заявки».

|  |  |
| --- | --- |
| Номер | 7 |
| Заголовок | Станица Заявки |
| Предусловие | Нажать на кнопку заявки |
| Шаг | Ожидаемый результат |
| Ввести в поле "количество" буквы | дынные не будут заполенны |
| Заполнить основную информацию и нажать на кнопку "Добавить автомобиль" | Добавление введенной информации в список с автомобилями |
| Заполнить основную информацию и нажать на кнопку "Добавить аксессуар" | Добавление введенной информации в список с аксессуарами |
| нажать правой кнопкой мыши по элементы списка | Вывод кнопки "удалить" |
| Нажать на кнопку "Удалить автомобили" | Список с автомобилями становиться пустым |
| Нажать на кнопку "Удалить аксессуары" | Список с аксессуарами становиться пустым |
| Нажать на кнопку "Сформировать заявку" предварительно заполнив необходимую информацию | Сообщение об успешном формировании заявки. Создание файла с рисширением .docx в котором находиться введенная информация |

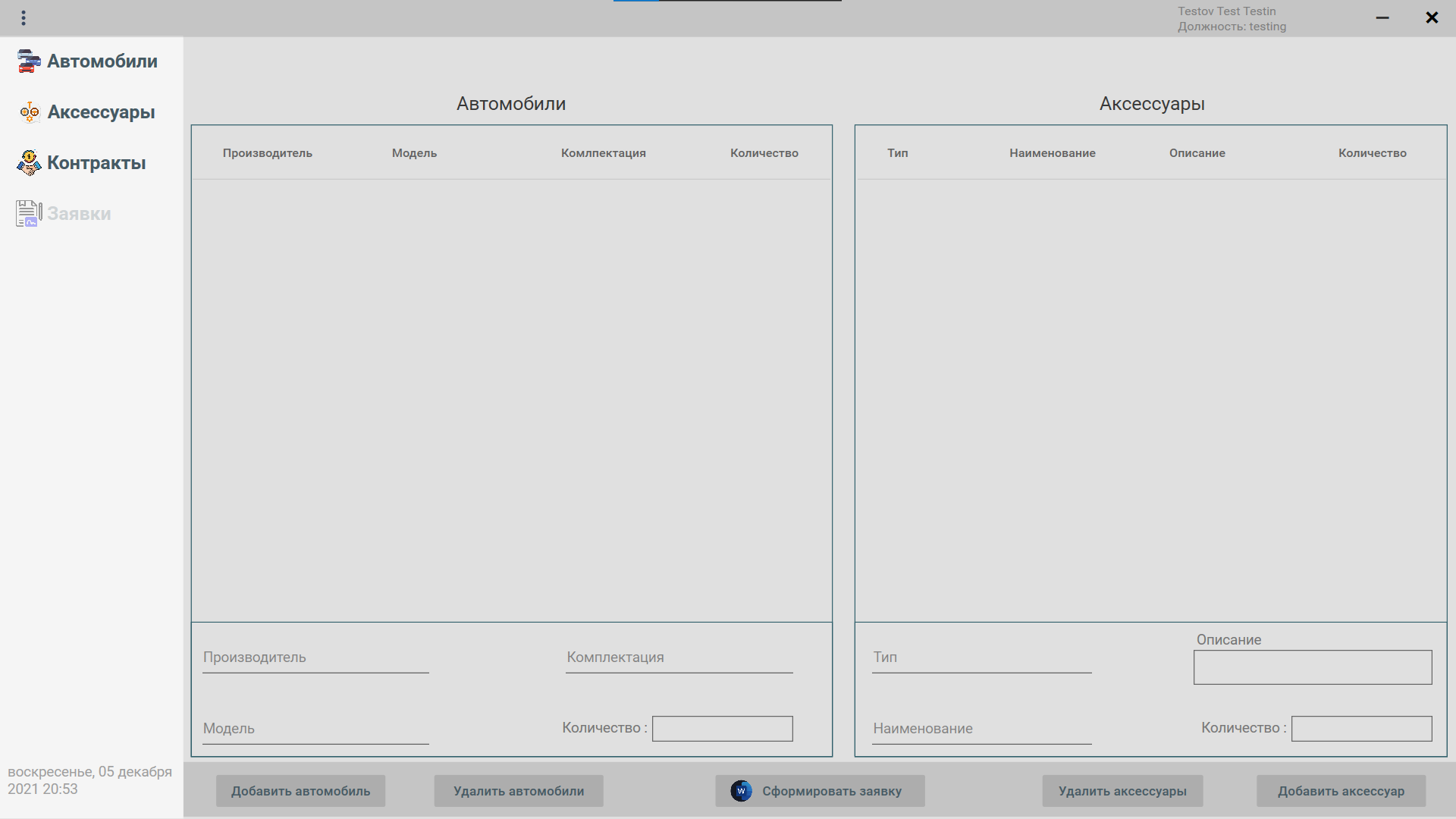


Рисунок 4.29 – Страница «Заявки»

# 5. ЗАКЛЮЧЕНИЕ

В курсовой работе было рассмотрено проектирование и разработка базы данных "Автосалон", которая позволяет хранить основную информацию о продажах, автомобилях, их аксессуарах и заявках. На начальном этапе была построена логическая модель базы данных, на основе которой была разработана физическая модель. А также была исследована и описана предметная область, проведен анализ аналогов данной системы.

Проведен анализ объектов автоматизации и разработаны методы решения технических задач. Также была разработана и реализована структура базы данных. База данных была наполнена тестовыми данными.

# 6. СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТУРАТУРЫ

Привязка данных с помощью WPF [Электронный ресурс] – URL: https://docs.microsoft.com/ru-ru/ef/ef6/fundamentals/databinding/wpf

Работа с базами данных на языке С#. Технология ADO.net. Учебное пособие. Ульяновск 2009

Документация PostgreSQL [Электронный ресурс] – URL: <https://postgrespro.ru/docs>

Шаблон Model-View-ViewModel [Электронный ресурс] – URL: <https://docs.microsoft.com/ru-ru/xamarin/xamarin-forms/enterprise-application-patterns/mvvm>

Райфайзенбанк как расчитать процент по кредиту [Электронный ресурс] – URL: https://www.raiffeisen.ru/wiki/kak-rasschitat-procenty-po-kreditu/

# ПРИЛОЖЕНИЕ 1

Сущность «Производитель»

CREATE TABLE manufacturer (

id serial PRIMARY KEY,

carbrand VARCHAR(100) NOT NULL);

Сущность «Модель»

CREATE TABLE model (

id serial PRIMARY KEY,

manufacturer\_id INTEGER,

name\_model VARCHAR(100) NOT NULL,

FOREIGN KEY (manufacturer\_id) REFERENCES manufacturer(id));

Сущность «Клиент»

CREATE TABLE customer (

id serial PRIMARY KEY,

FIO VARCHAR(100) NOT NULL,

phone\_number VARCHAR(100));

Сущность «Сотрудник»

CREATE TABLE employee (

id serial PRIMARY KEY,

FIO VARCHAR(100) NOT NULL,

departament VARCHAR(100) NOT NULL,

position VARCHAR(100) NOT NULL,

phone\_number VARCHAR(100),

login VARCHAR(100) UNIQUE,

password VARCHAR(100));

Сущность «Тип аксессуара»

CREATE TABLE type\_accessory (

id serial PRIMARY KEY,

name\_type\_accessory VARCHAR(100) NOT NULL);

Сущность «Аксессуар»

CREATE TABLE accessories (

id serial PRIMARY KEY,

type\_accessory\_id INTEGER,

car\_id INTEGER,

name\_accessory VARCHAR(100) NOT NULL,

price INTEGER NOT NULL,

description TEXT,

FOREIGN KEY (type\_accessory\_id) REFERENCES type\_accessory (id),

FOREIGN KEY (car\_id) REFERENCES car (id));

Сущность «Комплектация»

CREATE TABLE equipment (

id serial PRIMARY KEY,

model\_id INTEGER,

name\_equipment VARCHAR(100) NOT NULL,

type\_drive VARCHAR(100) NOT NULL,

fuel VARCHAR(100) NOT NULL,

FOREIGN KEY (model\_id) REFERENCES model (id));

Сущность «Договор»

CREATE TABLE contract (

id serial PRIMARY KEY,

customer\_id INTEGER,

employee\_id INTEGER,

date TIMESTAMP NOT NULL,

pay\_method VARCHAR(100) NOT NULL,

initial\_donat\_money INTEGER CHECK (initial\_donat\_money > 0),

monthly\_pay INTEGER CHECK (monthly\_pay > 0),

count\_month\_installment INTEGER CHECK (count\_month\_installment > 0)

FOREIGN KEY (customer\_id) REFERENCES customer (id),

FOREIGN KEY (employee\_id) REFERENCES employee (id));

Сущность «Автомобиль»

CREATE TABLE car (

id serial PRIMARY KEY,

color VARCHAR(100),

equipment\_id INTEGER,

contract\_id INTEGER,

Vin VARCHAR(17) UNIQUE,

release\_date DATE NOT NULL,

price INTEGER NOT NULL,

FOREIGN KEY (equipment\_id) REFERENCES equipment (id),

FOREIGN KEY (contract\_id) REFERENCES contract (id));

Сущность «Журнал событий»

CREATE TABLE Eventslog(

id serial PRIMARY KEY,

type VARCHAR(100),

employee\_id INTEGER,

date TIMESTAMP NOT NULL,

event TEXT NOT NULL,

FOREIGN KEY (employee\_id) REFERENCES employee (id));

Сущность «Заявки»

CREATE TABLE Bid(

id serial PRIMARY KEY,

type VARCHAR(100),

employee\_id INTEGER,

name Text,

count INTEGER CHECK (count > 0),

FOREIGN KEY (employee\_id) REFERENCES employee (id));

1. Райфайзенбанк как расчитать процент по кредиту [Электронный ресурс] – URL: https://www.raiffeisen.ru/wiki/kak-rasschitat-procenty-po-kreditu/ [↑](#footnote-ref-1)