# 4. ПРОЕКТНО – ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

# . 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.

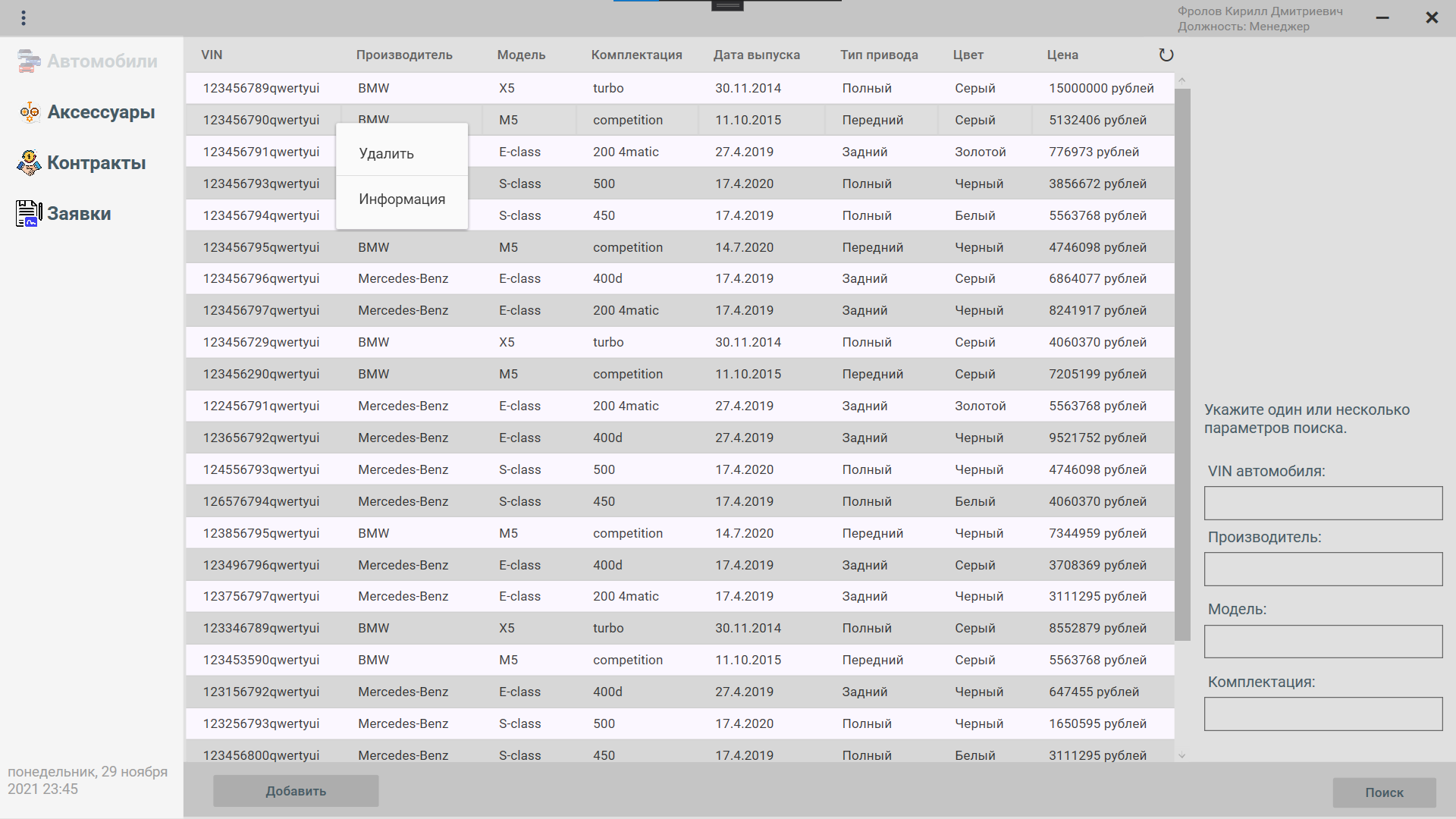


Рисунок 4.1 – Страница с автомобилями.



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.

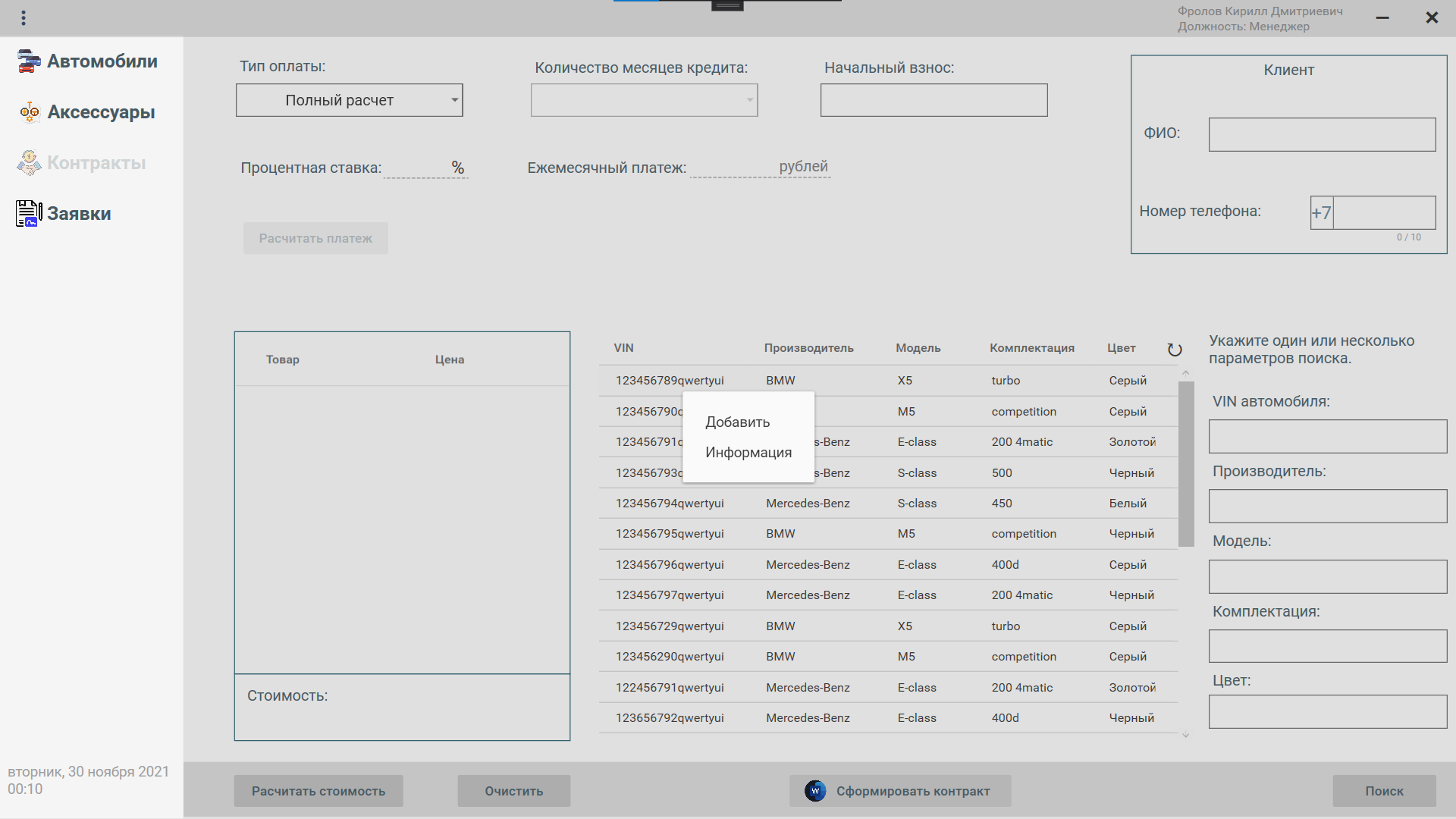


Рисунок 4.3 – Страница с договорами.

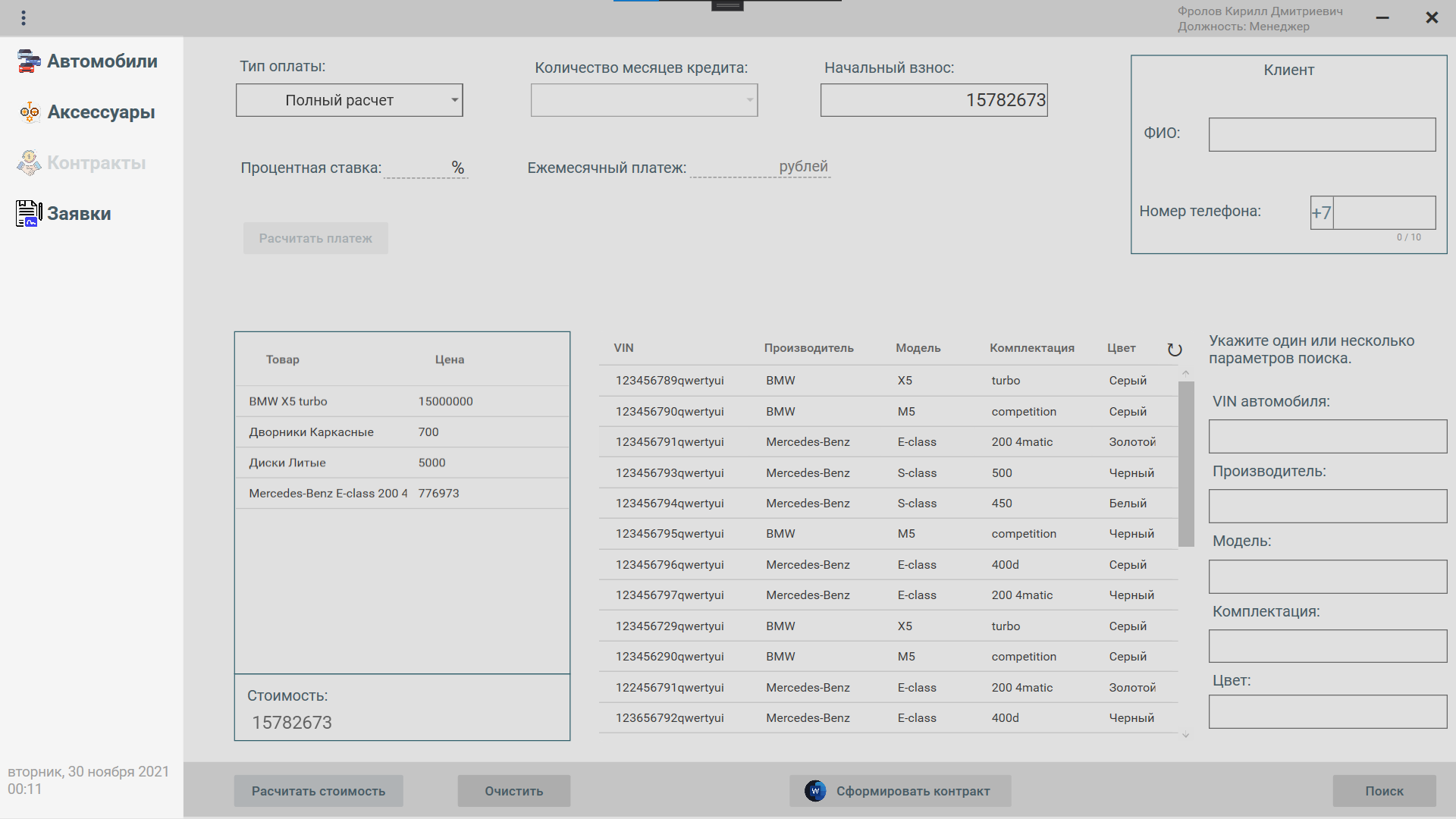


Рисунок 4.4 – Пример расчета стоимости и добавления автомобилей.

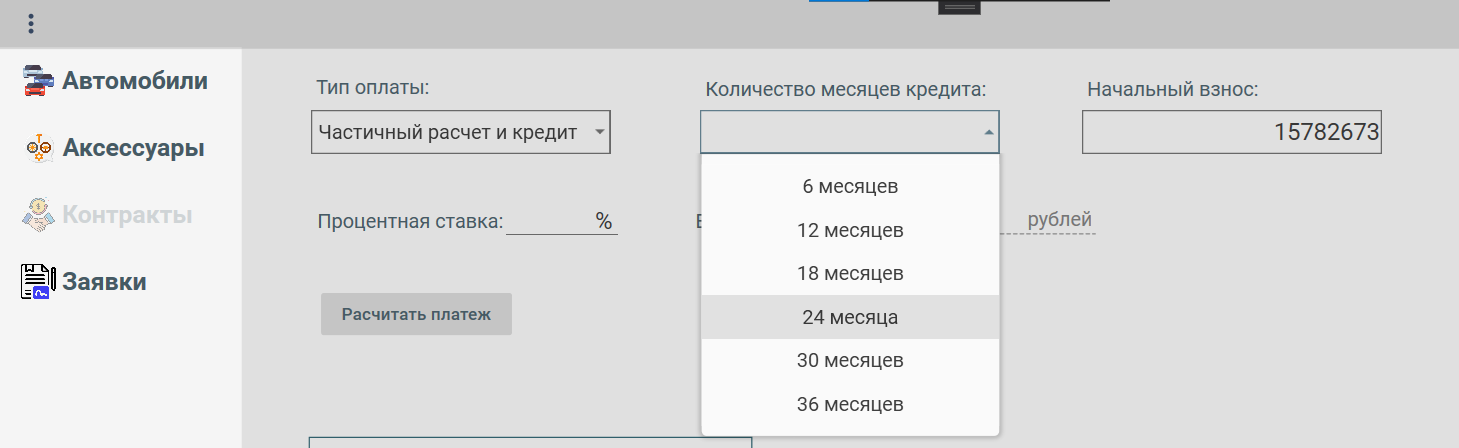


Рисунок 4.5 – Выбор длительности выплаты кредита.

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

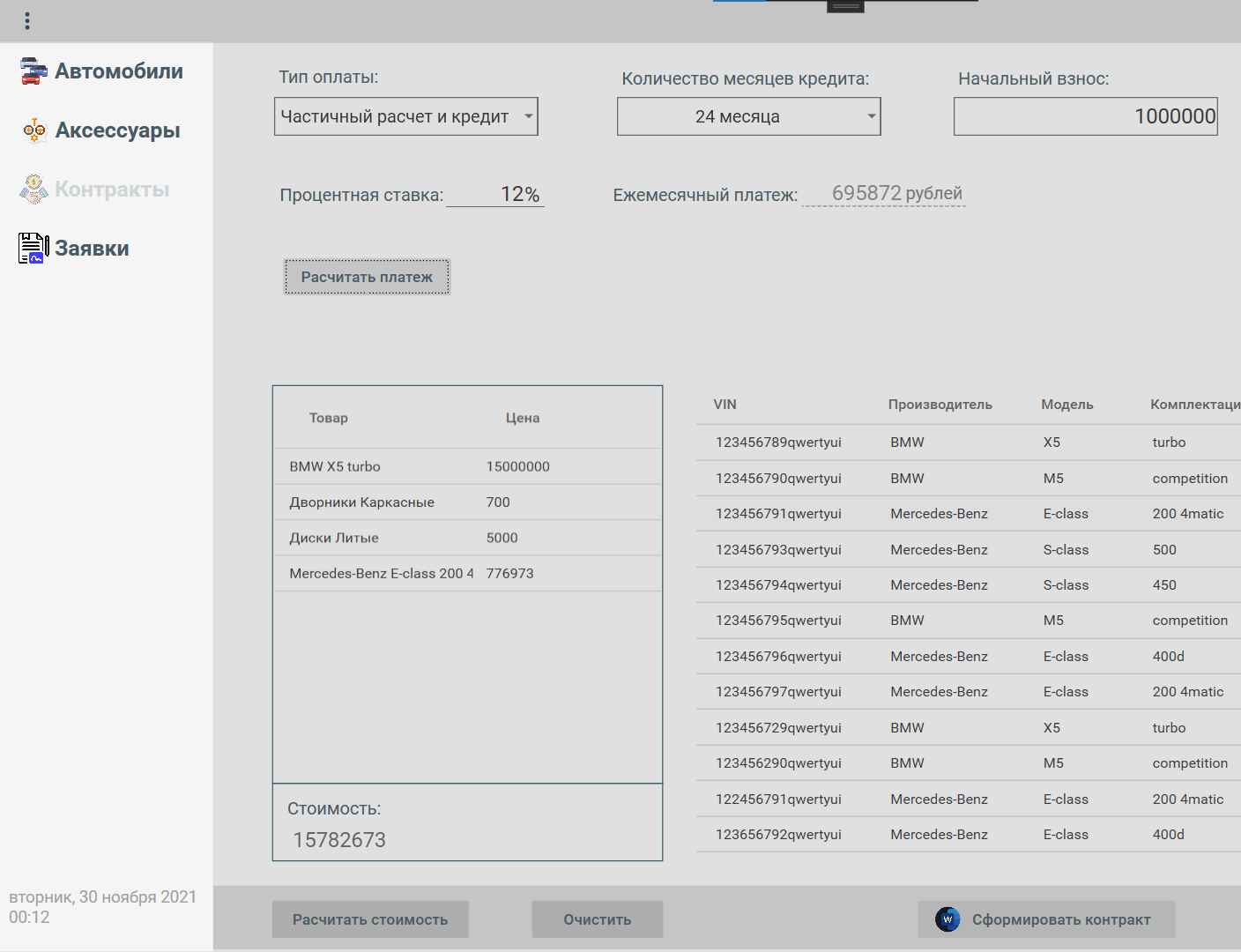


Рисунок 4.6 – Расчет ежемесячного платежа.

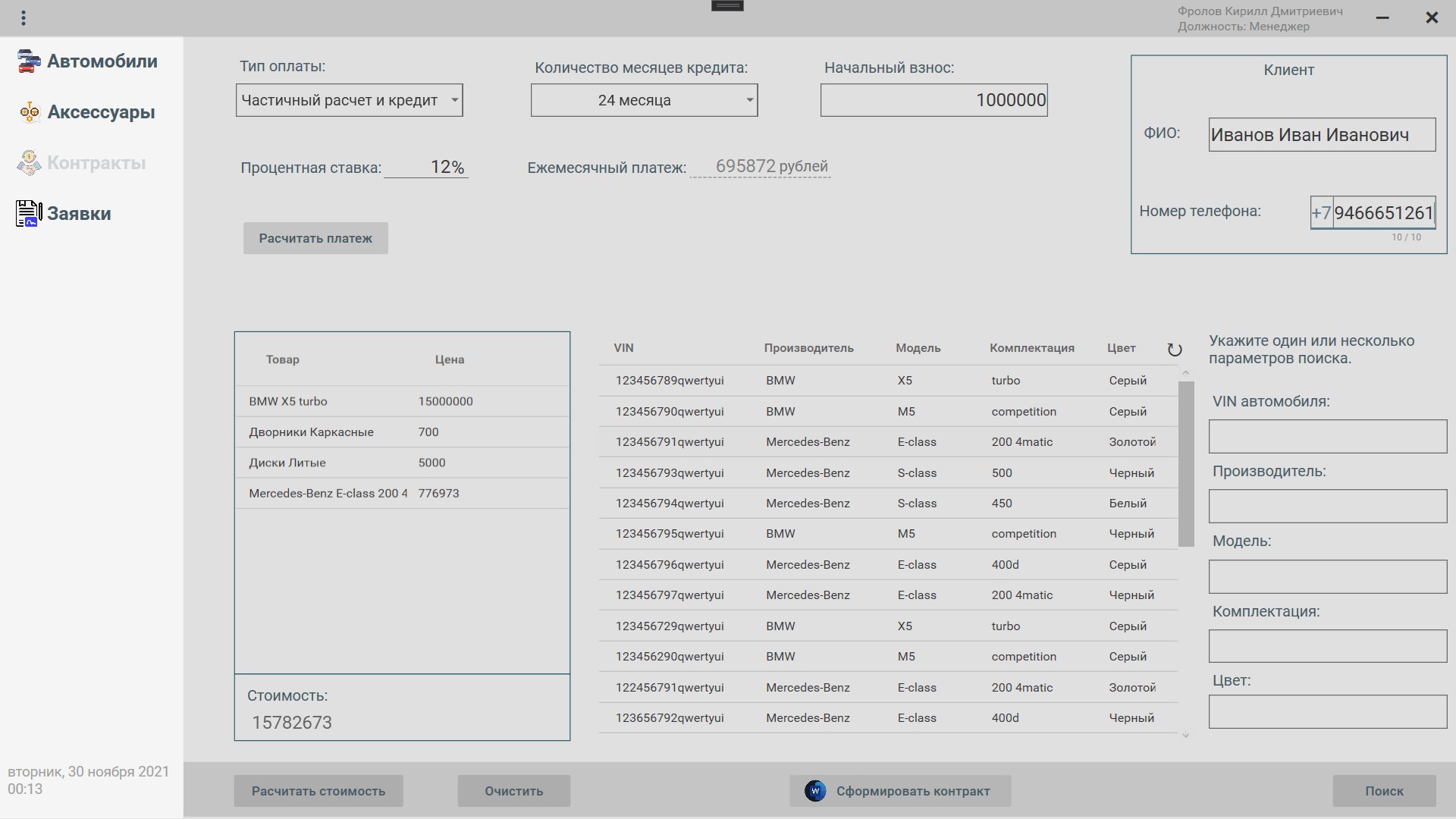


Рисунок 4.7 – Заполнение всех необходимых данных для формирования договора.

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

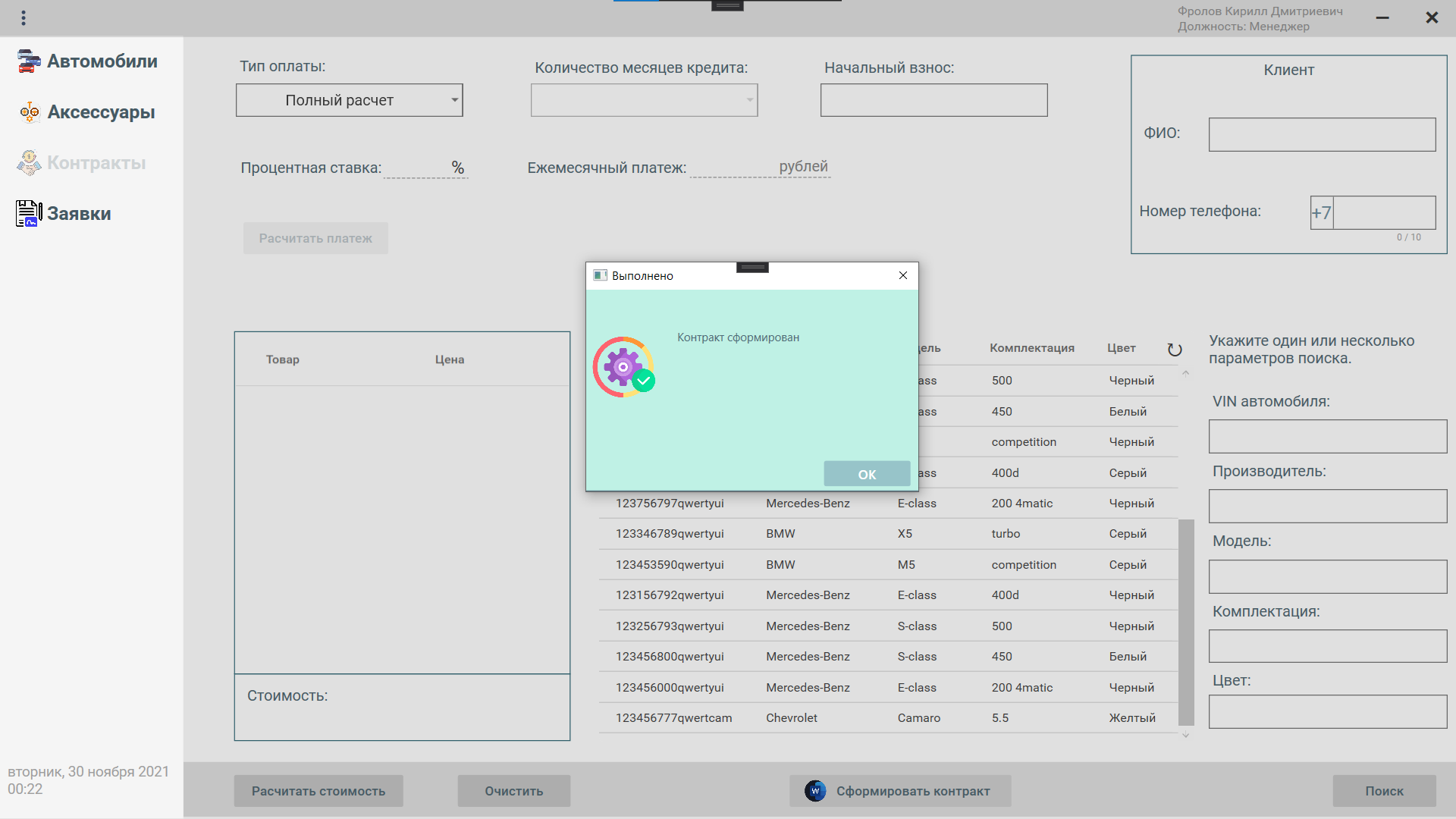


Рисунок 4.8 – Уведомление о сформированном контракте.

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

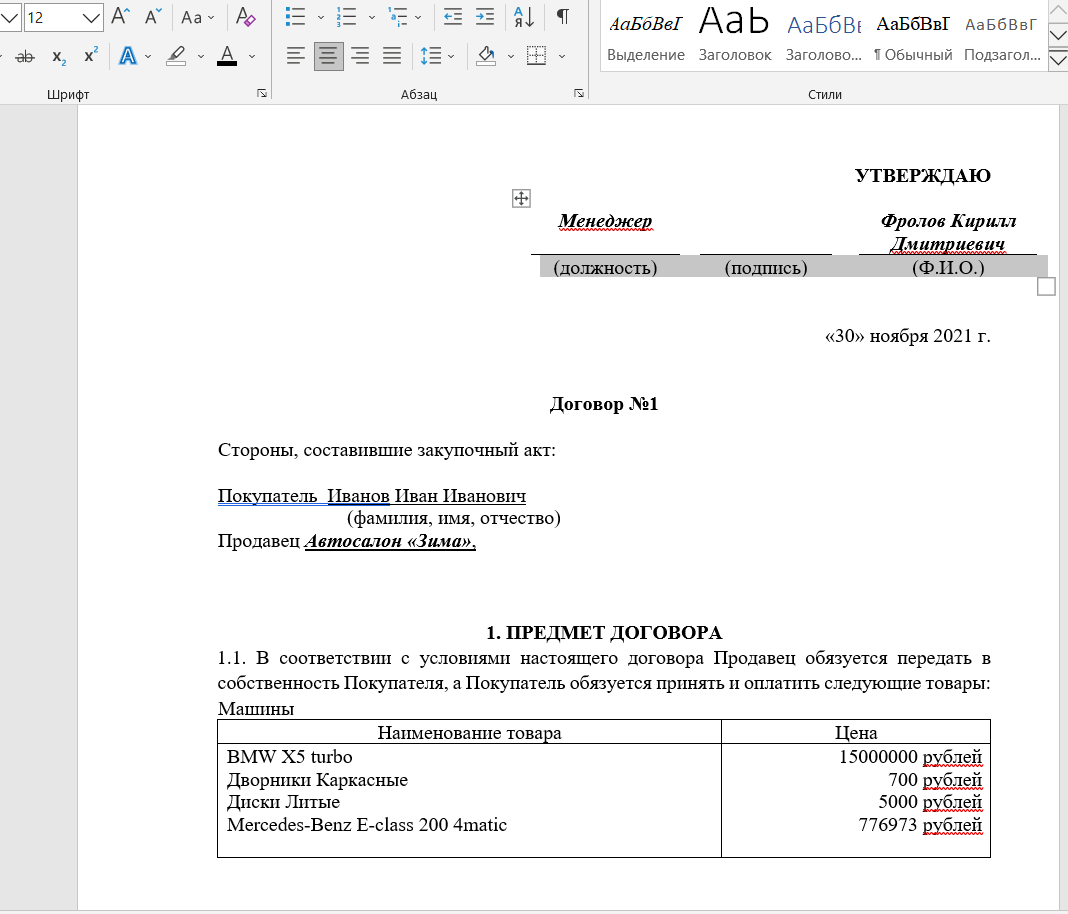


Рисунок 4.9 – Сформированный word документ.

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

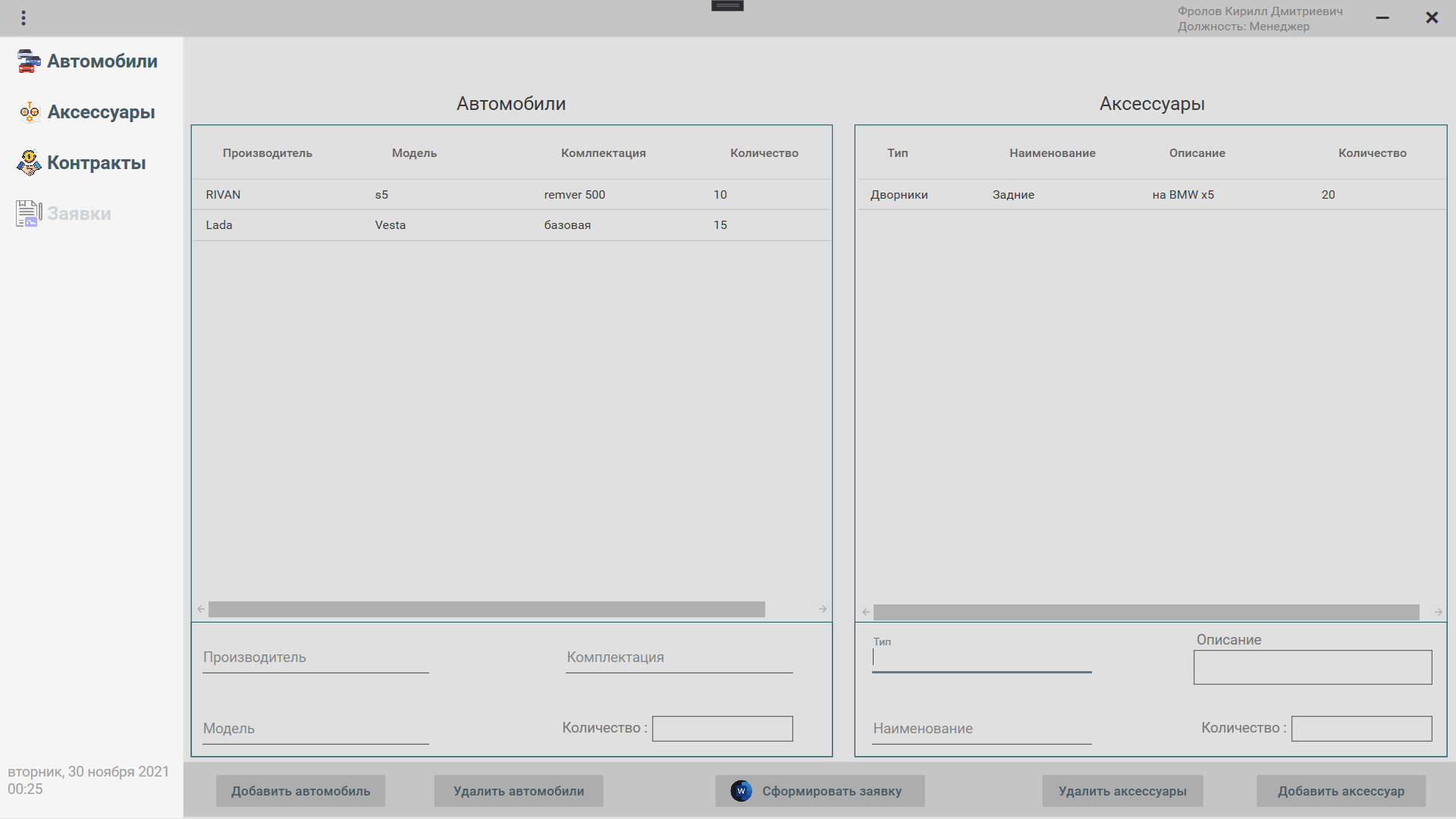


Рисунок 4.10 –Заполнение данных для формирования заявки.

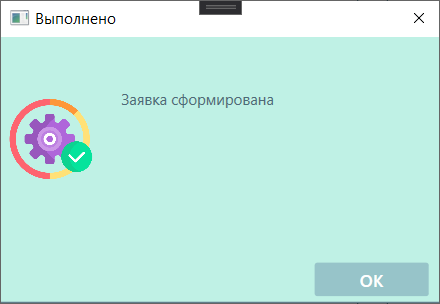


Рисунок 4.11 – Сообщение о сформированной заявке.

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

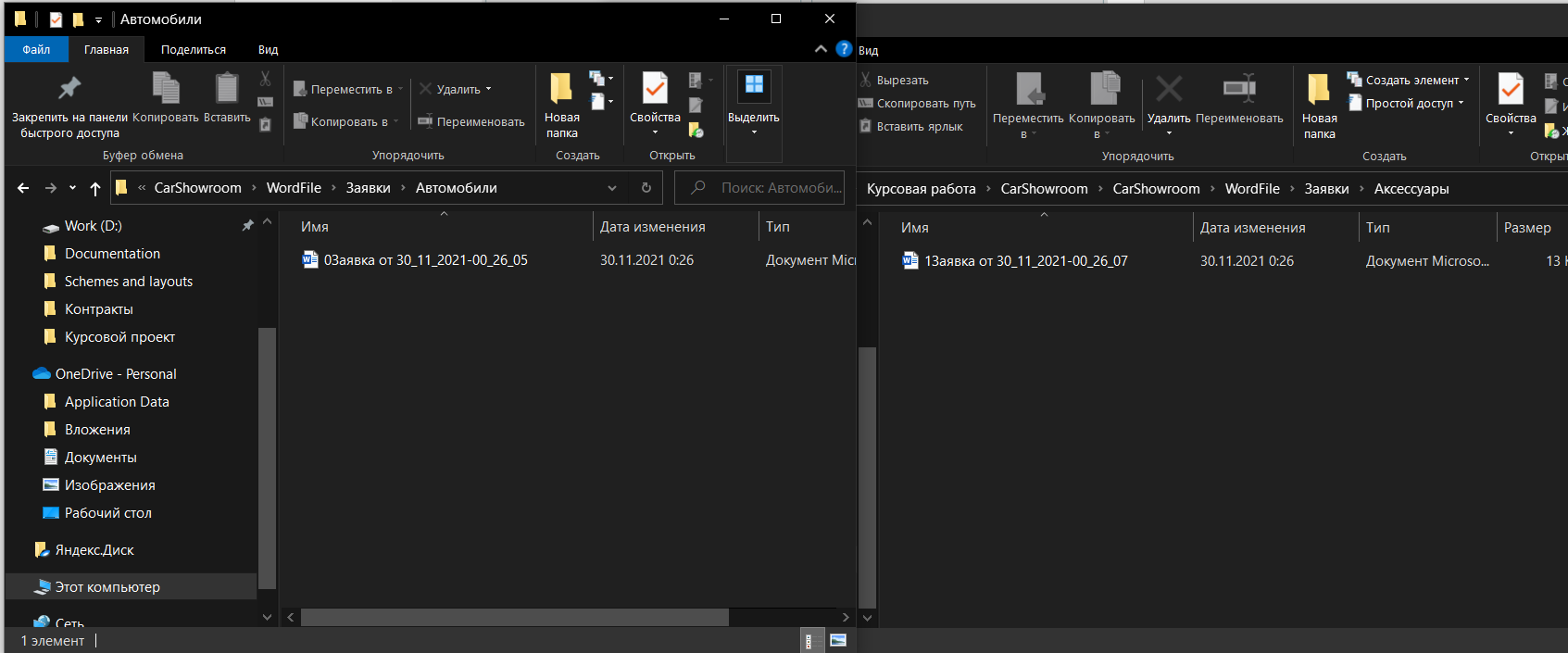


Рисунок 4.12 – 2 сформированных файла для автомобилей и аксессуаров.

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

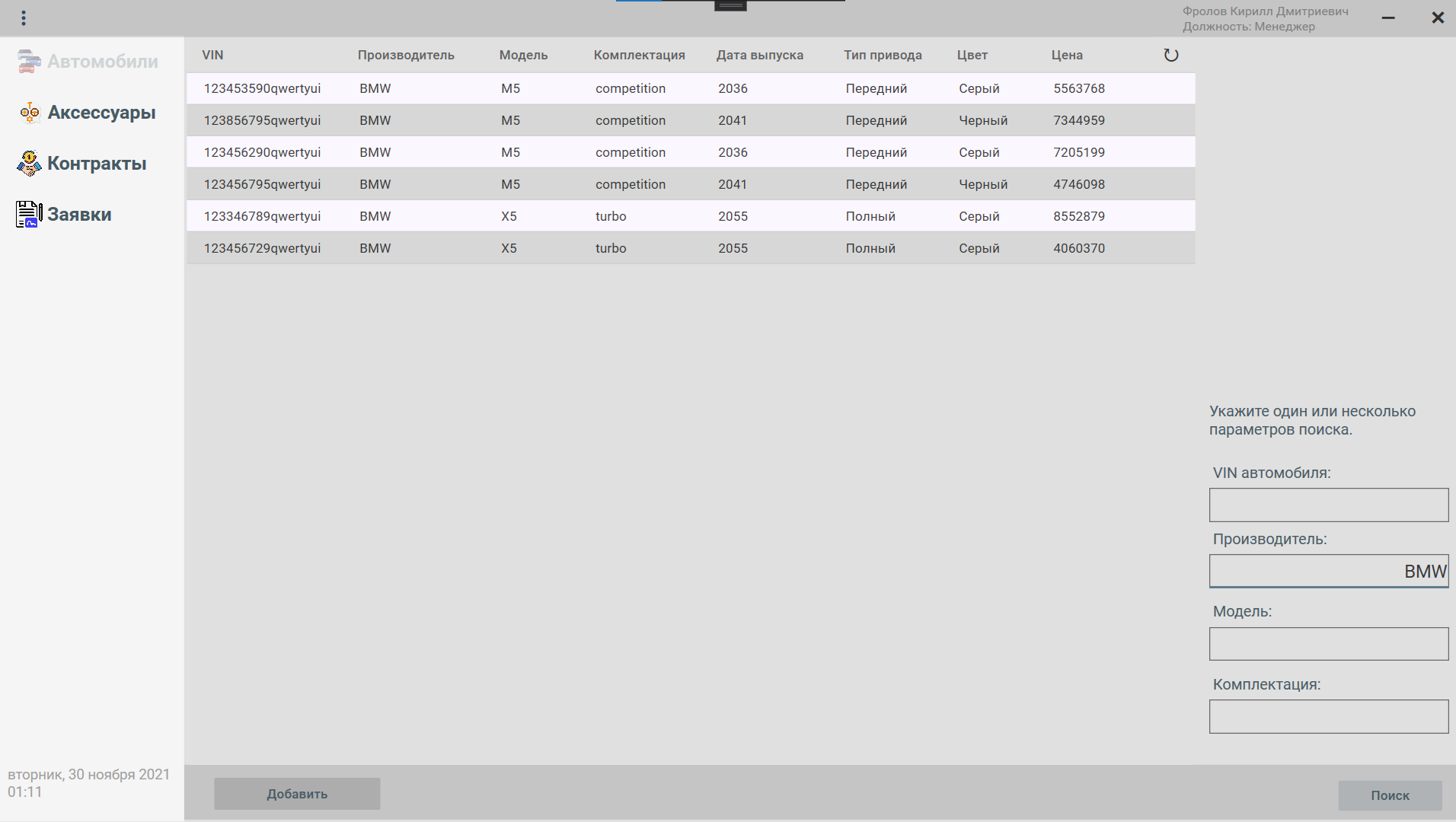


Рисунок 4.13 – Поиск автомобилей.

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.

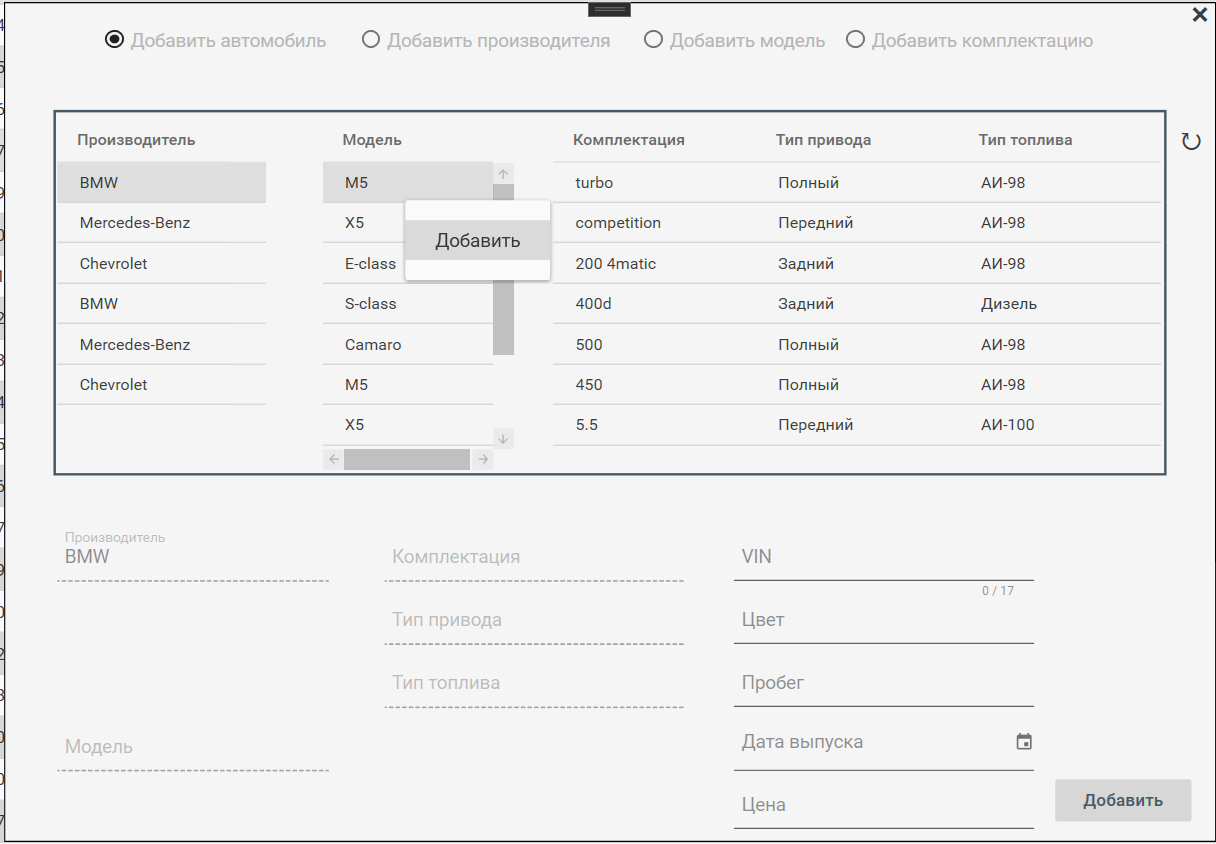


Рисунок 4.14 – Пример добавления автомобиля.

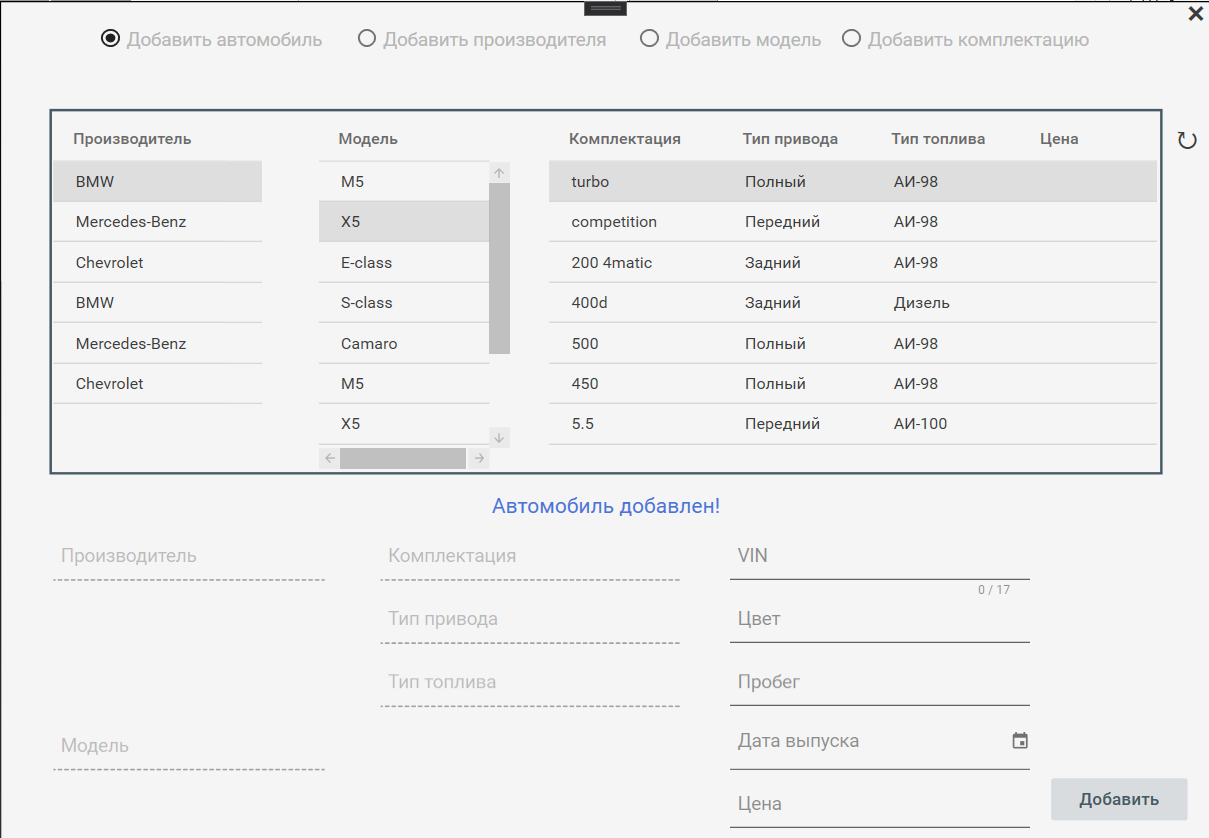


Рисунок 4.15 – Добавление автомобиля произошло успешно.

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

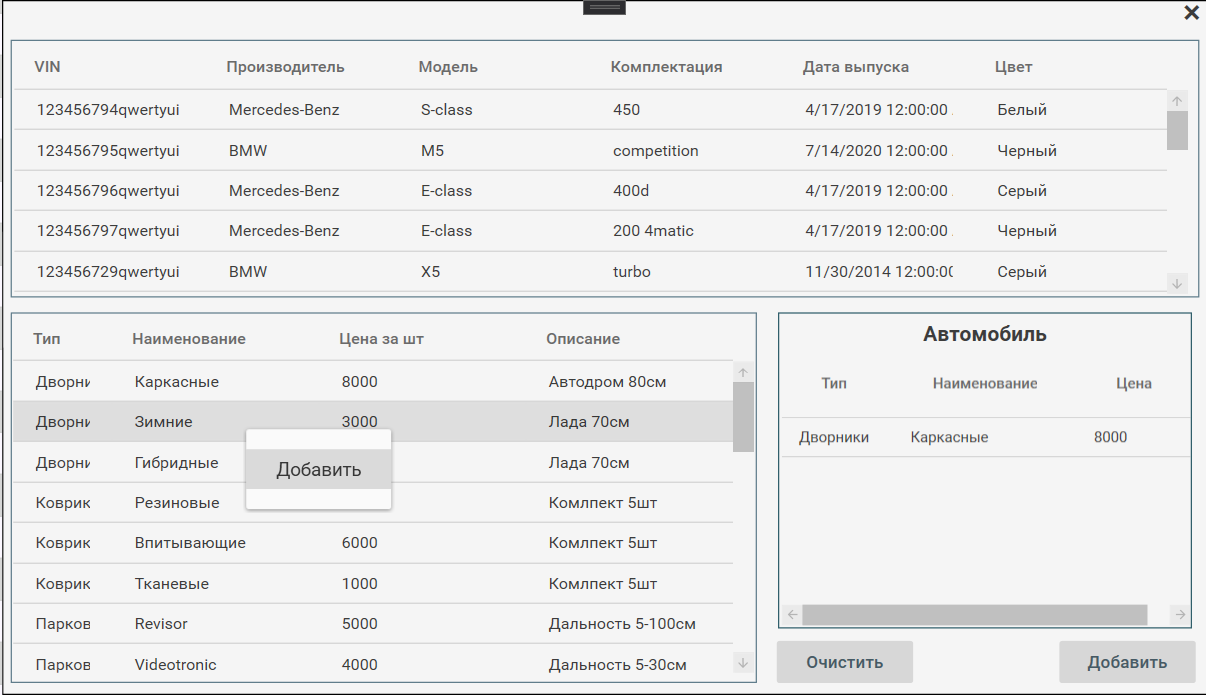


Рисунок 4.16 – Пример добавления аксессуара к автомобилю.



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.

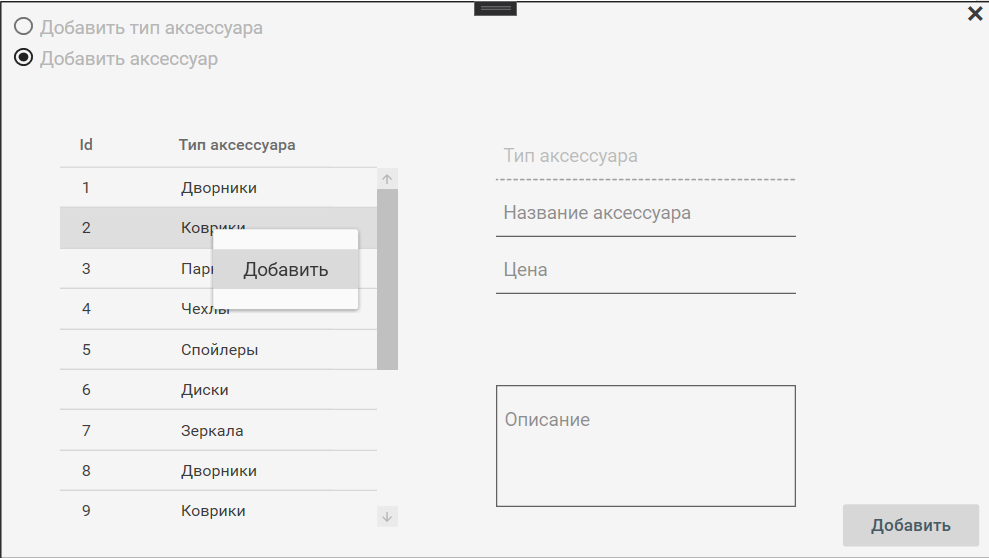


Рисунок 4.18 – Пример добавления аксессуара.

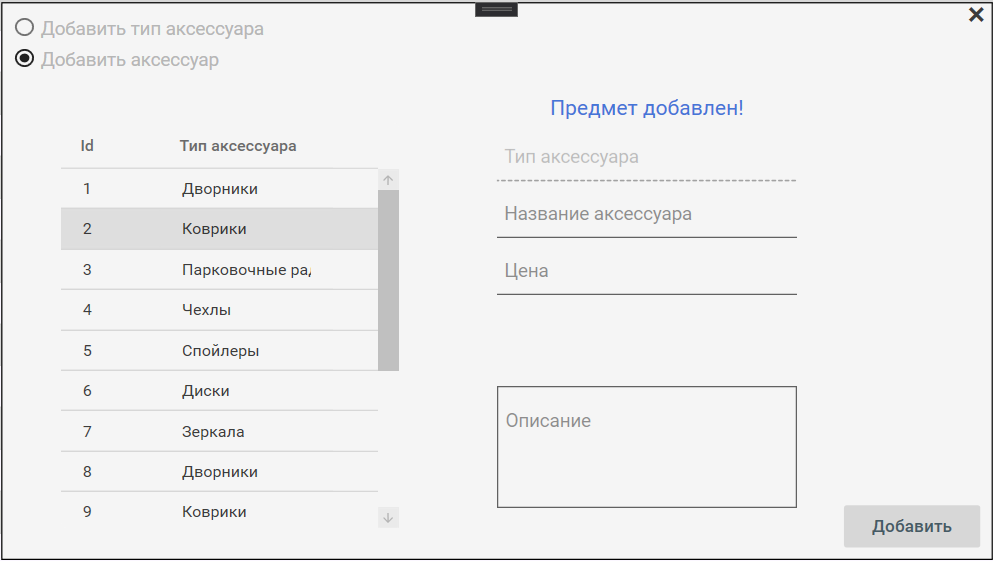


Рисунок 4.19 – Успешное добавление аксессуара.

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.

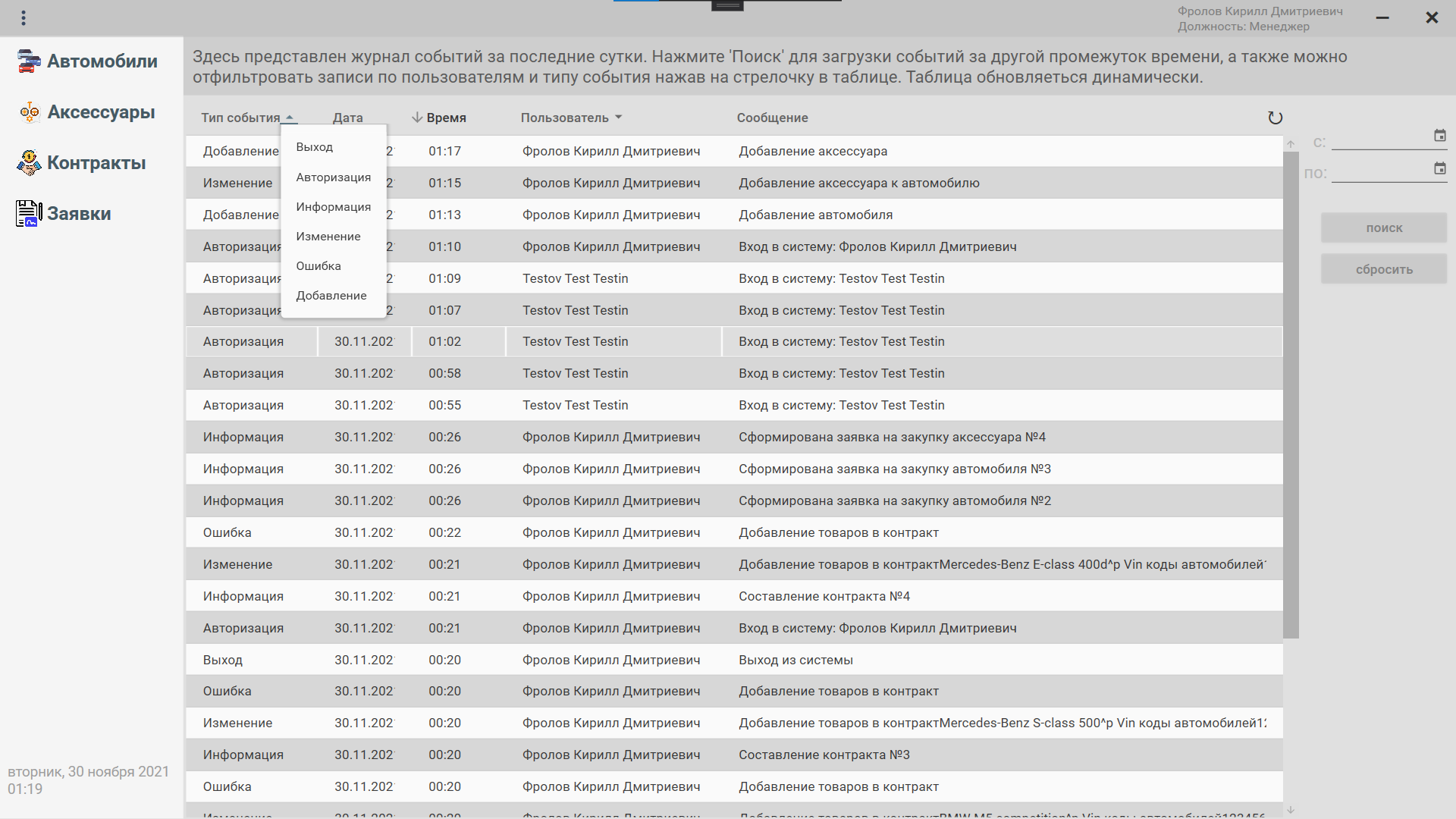


Рисунок 4.20 – Список типов событий в журнале событий.

На рисунке 4.20 был выбран тип события «Информация»

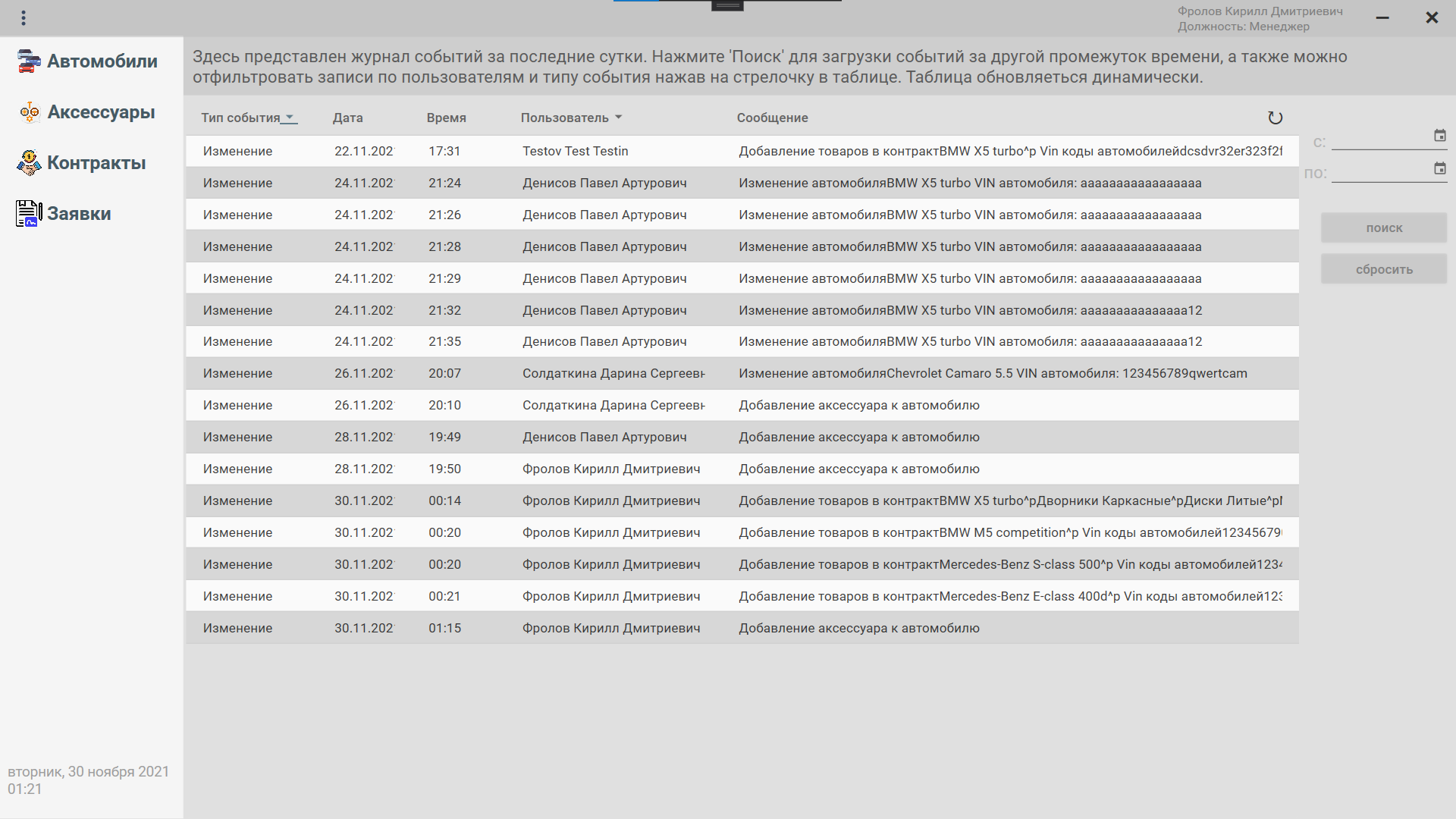


Рисунок 4.20 – Выбор типа события.

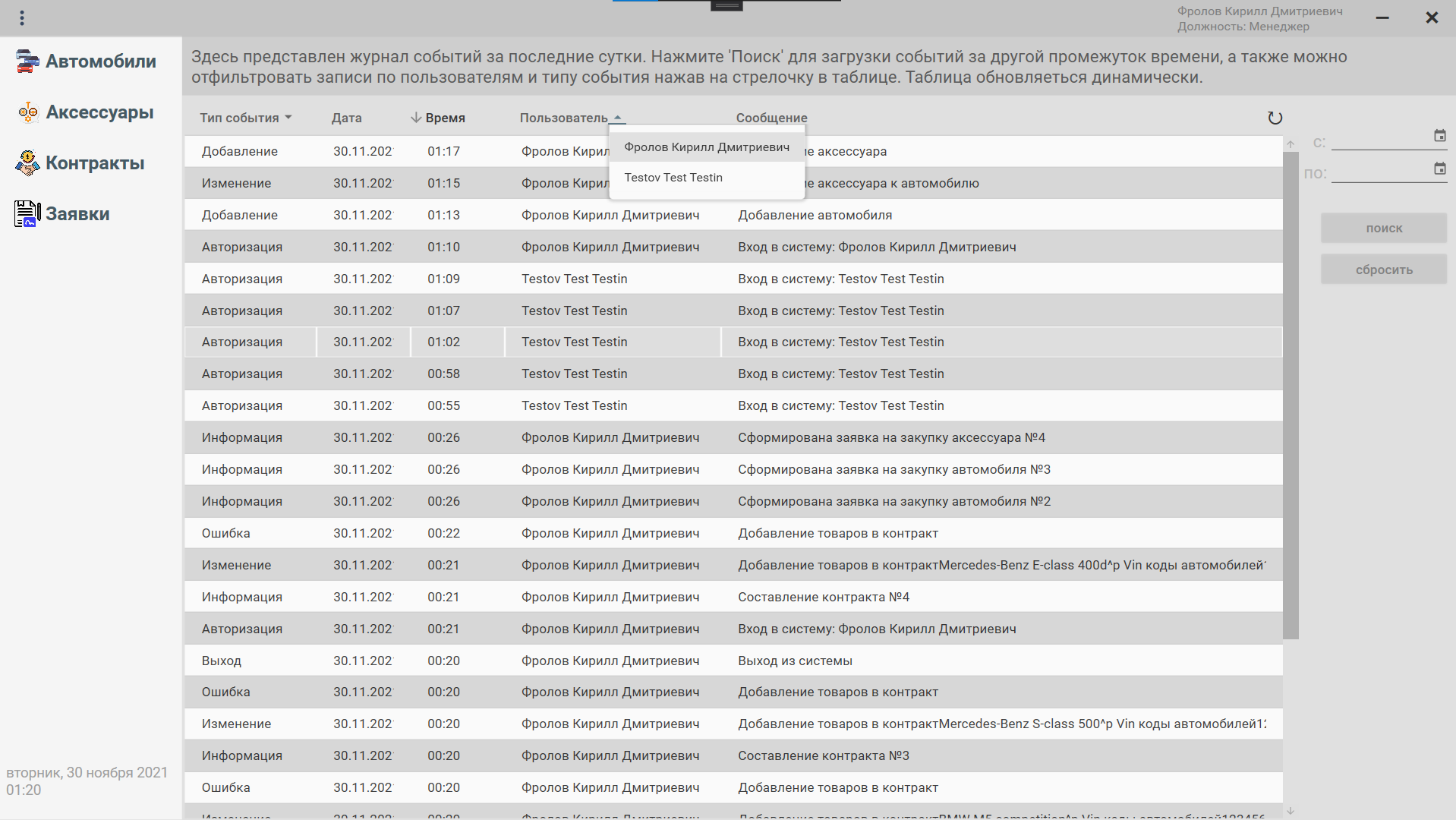


Рисунок 4.20 – Выбор зарегистрированного пользователя.

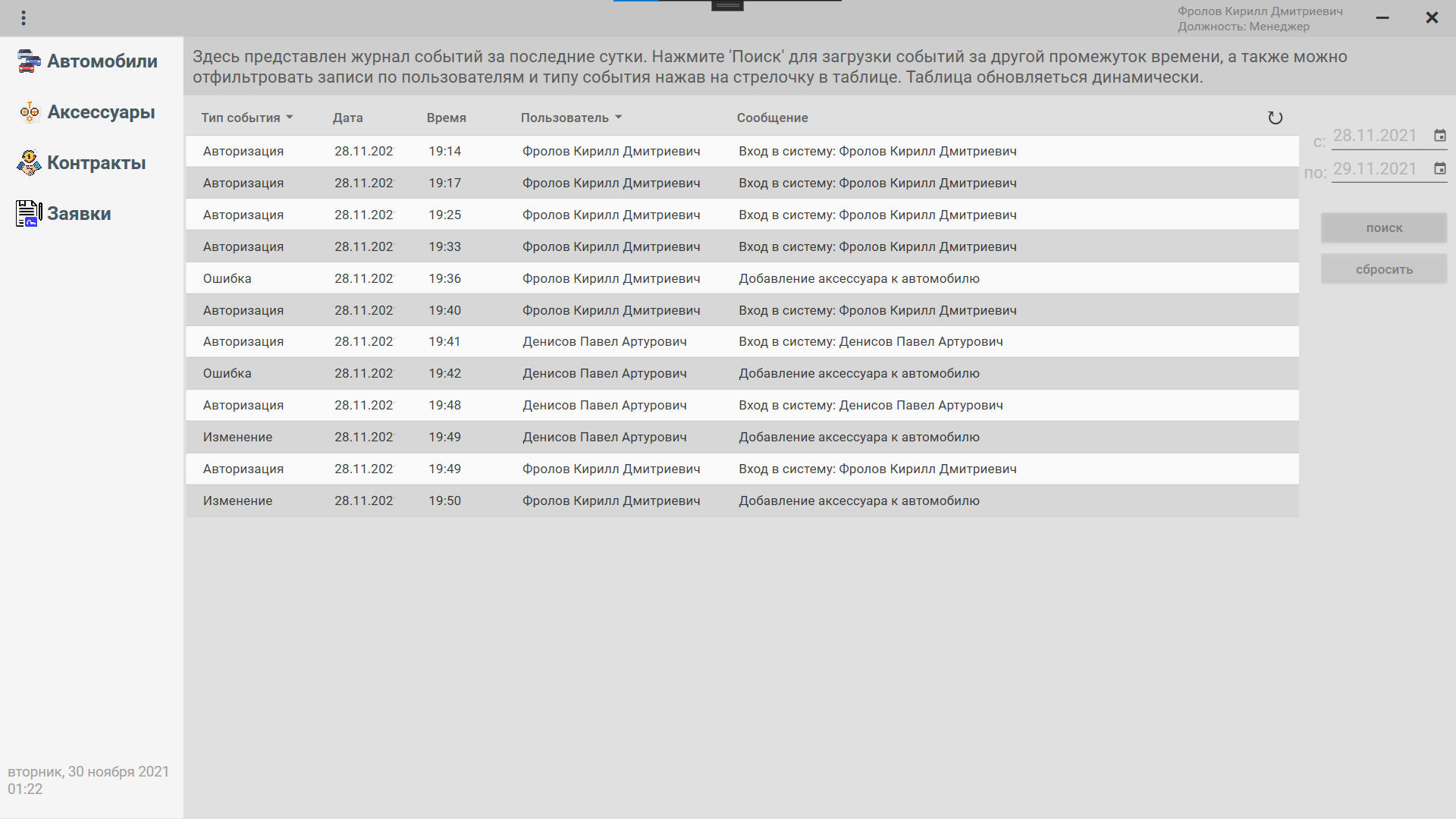


Рисунок 4.20 – Поиск по дате.

# . 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;

}

}

# . Тестирование и отладка рабочей программы.

При попытке если ввести некорректные данные, пользователь будет проинформирован, что данные не совпадают (Рисунок 4.21).

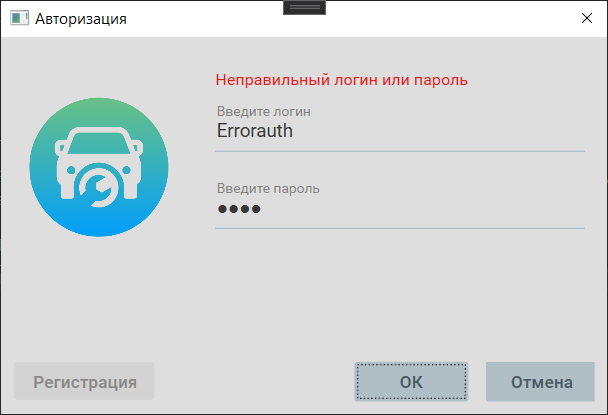


Рисунок 4.21 – Неправильный логин или пароль.

Попытка изменить vin номер на уже существующий приводит к обрабатываемому исключению (Рисунок 4.22).

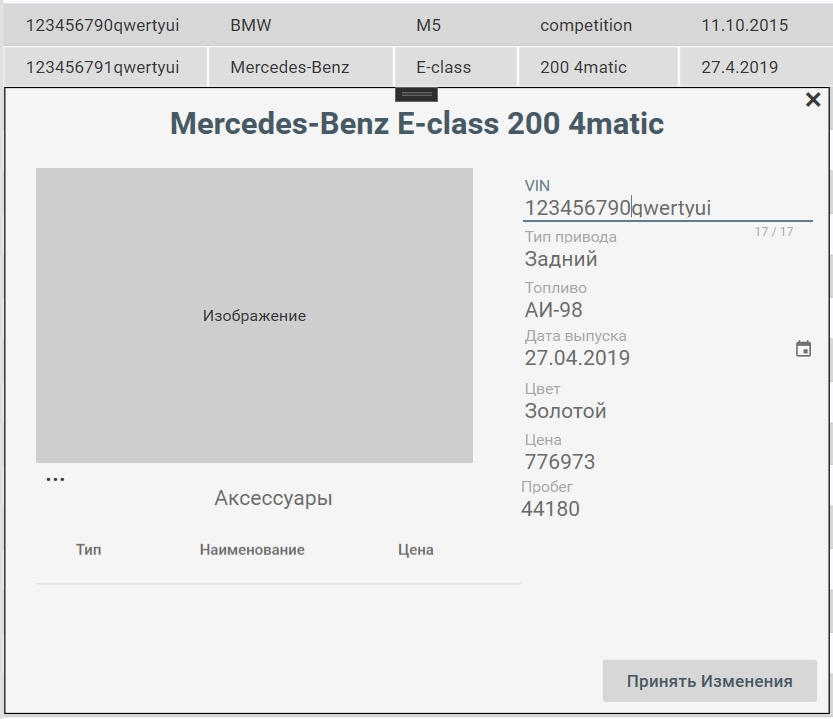


Рисунок 4.22 – Ввод уже существующего vin номера.

При нажатии на кнопку «принять изменения» выводиться сообщение о том, что автомобиль с таким vin номером существует и все изменения сбрасываться (Рисунок 4.23).



Рисунок 4.23 – Ошибка при изменении vin номера.

Попытка ввести некорректную сумму при типе оплаты «Полный расчет», что означает стоимость товаров должна совпадать с первоначальным взносом, а также не должна быть больше.

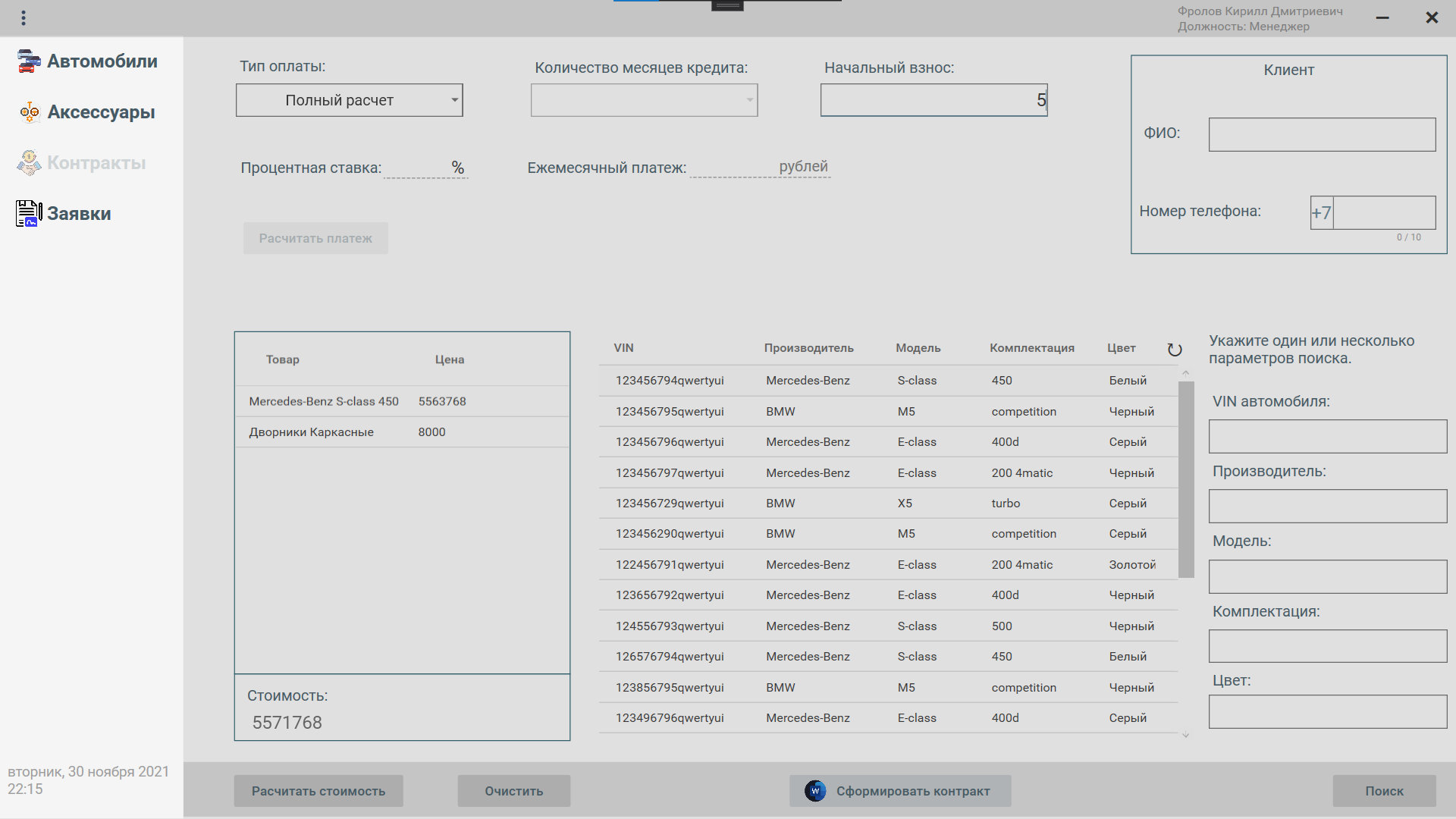


Рисунок 4.24 – Ввод некорректной суммы начального взноса.



Рисунок 4.25 – Ошибка при формировании договора.

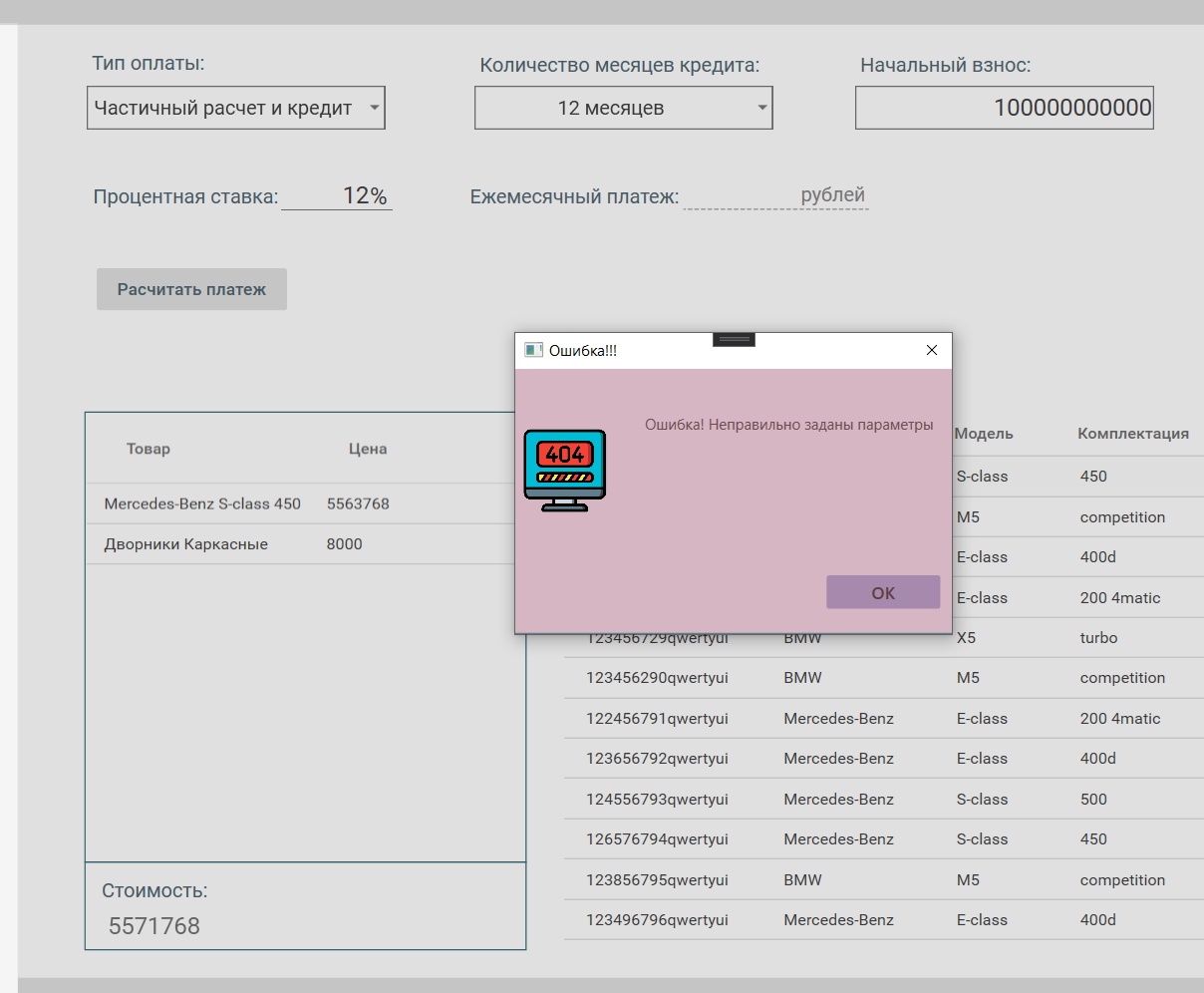


Рисунок 4.26 – Ошибка при расчете ежемесячного платежа.

# . Экспериментальные данные тестирования процедур и функциональных задач.

# 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>

**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));